Batería Completa de 20 Ejercicios: MongoDB con Node.js

Prácticas de Bases de Datos NoSQL

1. Bloque 1: Inserciones y Eliminaciones (Ejercicios 1-4)

Ejercicio 1: Insertar un solo documento (InsertOne)
Inserta un nuevo libro titulado *El alquimista* de Paulo Coelho, con 192 páginas, géneros *Ficción* y *Filosofía*, stock de 10 unidades, puntuación de 8.2 y detalles de tapa blanda en español.

```
await col.insertOne({
    titulo: "El alquimista",
    autor: "Paulo Coelho",
    paginas: 192,
    generos: ["Ficción", "Filosofía"],
    stock: 10,
    detalles: { idioma: "Español", tapa: "Blanda" },
    puntuacion: 8.2
});
```

Ejercicio 2: Insertar múltiples documentos (InsertMany)
Inserta de golpe dos libros descatalogados con stock 0 y sin puntuación definida.

```
await col.insertMany([
    { titulo: "Libro Viejo A", autor: "Anónimo", stock: 0 },
    { titulo: "Libro Viejo B", autor: "Anónimo", stock: 0 }
]);
```

Ejercicio 3: Eliminar un solo documento (DeleteOne)
Elimina de la base de datos el libro con el título exacto "Libro Viejo A".

```
await col.deleteOne({ titulo: "Libro Viejo A" });
```

Ejercicio 4: Eliminar múltiples documentos (DeleteMany)
Elimina todos los libros que tengan un stock de 0 y cuyo autor sea "Anónimo".

```
await col.deleteMany({ stock: 0, autor: "Anónimo" });
```

2. Bloque 2: Consultas, Proyecciones y Paginación (Ejercicios 5-11)

Ejercicio 5: Proyección básica (Inclusión y Exclusión)
Obtén el título y autor de todos los libros. Oculta el campo `_id`.

```
// Opción A: Objeto Options
const resA = await col.find({}, { projection: { titulo: 1, autor: 1, _id: 0 } }).toArray();

// Opción B: Métodos Encadenados
const resB = await col.find({}).project({ titulo: 1, autor: 1, _id: 0 }).toArray();
```

Ejercicio 6: Ordenación (Sort)
Obtén todos los libros ordenados por número de páginas de forma descendente.

```
// Opción A: Objeto Options
const resA = await col.find({}, { sort: { paginas: -1 } }).toArray();

// Opción B: Métodos Encadenados
const resB = await col.find({}).sort({ paginas: -1 }).toArray();
```

Ejercicio 7: Limitación de resultados (Limit)
Obtén únicamente los 3 libros con la peor puntuación registrada.

```
// Opción A: Objeto Options
const resA = await col.find({}, { sort: { puntuacion: 1 }, limit: 3 }).toArray();

// Opción B: Métodos Encadenados
const resB = await col.find({}).sort({ puntuacion: 1 }).limit(3).toArray();
```

Ejercicio 8: Paginación completa (Skip + Limit + Sort)
Implementa una paginación para obtener la tercera página de libros (4 elementos por página), ordenados alfabéticamente por autor.

```
// Opción A: Objeto Options (Saltamos 8 elementos para ir a la página 3)
const resA = await col.find({}, { sort: { autor: 1 }, skip: 8, limit: 4 }).toArray();

// Opción B: Métodos Encadenados
const resB = await col.find({}).sort({ autor: 1 }).skip(8).limit(4).toArray();
```

Ejercicio 9: Filtrado por campos anidados
Encuentra todos los libros editados en formato de tapa *Dura*.

```
const res = await col.find({ "detalles.tapa": "Dura" }).toArray();
```

Ejercicio 10: Operadores lógicos y comparación ($gt, $lte, $and)
Busca libros que tengan entre 300 y 600 páginas (inclusive ambos) y una puntuación mayor a 9.0.

```
// Opción Encadenada con Proyección limpia
const res = await col.find({
    paginas: { $gte: 300, $lte: 600 },
```

```
    puntuacion: { $gt: 9.0 }
}).project({ titulo: 1, _id: 0 }).toArray();
```

Ejercicio 11: Contar documentos (CountDocuments)
Cuenta cuántos libros tienen un stock crítico inferior a 5 unidades (sin incluir el 0).

```
const totalCritico = await col.countDocuments({ stock: { $gt: 0, $lt: 5 } });
```

3. Bloque 3: Actualizaciones Avanzadas (Ejercicios 12-15)

Ejercicio 12: Actualización simple de valor ($set)
Cambia la puntuación del libro *1984* a un valor perfecto de 10.0.

```
await col.updateOne({ titulo: "1984" }, { $set: { puntuacion: 10.0 } });
```

Ejercicio 13: Incremento numérico ($inc)
Suma 5 unidades al stock de todos los libros cuyo idioma sea *Inglés*.

```
await col.updateMany({ "detalles.idioma": "Inglés" }, { $inc: { stock: 5 } });
```

Ejercicio 14: Añadir elementos a un array sin duplicar ($addToSet)
Añade el género *Culto* al libro *Rayuela*, asegurando que no se duplique si ya existía.

```
await col.updateOne({ titulo: "Rayuela" }, { $addToSet: { generos: "Culto" } });
```

Ejercicio 15: Eliminar campos de un documento ($unset)
Elimina por completo el campo `puntuacion` de aquellos libros que tengan un stock igual a 0.

```
await col.updateMany({ stock: 0 }, { $unset: { puntuacion: "" } });
```

4. Bloque 4: Framework de Agregación (Ejercicios 16-20)

Ejercicio 16: Agregación básica ($match + $group)
Calcula cuántos libros hay acumulados en stock agrupándolos por idioma.

```
const pipeline = [
    { $group: { _id: "$detalles.idioma", stockTotal: { $sum: "$stock" } } }
];
const res = await col.aggregate(pipeline).toArray();
```

Ejercicio 17: Filtrado y obtención de promedios ($match + $group + $avg)
Obtén la puntuación media de los libros de tapa *Blanda* que superen las 200 páginas.

```
const pipeline = [
    { $match: { "detalles.tapa": "Blanda", paginas: { $gt: 200 } } },
    { $group: { _id: "$detalles.tapa", puntuacionMedia: { $avg: "$puntuacion" } } }
];
const res = await col.aggregate(pipeline).toArray();
```

Ejercicio 18: Descompresión de arrays y conteo ($unwind + $group + $sort)
Descompón los géneros de los libros para saber cuántos libros individuales pertenecen a cada género. Ordena los resultados de mayor a menor frecuencia.

```
const pipeline = [
    { $unwind: "$generos" },
    { $group: { _id: "$generos", totalLibros: { $sum: 1 } } },
    { $sort: { totalLibros: -1 } }
];
const res = await col.aggregate(pipeline).toArray();
```

Ejercicio 19: Proyecciones en pipelines e hilos de cálculo ($project)
Genera un listado donde muestres el título, y un nuevo campo calculado llamado `valorStock` (resultado de multiplicar `stock * paginas`) solo para libros con stock disponible.

```
const pipeline = [
    { $match: { stock: { $gt: 0 } } },
    { $project: { _id: 0, titulo: 1, valorStock: { $multiply: ["$stock", "$paginas"] } } }
];
const res = await col.aggregate(pipeline).toArray();
```

Ejercicio 20: Valores máximos, mínimos y ordenación final ($group + $sort)
Encuentra cuál es el libro más largo (máximo de páginas) y el más corto (mínimo de páginas) según cada autor, ordenando el resultado final alfabéticamente por el nombre del autor.

```
const pipeline = [
    { $group: {
        _id: "$autor",
        maxPaginas: { $max: "$paginas" },
        minPaginas: { $min: "$paginas" }
    }},
    { $sort: { _id: 1 } }
];
const res = await col.aggregate(pipeline).toArray();
```