Hướng Dẫn Tạo Kết Nối CSDL MongoDB, Sinh Dữ Liệu Test và Đánh Giá Hiệu Quả Index

Tác giả: Đặng Kim Thi

Mục lục

1. Giới thiệu
2. Cài đặt môi trường
3. Kịch bản chi tiết
    - Kết nối tới MongoDB
    - Sinh dữ liệu mẫu
    - Đánh giá hiệu quả truy vấn
4. Mã nguồn
5. Kết quả và so sánh
6. Mở rộng

1. Giới thiệu

MongoDB là một cơ sở dữ liệu NoSQL phổ biến với tính năng mạnh mẽ trong việc lưu trữ dữ liệu phi cấu trúc. Một trong những tính năng quan trọng của MongoDB là khả năng tạo **index** để tối ưu hóa truy vấn dữ liệu.

Bài giảng này hướng dẫn bạn cách:

- Kết nối tới MongoDB bằng JavaScript.
- Sinh dữ liệu mẫu (1 triệu bản ghi).
- Đo lường hiệu quả của việc sử dụng index.
- So sánh kết quả truy vấn với và không có index.

2. Cài đặt môi trường

1. **Yêu cầu hệ thống**:
    - Đã cài đặt **MongoDB** và đang chạy.
    - **Node.js** và trình quản lý gói **npm**.
2. **Cài đặt thư viện MongoDB**: Mở terminal và chạy lệnh sau để cài đặt thư viện `mongodb`:

```
npm install mongodb
```

3. Kịch bản chi tiết

Kết nối tới MongoDB

- Sử dụng thư viện `mongodb` để tạo kết nối tới cơ sở dữ liệu MongoDB.
- URI mặc định là `mongodb://localhost:27017`.
- Đảm bảo kết nối thành công trước khi tiếp tục.

Sinh dữ liệu mẫu

- Tạo một bộ sưu tập (collection) tên `testCollection`.
- Sinh dữ liệu mẫu bao gồm 1 triệu bản ghi với cấu trúc như sau:

```
{
    "name": "User_0",
    "age": 25,
    "email": "user_0@example.com",
    "createdAt": "2025-01-01T00:00:00Z"
}
```

- Sử dụng phương thức `insertMany` để chèn dữ liệu hàng loạt.

Đánh giá hiệu quả truy vấn

1. Thực hiện truy vấn với điều kiện tìm kiếm `{ age: 25 }` và đo thời gian.
2. Tạo index trên trường `age` bằng phương thức `createIndex`.
3. Thực hiện lại truy vấn tương tự và đo thời gian.
4. So sánh kết quả giữa hai truy vấn để đánh giá hiệu quả của index.

---

4. Mã nguồn

File: `indexTest.js`

```
const { MongoClient } = require("mongodb");

async function main() {
    const uri = "mongodb://localhost:27017"; // Thay bằng URI của bạn nếu
khác
    const client = new MongoClient(uri);

    try {
        // Kết nối tới MongoDB
        await client.connect();
        console.log("Connected to MongoDB!");

        const db = client.db("testDB");
        const collection = db.collection("testCollection");

        // Xóa dữ liệu cũ nếu có
        await collection.deleteMany({});
```

```
        console.log("Collection cleared.");

        // Sinh dữ liệu mẫu (1 triệu bản ghi)
        console.log("Inserting test data...");
        const bulkData = [];
        for (let i = 0; i < 1000000; i++) {
            bulkData.push({
                name: `User_${i}`,
                age: Math.floor(Math.random() * 100),
                email: `user_${i}@example.com`,
                createdAt: new Date()
            });
        }
        await collection.insertMany(bulkData);
        console.log("Test data inserted.");

        // Thử truy vấn mà không có index
        console.time("Query without index");
        await collection.find({ age: 25 }).toArray();
        console.timeEnd("Query without index");

        // Tạo index
        console.log("Creating index on 'age' field...");
        await collection.createIndex({ age: 1 });
        console.log("Index created.");

        // Thử truy vấn với index
        console.time("Query with index");
        await collection.find({ age: 25 }).toArray();
        console.timeEnd("Query with index");

        // Kiểm tra danh sách index
        const indexes = await collection.indexes();
        console.log("Indexes:", indexes);

    } catch (err) {
        console.error(err);
    } finally {
        // Đóng kết nối
        await client.close();
        console.log("Disconnected from MongoDB!");
    }
}

main().catch(console.error);
```

---

5. Kết quả và so sánh

- **Query without index**: Thời gian truy vấn lớn hơn, thường từ vài giây.
- **Query with index**: Thời gian truy vấn giảm đáng kể (vài mili giây).

Ví dụ kết quả:

```
Query without index: 2874.573ms
Query with index: 12.485ms
```

Điều này minh chứng rằng index giúp cải thiện hiệu suất truy vấn rất nhiều, đặc biệt khi làm việc với tập dữ liệu lớn.

---

6. Mở rộng

- **Thử nghiệm các loại index khác**: Compound index, text index, 2dsphere index.
- **Kết hợp truy vấn phức tạp**: Kết hợp nhiều điều kiện để đánh giá hiệu quả index.
- **Tăng kích thước dữ liệu**: Thử nghiệm với dữ liệu lớn hơn (10 triệu bản ghi hoặc hơn).

Tác giả: Đặng Kim Thi