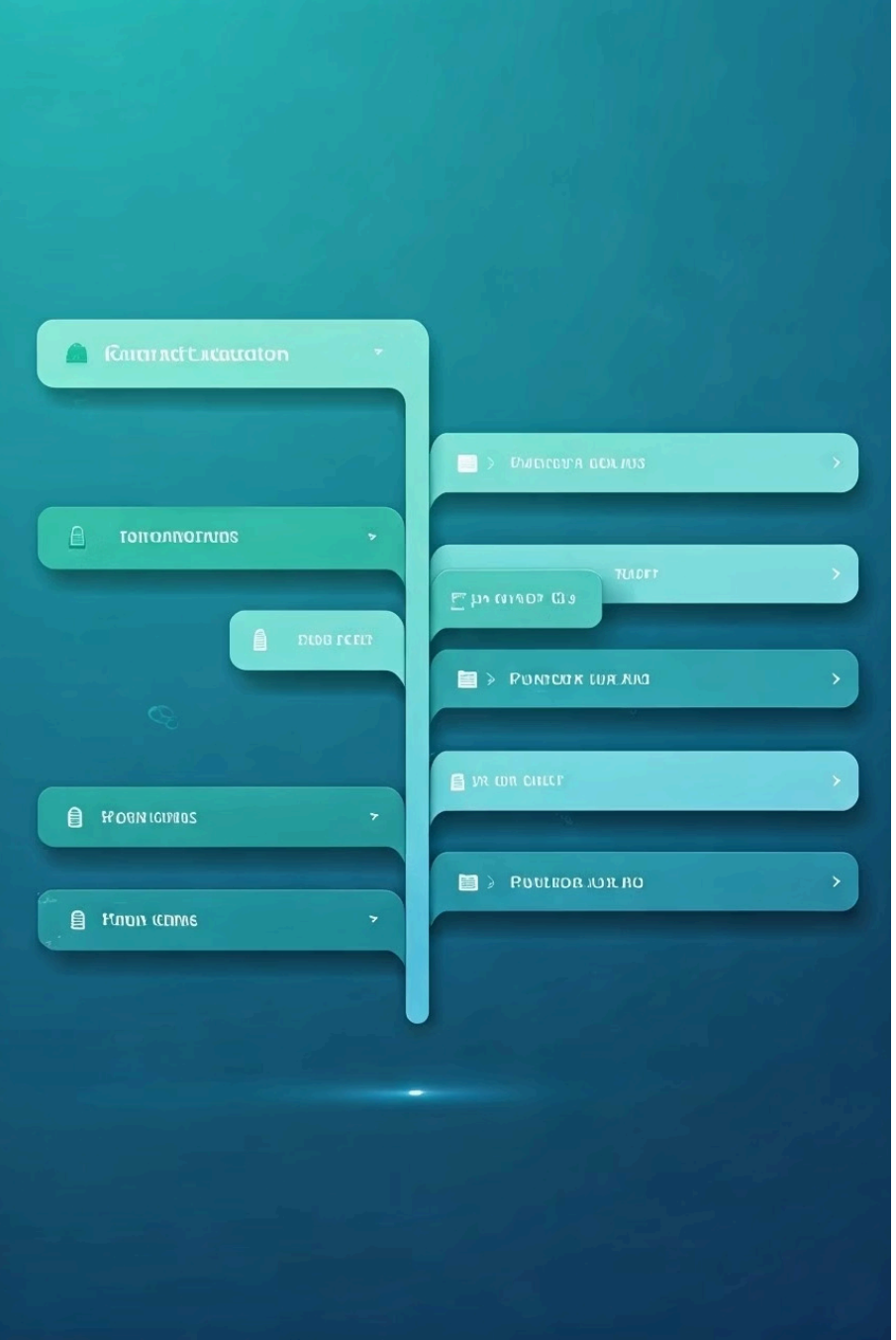


Operaciones Básicas de MongoDB: CRUD y Agregaciones

MongoDB es la base de datos NoSQL líder en el mercado. Utiliza un modelo de documentos con JSON flexible.

Su formato BSON ofrece mayor eficiencia y versatilidad para almacenar datos complejos.

Navegando por Bases de Datos y Colecciones

Mostrar Bases de Datos

Usa el comando **show dbs** para ver todas las bases de datos disponibles.

- Ejemplo: **show dbs** muestra listado como admin, config, local

Seleccionar Base de Datos

Conéctate a una base específica con **use nombreBaseDatos**.

- Ejemplo: **use tienda** cambia a la base "tienda"

Listar Colecciones

Visualiza las colecciones con **show collections**.

- Ejemplo: **show collections** muestra productos, clientes, ventas

Explorar Colección

Visualiza documentos con **db.nombreColeccion.find()**.

- Ejemplo: **db.productos.find().limit(3)** muestra los primeros 3 productos

Crear y Eliminar Bases de Datos y Colecciones

Crear Base de Datos

Usa **use nombreDB** para crear una base de datos. MongoDB la crea automáticamente al insertar el primer documento.

Crear Colección

db.createCollection("nombreColeccion") crea una colección explícitamente con parámetros opcionales.

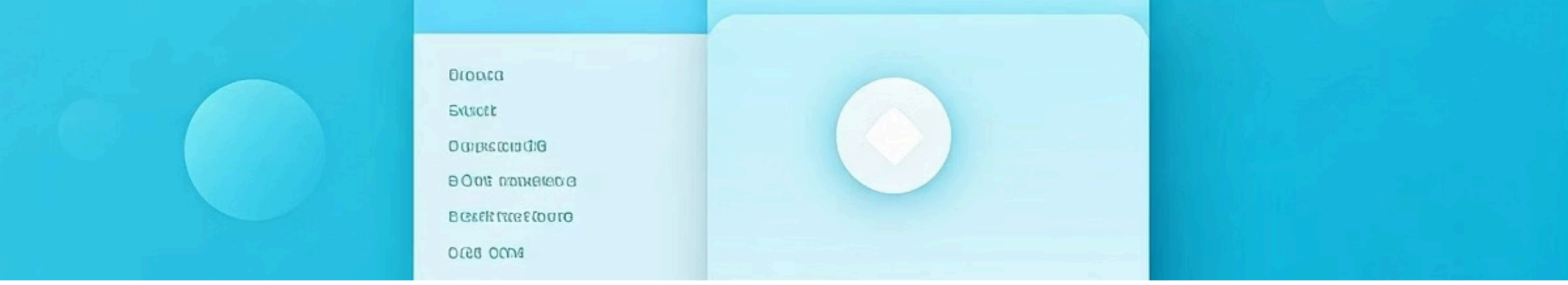
Eliminar Base de Datos

Usa **db.dropDatabase()** mientras estás conectado a la base que deseas eliminar.

Eliminar Colección

db.nombreColeccion.drop() elimina una colección completa con todos sus documentos.

Las colecciones también se crean automáticamente al insertar documentos con **db.nuevaColeccion.insertOne()**.



Crear Documentos: insertOne y insertMany

insertOne()

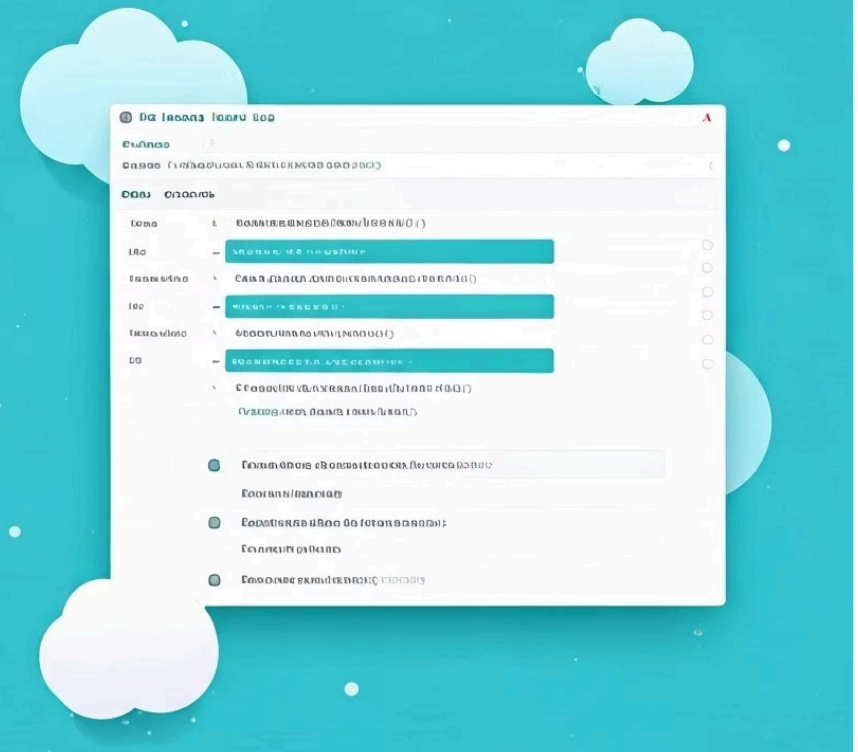
Inserta un único documento en una colección.

insertMany([ ])

Permite crear varios documentos simultáneamente.

Ejemplo Práctico

db.usuarios.insertOne({nombre: "Ana", edad: 28, email: "ana@mail.com"})



Leer Documentos: find y Filtros

find({})

Retorna todos los documentos de una colección.

Filtrado con Operadores

Utiliza $eq, $gt, $in y $and para consultas específicas.

Ejemplo

db.usuarios.find({edad: {$gt: 25}})

Actualizar Documentos: updateOne y updateMany

updateOne()

Modifica el primer documento que coincide con el filtro especificado.

Solo afecta a un documento, incluso si varios cumplen la condición.

updateMany()

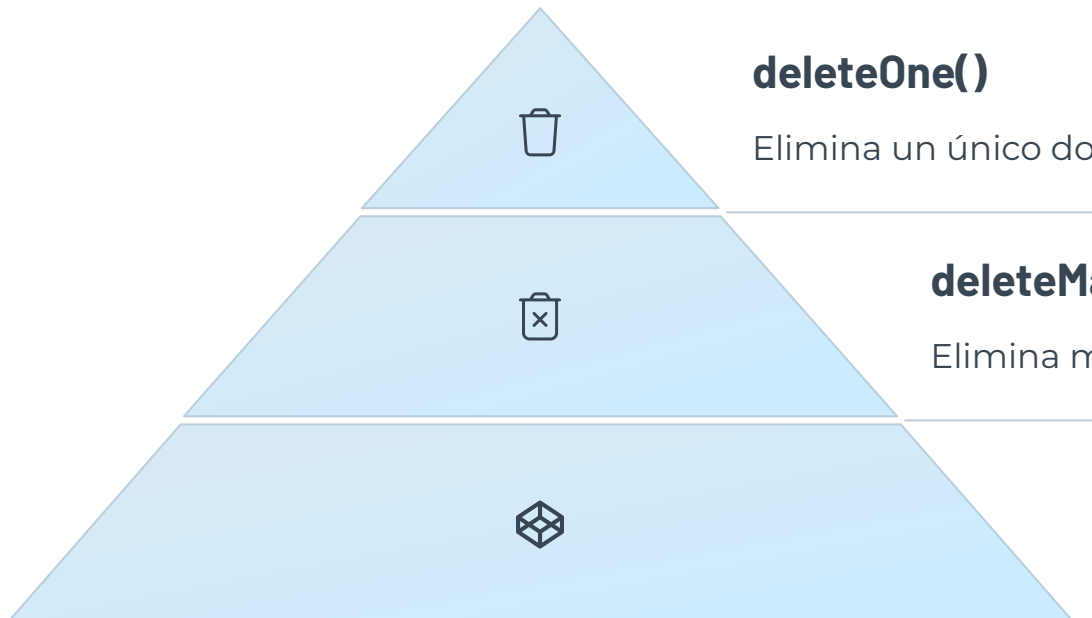
Actualiza todos los documentos que cumplen con el criterio de filtrado.

Ideal para actualizaciones masivas en la base de datos.

Operador $set

db.usuarios.updateOne({nombre: "Ana"}, {$set: {email: "ana.nueva@mail.com"}})

Eliminar Documentos: deleteOne y deleteMany



deleteOne()

Elimina un único documento según el filtro

deleteMany()

Elimina múltiples documentos que cumplen el criterio

Ejemplo Práctico

db.usuarios.deleteMany({estado: "inactivo"})

Agregaciones con Pipeline: match, group y sort

$match

Filtra documentos que cumplen con criterios específicos.

$group

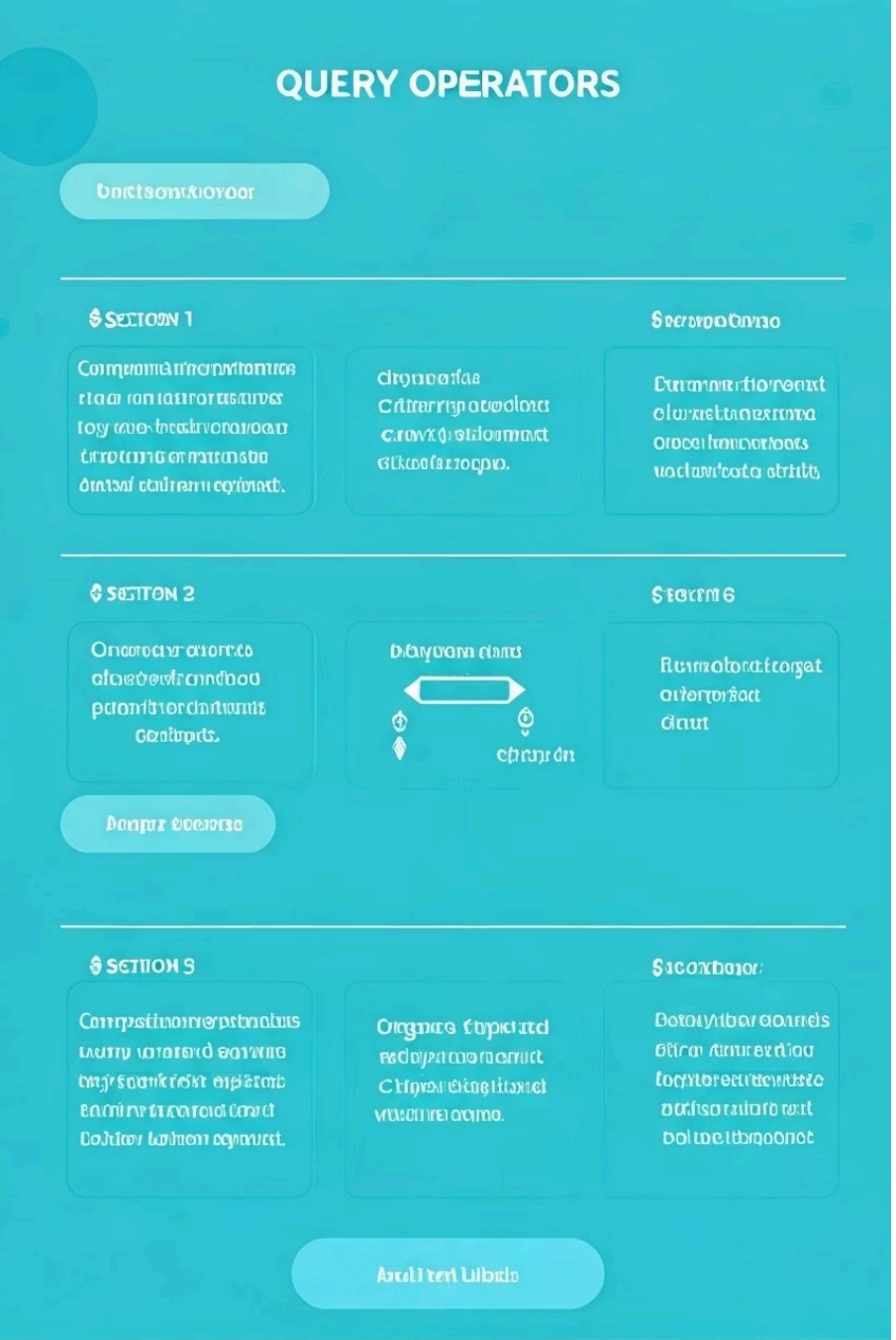
Agrupa documentos y calcula valores agregados.

Pipeline

Encadena operaciones para análisis avanzado de datos.

$sort

Ordena los resultados según los campos especificados.



Operadores Clave en MongoDB

Operador	Función	Ejemplo
$eq	Igual a	{ campo: { $eq: valor } }
$gt	Mayor que	{ campo: { $gt: valor } }
$in	En conjunto	{ campo: { $in: [v1, v2] } }
$and	Combina condiciones	{ $and: [cond1, cond2] }

Más Operadores Esenciales de MongoDB

Además de los operadores básicos, MongoDB ofrece una amplia gama de operadores para consultas avanzadas y manipulación de datos.

Operador	Función	Ejemplo
$lt	Menor que	{ campo: { $lt: valor } }
$lte	Menor o igual que	{ campo: { $lte: valor } }
$gte	Mayor o igual que	{ edad: { $gte: 18 } }
$ne	No igual a	{ estado: { $ne: "inactivo" } }
$or	O lógico	{ $or: [{condición1}, {condición2}] }
$nin	No en conjunto	{ tipo: { $nin: ["A", "B"] } }
$exists	Campo existe	{ email: { $exists: true } }
$regex	Coincide con patrón	{ nombre: { $regex: /^A/ } }

Estos operadores pueden combinarse para crear consultas potentes y precisas según tus necesidades específicas.

Operadores Útiles para Actualización de Documentos

$inc

Incrementa un valor numérico en una cantidad específica.

db.productos.updateOne({_id: 123}, {$inc: {stock: -1}})

$mul

Multiplica el valor actual por un número específico.

db.ventas.updateMany({promocion: true}, {$mul: {precio: 0.9}})

$push

Añade elementos a un array existente.

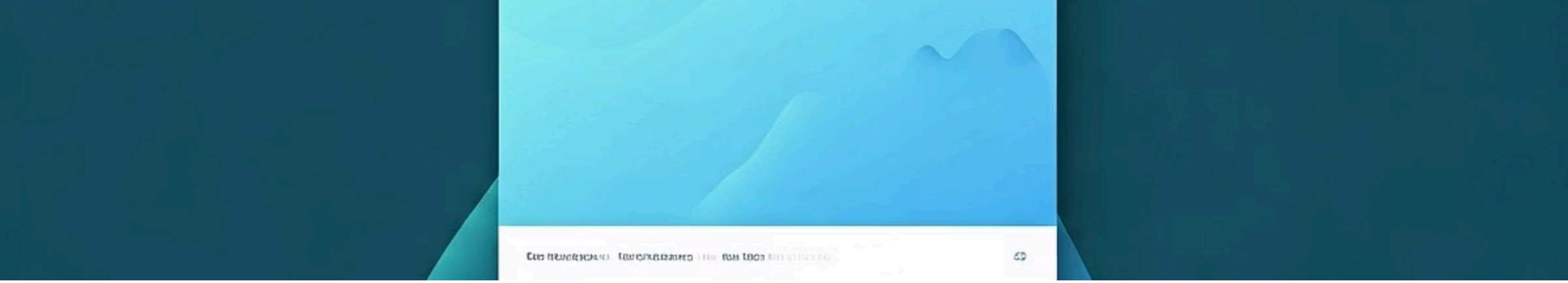
db.usuarios.updateOne({_id: "ana"}, {$push: {intereses: "MongoDB"}})

$pull

Elimina elementos de un array que cumplan cierta condición.

db.pedidos.updateOne({_id: 456}, {$pull: {productos: {id: 789}}})

Estos operadores son esenciales para manipulaciones precisas de documentos sin necesidad de reemplazarlos por completo, manteniendo la integridad y eficiencia de la base de datos.



Ejemplos Prácticos y Herramientas MongoDB

Insertar Usuario

db.usuarios.insertOne({nombre: "Carlos", edad: 32})

Buscar Mayores de 25

db.usuarios.find({edad: {$gt: 25}})

Actualizar Email

db.usuarios.updateOne({nombre: "Carlos"}, {$set: {email: "carlos@mail.com"}})

Calcular Promedio

db.usuarios.aggregate([{$group: {_id: "$categoria", promEdad: {$avg: "$edad"}}}])

Ejemplos Avanzados: Aggregate, InsertMany y UpdateMany

Inserción Múltiple

db.productos.insertMany([{nombre: "Laptop", precio: 1200, stock: 15}, {nombre: "Smartphone", precio: 800, stock: 25}])

Actualización Condicional

db.productos.updateMany({stock: {$lt: 20}}, {$set: {estado: "reposición"}, $inc: {precio: 50}})

Agregación Multinivel

db.ventas.aggregate([{$match: {fecha: {$gte: new Date("2023-01-01")}}}, {$group: {_id: "$categoria", total: {$sum: "$monto"}}}, {$sort: {total: -1}}])

Las operaciones avanzadas permiten manipulaciones complejas de datos en MongoDB. Puedes combinar múltiples etapas y condiciones para análisis sofisticados.