TÌM HIỂU VỀ MONGODB

1. **Tổng quan về MongoDB:**

MongoDB là một trong những hệ quản trị cơ sở dữ liệu NoSQL nổi bật, được thiết kế dưới dạng mã nguồn mở và hướng tài liệu. Khác với các cơ sở dữ liệu quan hệ truyền thống dùng bảng để lưu trữ, MongoDB sử dụng Document để lưu trữ và truy xuất dữ liệu. Dữ liệu trong MongoDB được định dạng bằng BSON, tương tự như JSON.

Được ra mắt vào năm 2009, MongoDB được lập trình bằng ngôn ngữ C++, giúp nó vận hành với tốc độ nhanh hơn so với nhiều hệ quản trị cơ sở dữ liệu khác. Chính điều này đã nâng cao giá trị của MongoDB trong mắt các nhà phát triển.

Hệ thống này hỗ trợ đa nền tảng và cho phép lưu trữ các dữ liệu có cấu trúc phức tạp. Thay vì sử dụng bảng truyền thống, MongoDB lưu trữ dữ liệu theo cách Collections dựa trên tài liệu JSON. Đây là mô hình dữ liệu Key-Value, mang lại khả năng truy xuất nhanh chóng và khả năng mở rộng linh hoạt mà không cần các ràng buộc khóa ngoại hay khóa chính.

* **NoSQL (Not Only SQL):** MongoDB không sử dụng cấu trúc bảng truyền thống như các cơ sở dư liệu quan hệ (SQL). Điều này cho phép MongoDB linh hoạt hơn trong việc xử lý dữ liệu phi cấu trúc hoặc bán cấu trúc.
* **Tài liệu JSON:** Dữ liệu trong MongoDB được lưu dưới dạng tài liệu JSON, giúp ta dễ dàng làm việc với dữ liệu có cấu trúc thay đổi theo thời gian.
* **Schema-less:** MongoDB là một cơ sở dữ liệu dựa trên Document, các tài liệu trong một Collection có thể có cấu trúc khác nhau.

1. **Các thành phần chính của MongoDB:**

* **Mongod:** Là tiến trình lõi của MongoDB, đảm nhận nhiệm vụ lưu trữ và quản lý dữ liệu.
* **Mongosh:** Với giao diện Shell giúp người dùng có thể tương tác thực hiện lệnh với cơ sở dữ liệu.
* **Collection:** Một tập hợp các Document MongoDB. Tương tự như bảng trong cơ sở dữ liệu quan hệ, Collection có tính linh hoạt cao, không ràng buộc bởi cấu trúc cố định, cho phép chứa các tài liệu với những cấu trúc khác nhau.
* **Document:**
* Đơn vị lưu trữ dữ liệu cơ bản trong cơ sở dữ liệu MongoDB.
* Document trong MongoDB không cần phải có cùng trường hoặc cấu trúc với các Document khác trong cùng một Collection.
* Đồng thời, các trường chung trong Document của một Collection có thể chứa các loại dữ liệu khác nhau.
* **Database (Cơ sở dữ liệu):**
* Database là một thùng chứa vật lý chứa tập hợp các Collection. Một Database có thể chứa không Collection hoặc nhiều Collection.
* Một phiên bản máy chủ MongoDB có thể lưu trữ nhiều Database và không có giới hạn về số lượng Database có thể được lưu trữ trên một phiên bản.
* MongoDB sẽ tự động tạo cơ sở dữ liệu mới khi lưu trữ Document lần đầu tiên nếu như cơ sở dữ liệu không tồn tại.
* **MongoDB Compass:** Là công cụ giao diện đồ họa (GUI) cho phép người dùng tương tác trực quan với cơ sở dữ liệu MongoDB thay vì sử dụng giao diện dòng truyền thống. Với khả năng hiển thị trực quan các cơ sở dữ liệu, Collection, Document, MongoDB Compass hỗ trợ người dùng trong việc truy vấn, thực hiện các hàm tổng hợp và phân tích dữ liệu hiệu quả.

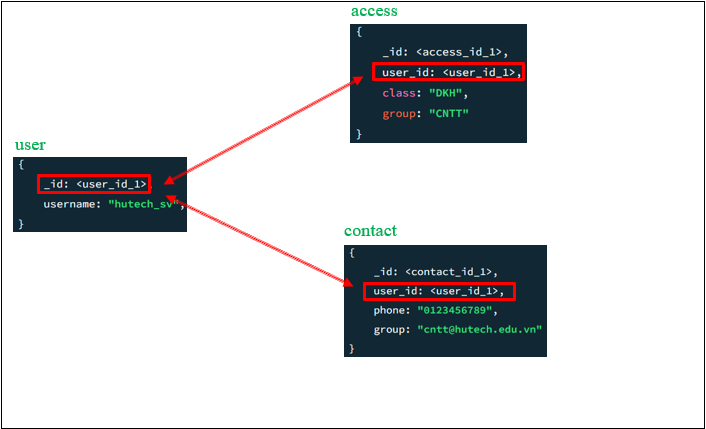
1. **Các cấu trúc của mô hình dữ liệu:**

Một điểm mấu chốt trong việc tạo lập mô hình dữ liệu cho những ứng dụng sử dụng MongoDB liên quan đến cách tổ chức các Collection và cách thức mà ứng dụng thể hiện mối liên kết giữa các dữ liệu.

* **Nhúng dữ liệu:** MongoDB cho phép tích hợp mọi dữ liệu liên quan vào một tài liệu duy nhất. Dữ liệu này phản ánh mối quan hệ giữa các thành phần thông qua việc tổ chức chúng trong một cấu trúc tài liệu duy nhất. Nền tảng cũng hỗ trợ việc tích hợp các cấu trúc dữ liệu vào một trường hoặc mảng trong tài liệu. Các mô hình dữ liệu không theo tiêu chuẩn này tạo điều kiện cho ứng dụng thực hiện truy xuất và xử lý dữ liệu liên quan trong một thao tác cơ sở dữ liệu duy nhất.

*Ví dụ về Nhúng dữ liệu*

* **Tham chiếu:** Được gọi là kiến trúc chuẩn hóa của mô hình dữ liệu, các tham chiếu lưu trữ các mối quan hệ giữa dữ liệu thông qua việc thêm liên kết hoặc tham chiếu giữa các tài liệu. Các ứng dụng có thể sử dụng những tham chiếu này để truy cập các dữ liệu liên quan. Đây là một dạng mô hình dữ liệu đã được chuẩn hóa. Trong ví dụ dưới đây, trường “user\_id” trong bộ dữ liệu “contact” và “access” được sử dụng để tham chiếu đến người dùng tương ứng

1. **Các mẫu thiết kế của mô hình dữ liệu:**

*Ví dụ về Tham chiếu*

Ví dụ tham chiếu trong mô hình dữ liệu của MongoDB

* 1. **Mô hình quan hệ giữa các Document:**

1. **Quan hệ một – một**

Ví dụ nếu xây dựng sơ đồ mối liên hệ một – một giữa sinh viên và địa chỉ của họ. Trong mô hình dữ liệu chuẩn hóa, tài liệu “DiaChi” sẽ bao gồm tham chiếu đến tài liệu “SinhVien”. Nếu ứng dụng thường xuyên cần truy cập dữ liệu địa chỉ song song với dữ liệu sinh viên, thì việc sử dụng tham chiếu sẽ yêu cầu thực hiện nhiều truy vấn hơn để xử lý các tham chiếu đó.

*Hình 4.1: Ví dụ về Quan hệ một – một*

1. **Quan hệ một – nhiều sử dụng nhúng dữ liệu**

Mô hình dữ liệu quan hệ mang tính một – nhiều áp dụng phương pháp nhúng dữ liệu khi cần truy xuất nhiều thực thể liên quan đến một thực thể trong tập dữ liệu khác. Ví dụ như, một sinh viên có thể có nhiều địa chỉ liên lạc. Nếu việc truy cập thông tin địa chỉ cùng lúc với dữ liệu sinh viên diễn ra thường xuyên, thì phương án hiệu quả hơn là nhúng các thực thể địa chỉ vào trong dữ liệu sinh viên.



*Hình 4.2: Ví dụ về Quan hệ một – nhiều*

1. **Quan hệ một – nhiều sử dụng tham chiếu**

Xem xét ví dụ về cấu trúc dữ liệu mô tả mối quan hệ giữa sinh viên và các khóa học đã đăng ký. Một môn học có thể thu hút nhiều sinh viên, vì vậy việc tích hợp thông tin môn học vào dữ liệu của từng sinh viên có thể dẫn đến tình trạng lặp lại thông tin môn học, gây lãng phí tài nguyên hệ thống. Để ngăn chặn việc lưu trữ trùng lặp thông tin môn học, có thể tạo ra một bộ sưu tập riêng biệt để lưu trữ dữ liệu về môn học. Cần ghi nhớ rằng trong trường hợp này, một môn học sẽ có rất nhiều sinh viên đăng ký, do đó, việc tham chiếu từ dữ liệu môn học sang dữ liệu sinh viên có thể khiến lượng thông tin cần lưu trữ trở nên khổng lồ.



Cần xem xét việc tham khảo dữ liệu môn học từ thông tin sinh viên theo các cách như sau:

* 1. Mô hình cây:



*Hình 4.3: Ví dụ hệ thống danh mục sách được mô tả theo cấu trúc cây*

1. Các phương thức CRUD trong MongoDB:

* Thêm dữ liệu: Sử dụng cú pháp ***db.collection.insertOne()*** để thêm một Document vào Collection.



* Khi cần thêm nhiều Document cùng một lúc, sử dụng cú pháp ***db.collection.insertMany()*** và truyền vào một mảng gồm các Document.



* **Truy vấn dữ liệu:**

Để truy xuất tất cả tài liệu trong một Collection, bạn có thể áp dụng phương pháp ***db.collection.find()*** và cung cấp một đối tượng JSON rỗng {} làm tham số.

*Hình 5.1: Sử dụng phương thức db.collection.find() để truy xuất tất cả dữ liệu trong tập hợp students*

* **Cập nhật dữ liệu:**

Cập nhật dữ liệu trên MongoDB được thực hiện thông qua các câu lệnh:

* ***db.collection.updateOne()***
* ***db.collection updateMany()***
* **Xóa dữ liệu:**

Xóa tất cả Document trong Collection, ta sử dụng:

* ***db.collection.deleteMany()*** và truyền vào tham số là một đối tượng JSON rỗng.

*Ví dụ: db.students.deleteMany({})*

1. **Các tính năng nổi bật của MongoDB:**

* **Tìm kiếm và lập chỉ mục mạnh:** MongoDB cung cấp khả năng tìm kiếm dữ liệu nhanh chóng nhờ hệ thống lập chỉ mục linh hoạt.
* **Aggregation framework:** Cho phép xử lý dữ liệu phức tạp và tạo các báo cáo chi tiết từ dữ liệu.
* **Replication:** Đảm bảo tính sẵn sàng cao bằng cách tạo bản sao dữ liệu trên nhiều máy chủ. Ngoài ra, còn có khả năng tự động sao lưu dữ liệu, đảm bảo độ tin cậy và khả năng phục hồi của cơ sở dữ liệu.
* **BSON:** BSON là một định dạng dữ liệu nhị phân được sử dụng để lưu trữ tài liệu và thực hiện các cuộc gọi từ xa trong MongoDB. BSON bao gồm nhiều kiểu dữ liệu khác nhau, cho phép sử dụng số hoặc chuỗi định danh để xác định kiểu dữ liệu của một đối tượng.

1. **So sánh MongoDB với các cơ sở dữ liệu khác:**

* **So với SQL (MySQL, PostgreSQL):** MongoDB có ưu điểm về tính linh hoạt và mở rộng, nhưng không mạnh trong việc xử lý các mối quan hệ phức tạp giữa các bảng.
* **So với các cơ sở dữ liệu NoSQL khác (Cassandra, Couchbase):** MongoDB nổi bật nhờ mô hình dữ liệu tài liệu dễ tiếp cận và cộng đồng hỗ trợ rộng lớn.



1. **Công cụ hỗ trợ MongoDB:**

* **MongoDB Compass:** Một công cụ GUI (giao diện đồ họa) cho phép người dùng trực hóa, quản lý và thao tác với dữ liệu MongoDB.
* **Mongoose:** Một ORM (Object-Relational Mapping) cho Node.js, giúp dễ dàng tương tác với MongoDB từ các ứng dụng Node.js

1. **Kết luận**

* ***Ưu điểm:***
* **Không Schema:** Giống như các cơ sở dữ liệu NoSQL khác, MongoDB không yêu cầu các Schema được xác định trước.
* **Lưu trữ bất kỳ loại dữ liệu nào:** Điều này cho phép người dùng linh hoạt tạo số lượng trường trong Document theo nhu cầu, và giúp việc mở rộng cơ sở dữ liệu MongoDB trở nên dễ dàng hơn so với cơ sở dữ liệu quan hệ truyền thống.
* **Hướng Document:** Một trong những ưu điểm của việc sử dụng Document là các đối tượng này ánh xạ tới các kiểu dữ liệu gốc trong một số ngôn ngữ lập trình. Việc có các Document được nhúng cũng làm giảm nhu cầu kết nối cơ sở dữ liệu, điều này có thể làm giảm chi phí.
* **Khả năng mở rộng:** MongoDB có khả năng mở rộng dễ dàng bằng cách phân tán dữ liệu trên nhiều máy chủ qua việc phân chia dữ liệu ( sharding ). Ngoài ra, MongoDB cũng hỗ trợ tạo vùng dữ liệu dựa trên Shard Key.
* **Hiệu suất cao:** MongoDB đặc biệt tốt trong việc xử lý lượng lớn dữ liệu phi cấu trúc hoặc bán cấu trúc.
* Sử dụng các tính năng tìm hiểu được trên một ứng dụng trên môi trường localhost.
* ***Nhược điểm:***
* **Tốn tài nguyên:** MongoDB sử dụng nhiều tài nguyên hơn so với cơ sở dữ liệu quan hệ vì cần lưu thêm metadata trong tài liệu.
* **Tính nhất quán của dữ liệu:** MongoDB không cung cấp tính toàn vẹn tham chiếu đầy đủ thông qua việc sử dụng các ràng buộc khóa ngoại (foreign-key), điều này có thể ảnh hưởng đến tính nhất quán của dữ liệu.
* **Tính liên tục:** Với chiến lược chuyển đổi dự phòng tự động, người dùng chỉ có thể thiết lập một node master trong cụm MongoDB. Nếu node master bị lỗi, một node khác sẽ tự động chuyển đổi thành master mới. Quá trình chuyển đổi này đảm bảo tính liên tục, nhưng không diễn ra tức thời mà có thể mất tới một phút.
* ***Hướng phát triển:***
* Tối ưu hóa việc sử dụng tài nguyên bằng cách nghiên cứu và phát triển thêm các thuật toán nén hiệu quả hơn để tiết kiệm dung lượng lưu trữ và tài nguyên. Đồng thời, cải tiến cách lưu trữ metadata bằng cách giảm bớt thông tin thừa hoặc tìm kiếm định dạng nhẹ hơn mà vẫn giữ được tính linh hoạt của cơ sở dữ liệu.
* Tăng cường tính nhất quán của dữ liệu bằng việc tích hợp hỗ trợ khóa ngoại hoặc thiết lập cơ chế kiểm tra mối quan hệ giữa các collection để bảo vệ dữ liệu tốt hơn.
* Phát triển chiến lược mở rộng tự động tài nguyên theo thời gian thực khi cần thiết, chẳng hạn như tự động bổ sung node để cải thiện hiệu suất và giảm tải cho hệ thống khi nhu cầu gia tăng.

Link tham khảo

<https://vi.wikipedia.org/wiki/MongoDB>

<https://itviec.com/blog/mongodb-la-gi/>

<https://viblo.asia/p/mongodb-la-gi-cac-tinh-nang-noi-bat-cua-mongodb-PAoJex2N41j>

<https://aws.amazon.com/vi/compare/the-difference-between-cassandra-and-mongodb/>

<https://viblo.asia/p/mo-ta-ve-data-models-trong-mongodb-7ymwGXA0v4p1>

<https://www.mongodb.com/docs/manual/>