

LỜI CẢM ƠN

Trước tiên xin gửi lời cảm ơn chân thành đến Ban Giám hiệu Trường Đại học Trà Vinh, đặc biệt là Khoa Công nghệ thông tin – trực thuộc Trường Kỹ thuật & Công nghệ – vì đã tạo điều kiện và hỗ trợ em trong suốt quá trình học tập và rèn luyện.

Đặc biệt, em xin gửi lời cảm ơn sâu sắc đến cô Phan Thị Phương Nam – Giảng viên hướng dẫn học phần môn Thực tập đồ án cơ sở ngành. Khi bắt đầu tiếp cận với đề tài, em đã gặp khá nhiều khó khăn trong việc thay đổi tư duy từ cơ sở dữ liệu quan hệ sang cơ sở dữ liệu phi quan hệ, cũng như các bài toán phát sinh trong quá trình thực hiện. Chính nhờ sự góp ý và hỗ trợ nhiệt tình từ cô, em đã hiểu được cốt lõi vấn đề và hoàn thành đề tài “Thiết kế và cài đặt cơ sở dữ liệu NoSQL cho hệ thống diễn đàn Hỏi-Đáp chuyên ngành với tính năng đánh giá và vinh danh”. Những kiến thức này không chỉ giúp em hoàn thành đề tài này mà còn là hành trang để em có thể ứng dụng chúng sau này.

Mặc dù đã rất cố gắng trong quá trình thực hiện, nhưng với kiến thức và kinh nghiệm còn hạn chế, đề tài chắc chắn vẫn không tránh khỏi những thiếu sót. Em rất mong nhận được sự góp ý, chia sẻ từ quý thầy cô để đề tài được hoàn thiện hơn, cũng như giúp nhóm có thêm kinh nghiệm cho những dự án học tập sau này.

Em xin chân thành cảm ơn!

Vĩnh Long, ngày tháng năm
Sinh viên thực hiện

Lâm Tấn Hưng

MỤC LỤC

DANH MỤC HÌNH ẢNH.....	7
DANH MỤC BẢNG BIỂU.....	8
DANH MỤC TỪ VIẾT TẮT VÀ THUẬT NGỮ.....	9
TÓM TẮT ĐỒ ÁN CƠ SỞ NGÀNH.....	11
MỞ ĐẦU	12
CHƯƠNG 1: TỔNG QUAN	14
CHƯƠNG 2: NGHIÊN CỨU LÝ THUYẾT	15
2.1 Tổng quan về cơ sở dữ liệu phi quan hệ.....	15
2.1.1 Khái niệm và đặc điểm.....	15
2.1.2 Các dạng cơ sở dữ liệu phi quan hệ	16
2.1.3 Một số thuật ngữ và các khái niệm liên quan.....	17
2.2 Tổng quan về MongoDB	18
2.2.1 Mục đích của MongoDB	18
2.2.2 Ưu điểm của MongoDB so với các hệ quản trị CSDL khác	18
2.2.3 Một số khái niệm cơ bản trong MongoDB	19
2.2.4 Các kiểu dữ liệu trong MongoDB	20
2.2.5 Các thao tác CRUD trong MongoDB	20
2.2.6 Các toán tử cơ bản trong MongoDB	22
2.3 Xử lý dữ liệu nâng cao.....	23
2.3.1 Tổng quan về Aggregation Framework	23
2.3.2 Tổng quan về toán tử cập nhật nguyên tử	25
2.4 Lý thuyết về thiết kế lược đồ	27
2.4.1 Kỹ thuật Nhúng (Embedding)	27
2.4.2 Kỹ thuật Tham chiếu (Referencing).....	27

CHƯƠNG 3: HIỆN THỰC HÓA NGHIÊN CỨU	28
3.1 Phân tích yêu cầu và đặc tả chức năng	28
3.1.1 Mô tả.....	28
3.1.2 Sơ đồ Use Case tổng quát.....	30
3.2 Thiết kế lược đồ cơ sở dữ liệu	32
3.2.1 Mô hình dữ liệu	32
3.2.2 Chiến lược thiết kế lược đồ	35
3.3 Môi trường và công cụ thực hiện.....	35
3.4 Chiến lược sinh dữ liệu mẫu.....	36
CHƯƠNG 4: KẾT QUẢ NGHIÊN CỨU	38
4.1 Triển khai dữ liệu mẫu.....	38
4.1.1 Quy trình khởi tạo	38
4.1.2 Dữ liệu cho users	39
4.1.3 Dữ liệu cho questions	44
4.1.4 Dữ liệu cho answers	55
4.1.5 Các trường hợp được tạo trong dữ liệu mẫu	71
4.2 Kết quả và thực hiện các chức năng Quản lý (CRUD).....	72
4.2.1 Đăng câu hỏi mới (UC03)	72
4.2.2 Đăng câu trả lời (UC04).....	73
4.2.3 Xem chi tiết bài đăng (UC01)	75
4.2.4 Sửa bài đăng (UC08).....	76
4.2.5 Quản lý bài đăng - Ẩn bài vi phạm (UC10)	78
4.3 Kết quả và thực hiện nghiệp vụ Đánh giá.....	78
4.3.1 Cơ chế bình chọn đồng thời	79
4.3.2 Đánh giá câu hỏi (UC05)	79

4.3.3	Đánh giá câu trả lời (UC06)	80
4.3.4	Chấp nhận câu trả lời (UC07)	81
4.4	Kết quả và thực hiện nghiệp vụ Vinh danh	82
4.4.1	Top 5 thành viên uy tín nhất.....	82
4.4.2	Vinh danh thành viên giải quyết vấn đề xuất sắc nhất.....	84
4.4.3	Vinh danh thành viên hoạt động chăm chỉ nhất (Nhiều câu trả lời)	85
4.4.4	Vinh danh thành viên đặt vấn đề nhiều nhất (Nhiều câu hỏi).....	86
4.4.5	Vinh danh theo huy chương	87
4.4.6	Câu hỏi được thảo luận sôi nổi nhất.....	87
4.4.7	Câu trả lời chất lượng nhất.....	89
4.4.8	Chuyên gia lĩnh vực MongoDB	90
4.4.9	Phân loại danh hiệu người dùng.....	91
CHƯƠNG 5: KẾT LUẬN VÀ HƯỚNG PHÁT TRIỂN		93
5.1	Về mặt lý thuyết.....	93
5.2	Về mặt thực tiễn và kỹ thuật:.....	93
5.3	Đánh giá ưu điểm và nhược điểm.....	93
5.4	Hướng phát triển	94
DANH MỤC TÀI LIỆU THAM KHẢO		96

DANH MỤC HÌNH ẢNH

<i>Hình 3.1: Sơ đồ Use Case tổng quát cho hệ thống</i>	<i>30</i>
<i>Hình 4.1: Nạp dữ liệu trên MongoDB Shell</i>	<i>38</i>
<i>Hình 4.2: Thông báo kết quả thực thi thành công.....</i>	<i>39</i>

DANH MỤC BẢNG BIỂU

<i>Bảng 2.1: Một số thuật ngữ tương đương trong cơ sở dữ liệu quan hệ và cơ sở dữ liệu phi quan hệ [1].</i>	17
<i>Bảng 2.2: Các kiểu dữ liệu trong MongoDB.</i>	20
<i>Bảng 2.3: Danh sách các toán tử truy vấn.</i>	22
<i>Bảng 2.4: Danh sách các toán tử cập nhật.</i>	23
<i>Bảng 2.5: Bảng tổng hợp các toán tử (operators) đại diện cho các giai đoạn phổ biến nhất trong Aggregation Pipeline.</i>	24
<i>Bảng 2.6: Danh sách các hàm tích lũy trong giai đoạn \$group.</i>	25
<i>Bảng 2.7: Các toán tử cập nhật.</i>	26
<i>Bảng 3.1: Cơ chế thưởng/phạt dựa trên các hành động.</i>	28
<i>Bảng 3.2: Các mốc danh hiệu người dùng.</i>	29
<i>Bảng 3.3: Tiêu chí đạt huy chương cho câu trả lời.</i>	29
<i>Bảng 3.4: Tổng hợp các đặc tả use case của hệ thống.</i>	31

DANH MỤC TỪ VIẾT TẮT VÀ THUẬT NGỮ

STT	Từ viết tắt	Từ đầy đủ (Tiếng Anh)	Ý nghĩa (Tiếng Việt)
1	ACID	Atomicity, Consistency, Isolation, Durability	4 tính chất của giao dịch: Nguyên tử, Nhất quán, Cô lập, Bền vững.
2	AI	Artificial Intelligence	Trí tuệ nhân tạo.
3	API	Application Programming Interface	Giao diện lập trình ứng dụng.
4	BASE	Basically Available, Soft state, Eventual consistency	Mô hình nhất quán trong NoSQL: Cơ bản sẵn sàng, Trạng thái mềm, Nhất quán cuối cùng.
5	BSON	Binary JSON	Định dạng mã hóa dữ liệu dạng nhị phân của JSON (dùng trong MongoDB).
6	CAP	Consistency, Availability, Partition Tolerance	Định lý CAP: Tính nhất quán, Tính sẵn sàng, Khả năng chịu lỗi phân vùng.
7	CNTT	—	Công nghệ thông tin.
8	CPU	Central Processing Unit	Bộ xử lý trung tâm.
9	CQL	Cassandra Query Language	Ngôn ngữ truy vấn của cơ sở dữ liệu Cassandra.
10	CRUD	Create, Read, Update, Delete	Bốn thao tác cơ bản của cơ sở dữ liệu: Tạo, Đọc, Cập nhật, Xóa.
11	CSDL	—	Cơ sở dữ liệu.
12	GB	Gigabyte	Đơn vị đo dung lượng lưu trữ thông tin.
13	GridFS	Grid File System	Đặc tả kỹ thuật để lưu trữ và truy xuất các tệp lớn trong MongoDB.
14	GUI	Graphical User Interface	Giao diện người dùng đồ họa.
15	ID	Identifier	Mã định danh.
16	JSON	JavaScript Object Notation	Định dạng trao đổi dữ liệu văn bản nhẹ, độc lập ngôn ngữ.

Thiết kế và cài đặt cơ sở dữ liệu NoSQL cho hệ thống diễn đàn Hỏi-Đáp chuyên ngành với tính năng đánh giá và vinh danh.

STT	Từ viết tắt	Từ đầy đủ (Tiếng Anh)	Ý nghĩa (Tiếng Việt)
17	LLM	Large Language Model	Mô hình ngôn ngữ lớn (ví dụ: ChatGPT, Gemini).
18	MongoDB	Humongous Database	Hệ quản trị cơ sở dữ liệu mã nguồn mở theo mô hình hướng tài liệu (NoSQL).
19	MySQL	My Structured Query Language	Một hệ quản trị cơ sở dữ liệu quan hệ mã nguồn mở phổ biến.
20	NoSQL	Not Only SQL	Thuật ngữ chung cho các hệ quản trị CSDL không sử dụng mô hình quan hệ truyền thống.
21	ObjectId	—	Kiểu dữ liệu dùng làm khóa chính (_id) mặc định, tự động sinh ra trong MongoDB.
22	RAM	Random Access Memory	Bộ nhớ truy cập ngẫu nhiên.
23	RDBMS	Relational Database Management System	Hệ quản trị cơ sở dữ liệu quan hệ.
24	REST	Representational State Transfer	Một kiểu kiến trúc phần mềm cho các hệ thống siêu văn bản phân tán.
25	SQL	Structured Query Language	Ngôn ngữ truy vấn mang tính cấu trúc.
26	STT	—	Số thứ tự.
27	UC	Use Case	Trường hợp sử dụng (Mã chức năng trong đặc tả hệ thống).
28	UTF-8	Unicode Transformation Format – 8-bit	Bộ mã hóa ký tự Unicode độ dài biến đổi (mặc định trong JSON/BSON).
29	XML	eXtensible Markup Language	Ngôn ngữ đánh dấu mở rộng.

TÓM TẮT ĐỒ ÁN CƠ SỞ NGÀNH

Vấn đề nghiên cứu: Trong bối cảnh dữ liệu lớn và nhu cầu trao đổi kiến thức chuyên môn ngày càng tăng. Các diễn đàn Hỏi-Đáp nói riêng và các nền tảng chuyên ngành nói chung phải xử lý lượng dữ liệu khổng lồ. Các cơ sở dữ liệu quan hệ truyền thống thường gặp hạn chế về khả năng mở rộng cấu trúc, xử lý dữ liệu không cấu trúc và hiệu suất khi lượng truy cập cao.

Các hướng tiếp cận: Phân tích yêu cầu, nghiệp vụ cho hệ thống Hỏi-Đáp chuyên ngành, phân tích đề tài theo mô hình cơ sở dữ liệu phi quan hệ dưới dạng Document. Áp dụng tư duy phi chuẩn hóa và sự linh hoạt của lược đồ để tối ưu hiệu suất truy vấn, phù hợp với đặc thù của một hệ thống diễn đàn hiện đại.

Cách giải quyết vấn đề: Xây dựng được lược đồ dữ liệu theo mẫu thiết kế phù hợp, thực hiện được các chức năng của một diễn đàn. Sử dụng MongoDB để giải quyết các vấn đề nghiệp vụ của đề tài.

Một số kết quả đạt được: Triển khai được cơ sở dữ liệu phi quan hệ đáp ứng đầy đủ các thao tác cơ bản như CRUD và các nghiệp vụ phức tạp về đánh giá và vinh danh. Đánh giá kết quả để cho thấy ưu điểm của cơ sở dữ liệu phi quan hệ trong việc xây dựng, triển khai và tính linh hoạt của nó, tạo nền tảng cho việc phát triển và mở rộng trong tương lai.

MỞ ĐẦU

Trong bối cảnh dữ liệu lớn và nhu cầu trao đổi kiến thức chuyên môn ngày càng tăng, các diễn đàn Hỏi-Đáp nói riêng và các nền tảng chuyên ngành nói chung phải xử lý lượng dữ liệu khổng lồ. Thực tế cho thấy các cơ sở dữ liệu quan hệ truyền thống thường gặp hạn chế về khả năng mở rộng cấu trúc, xử lý dữ liệu không cấu trúc và hiệu suất khi lượng truy cập cao. Điều đó cho thấy cần phải có một giải pháp lưu trữ linh hoạt, đáp ứng các nhu cầu trên.

Để giải quyết vấn đề này, đề án được lựa chọn theo hướng tiếp cận với mô hình cơ sở dữ liệu phi quan hệ, đặc biệt là sử dụng hệ quản trị MongoDB. Khác với tư duy thiết kế truyền thống, thay vì ép buộc dữ liệu vào các bảng với dòng và cột cố định như SQL, đề án thực hiện dưới dạng các Document có cấu trúc linh hoạt, nhờ đó lưu trữ một đối tượng trọn vẹn kèm các thông tin khác như hình ảnh, bình luận,..., giúp phản ánh bản chất đa dạng của dữ liệu thực tế. Ngoài ra, kỹ thuật “phi chuẩn hóa” được áp dụng, chấp nhận sự trùng lặp dữ liệu có chủ đích để tối ưu tốc độ truy xuất, đảm bảo hệ thống phản hồi nhanh chóng ngay cả khi dữ liệu ngày càng lớn.

Đề án tập trung thiết kế và cài đặt cấu trúc dữ liệu, đặc biệt chú trọng vào hai nghiệp vụ chính là “đánh giá” và “vinh danh”. Đối với chức năng đánh giá, áp dụng các toán tử cập nhật nguyên tử để xử lý các tác vụ khi người dùng tương tác. Đối với chức năng vinh danh, sử dụng sức mạnh của Aggregation Framework để thực hiện các thống kê nâng cao, phức tạp như phân loại huy chương và xếp hạng thành viên theo tiêu chí mà không làm ảnh hưởng đến toàn hệ thống.

Để giải quyết các bài toán đặt ra, đề tài xác định đối tượng nghiên cứu trọng tâm là cơ sở dữ liệu phi quan hệ, cụ thể là mô hình hướng tài liệu và hệ quản trị MongoDB cùng các công cụ hỗ trợ đi kèm. Song song với đó là việc tập trung nghiên cứu phương pháp phân tích và thiết kế hệ thống chuyên biệt cho hệ thống, chuyển dịch từ tư duy chuẩn hóa của cơ sở dữ liệu quan hệ sang tư duy phi chuẩn hóa nhằm tối ưu hiệu suất. Các công cụ như toán tử cập nhật nguyên tử và Aggregation Framework cũng được nghiên cứu để giải quyết bài toán nghiệp vụ về đánh giá và vinh danh trên tập dữ liệu mẫu của diễn đàn Hỏi-Đáp chuyên ngành.

Về phạm vi thực hiện, đề án được giới hạn cụ thể để đảm bảo tính khả thi và chiều sâu nghiên cứu. Về mặt nội dung, đề tài chỉ tập trung vào các công đoạn gồm: thiết kế

Thiết kế và cài đặt cơ sở dữ liệu NoSQL cho hệ thống diễn đàn Hỏi-Đáp chuyên ngành với tính năng đánh giá và vinh danh.

lược đồ, cài đặt cơ sở dữ liệu, nhập liệu và xây dựng các câu truy vấn xử lý dữ liệu theo yêu cầu, không bao gồm việc phát triển giao diện người dùng hoàn chỉnh. Về mặt không gian và môi trường, toàn bộ hệ thống được cài đặt, triển khai và thử nghiệm các kịch bản truy vấn trên môi trường cục bộ của máy tính cá nhân.

CHƯƠNG 1: TỔNG QUAN

Xuất phát từ yêu cầu và định hướng nghiên cứu về các hệ quản trị cơ sở dữ liệu hiện đại, đề tài tập trung giải quyết các nghiệp vụ cốt lõi của một diễn đàn chuyên ngành. Vấn đề mà đề tài giải quyết không chỉ dừng lại ở việc lưu trữ dữ liệu lớn, mà còn là bài toán về xử lý các tương tác và thiết kế lược đồ để tối ưu cho việc truy vấn. Đối với hệ thống này, thách thức lớn nhất nằm ở việc xử lý đồng thời hàng ngàn thao tác như đánh giá và tính toán “vinh danh” ngay lập tức. Trong cơ sở dữ liệu quan hệ truyền thống, việc thực hiện các tác vụ này đòi hỏi các cơ chế ràng buộc như khóa chính, khóa ngoại, kèm theo các phép kết (JOIN) giữa các bảng với nhau để tổng hợp dữ liệu, dẫn đến chưa tối ưu tốt cho việc truy vấn.

Để giải quyết các vấn đề trên, đề tài sử dụng các giải pháp dựa trên cơ sở dữ liệu hướng tài liệu, cụ thể là hệ quản trị cơ sở dữ liệu MongoDB để thực hiện việc thử nghiệm. Nó cho phép lưu trữ dữ liệu dưới dạng linh hoạt để tối ưu hóa thao tác truy vấn dữ liệu mà không bị ràng buộc bởi các ràng buộc của các hệ quản trị cơ sở dữ liệu quan hệ truyền thống.

Hai nghiệp vụ chính của đề tài là “Đánh giá” và “Vinh danh” được thực hiện thông qua việc ứng dụng các toán tử cập nhật nguyên tử (\$inc, \$set,...) và Aggregation Framework. Chúng giải quyết triệt để vấn đề xung đột dữ liệu khi có nhiều người dùng cùng đánh giá tại một thời điểm, đảm bảo tính nhất quán cao, các thống kê phức tạp để vinh danh và xếp hạng người dùng theo thời gian thực mà không làm giảm hiệu năng hệ thống như các phép JOIN trong SQL.

Việc thực hiện đề tài này có ý nghĩa quan trọng trong việc xây dựng nền tảng hạ tầng dữ liệu cho các ứng dụng web hiện đại. Nó không chỉ chứng minh tính khả thi của việc thay thế SQL trong các nghiệp vụ đặc thù mà còn góp phần thúc đẩy môi trường chia sẻ tri thức chuyên ngành thông qua các cơ chế ghi nhận thành tích nhanh chóng. Đây là nền tảng để tiếp cận các bài toán dữ liệu lớn phức tạp hơn trong tương lai.

CHƯƠNG 2: NGHIÊN CỨU LÝ THUYẾT

2.1 Tổng quan về cơ sở dữ liệu phi quan hệ

Các CSDL quan hệ được phát triển vào thế kỷ 20 với mục đích là quản lý dữ liệu vận hành một cách chặt chẽ và có tổ chức. Mặc dù mô hình này mang lại sự ổn định, nhưng nó lại hạn chế về tính linh hoạt. Khi đối mặt với kỷ nguyên Dữ liệu lớn, công nghệ này đã phát sinh một số vấn đề. Các hệ thống sử dụng CSDL quan hệ cung cấp logic xử lý mạnh mẽ như đảm bảo tính nhất quán, toàn vẹn tham chiếu, phục hồi và đồng bộ hóa đa người dùng. Tuy nhiên, việc kiểm soát chặt chẽ dữ liệu này đòi hỏi nhiều công sức và năng lực xử lý, khiến chúng nhanh chóng đạt giới hạn khi xử lý lượng dữ liệu lớn. Tính hướng giao dịch và nhất quán của CSDL quan hệ trở thành trở ngại khi yêu cầu hiệu suất cao trong xử lý dữ liệu lớn. Các hệ thống này vốn thường dựa vào việc mở rộng chiều dọc (nâng cấp phần cứng như CPU, RAM của máy chủ đơn lẻ) và gặp khó khăn khi phân tán dữ liệu. Do đó, sự ra đời của NoSQL chính là giải pháp để giải quyết những hạn chế nêu trên.

2.1.1 Khái niệm và đặc điểm

Cơ sở dữ liệu phi quan hệ (NoSQL) là thuật ngữ được sử dụng cho các phương pháp quản lý dữ liệu phi quan hệ. Mặc dù NoSQL ban đầu có thể được hiểu là "No SQL" (Không dùng SQL), nhưng ngày nay nó thường được diễn giải là "Not only SQL" (Không chỉ SQL), nhằm thể hiện rằng các công nghệ khác ngoài công nghệ dữ liệu quan hệ được sử dụng trong các ứng dụng web phân tán khối lượng lớn [1].

Thuật ngữ NoSQL hiện được sử dụng cho bất kỳ phương pháp quản lý dữ liệu phi quan hệ nào đáp ứng hai tiêu chí:

- Thứ nhất, dữ liệu không được lưu trữ trong các bảng;
- Thứ hai, ngôn ngữ cơ sở dữ liệu không phải là SQL.

Mục tiêu chính là việc mở rộng theo chiều ngang (thêm nhiều máy chủ để mở rộng dung lượng và cải thiện hiệu suất) và tính linh hoạt của lược đồ (không cần phải khai báo trước cấu trúc dữ liệu trước khi lưu thông tin vào cơ sở dữ liệu). NoSQL được phát triển để giải quyết các vấn đề của Big Data như: khối lượng (lượng dữ liệu lớn), sự đa dạng (đa định dạng) và tốc độ (xử lý thời gian thực và tốc độ cao) [2].

Các cơ sở dữ liệu NoSQL có các đặc điểm sau [1]:

- Mô hình cơ sở dữ liệu không quan hệ;
- Tập trung vào khả năng mở rộng phân tán và theo chiều ngang;
- Có hạn chế về lược đồ yếu hoặc không có;
- Việc sao chép dữ liệu dễ dàng;
- Cung cấp truy cập dễ dàng thông qua API;
- Mô hình nhất quán không phải là ACID (ACID là viết tắt của tính nguyên tử (A), tính nhất quán (C), tính cô lập (I) và tính bền vững (D)).

2.1.2 Các dạng cơ sở dữ liệu phi quan hệ

Dựa trên các mô hình dữ liệu, CSDL phi quan hệ thường được phân chia thành bốn loại chính như sau:

2.1.2.1 Key-Value stores

Dữ liệu được tổ chức dưới dạng các cặp khóa (Key) và giá trị (Value). Đối với mỗi khóa, có chính xác một đối tượng dữ liệu mô tả có liên quan, giá trị cho khóa đó. Chỉ định một khóa cho phép truy vấn giá trị liên quan trong cơ sở dữ liệu, dựa vào key để lấy dữ liệu ra để truy vấn dữ liệu trong CSDL. Các CSDL dạng này có tốc độ truy vấn rất nhanh [1].

2.1.2.2 Document stores

Mỗi đối tượng sẽ được lưu trữ trong CSDL dưới dạng một document. Dữ liệu sẽ được lưu trữ dưới dạng BSON/JSON/XML dưới database. Dữ liệu không có lược đồ cứng như SQL. Do đó có thể thêm/sửa các thuộc tính, thay đổi table, ... rất nhanh và đơn giản. Mỗi quan hệ có thể được thể hiện thông qua tham chiếu (references) hoặc nhúng (embedding). Cơ sở dữ liệu dạng này có tốc độ truy vấn nhanh, có thể thực hiện các câu truy vấn phức tạp, dễ mở rộng (scalability). Mỗi cơ sở dữ liệu có một kiểu truy vấn riêng [1].

2.1.2.3 Column-Family Database

Dữ liệu được lưu trong CSDL dưới dạng các cột, thay vì các hàng như SQL.

Thiết kế và cài đặt cơ sở dữ liệu NoSQL cho hệ thống diễn đàn Hỏi-Đáp chuyên ngành với tính năng đánh giá và vinh danh.

Mỗi hàng sẽ có một key/id riêng. Điểm đặc biệt là các hàng trong một bảng sẽ có số lượng cột khác nhau. Dữ liệu được tổ chức thành các Column Families (nhóm cột) để tối ưu hiệu năng truy vấn. Một số hệ thống cung cấp ngôn ngữ truy vấn CQL với cú pháp tương tự SQL, nhưng với các hạn chế về quan hệ phức tạp [1].

2.1.2.4 Graph Database

Dữ liệu trong graph database được lưu dưới dạng các node. Mỗi node sẽ có 1 label, 1 số properties như một row trong SQL. Các node này được kết nối với nhau bằng các relationship. Graph database tập trung nhiều vào relationship giữa các node, áp dụng nhiều thuật toán duyệt node để tăng tốc độ [1].

2.1.3 Một số thuật ngữ và các khái niệm liên quan

Dù có sự khác biệt về mặt cấu trúc, giữa NoSQL (dạng Document) và CSDL quan hệ vẫn có những điểm tương đương nhau về mặt logic lưu trữ. Bảng sau đây thể hiện rõ hơn mối liên hệ giữa hai hệ thống.

Bảng 2.1: Một số thuật ngữ tương đương trong cơ sở dữ liệu quan hệ và cơ sở dữ liệu phi quan hệ [1].

STT	NoSQL (Document store)	RDBMS (CSDL quan hệ)
1	Database	Database
2	Collection	Table
3	Document	Row / Record
4	Field	Column / Attribute
5	_id	Primary Key
6	Embedding	JOIN
7	Referencing	Foreign Key

Bên cạnh các thuật ngữ trên, một số khái niệm liên quan khác là nền tảng để làm rõ hơn về cơ sở dữ liệu phi quan hệ gồm:

- Định lý CAP: Trong một hệ thống phân tán, chỉ có thể đạt được tối đa 2 trong 3 yếu tố: tính nhất quán (Consistency), tính sẵn sàng (Availability) và Khả năng chịu lỗi phân vùng (Partition Tolerance) [1].

- Mô hình BASE: Đối lập với ACID của SQL, NoSQL thường tuân theo BASE

Thiết kế và cài đặt cơ sở dữ liệu NoSQL cho hệ thống diễn đàn Hỏi-Đáp chuyên ngành với tính năng đánh giá và vinh danh.

để ưu tiên tính sẵn sàng: Basically Available (cơ bản là sẵn sàng), Soft state (trạng thái mềm, tức là có thể thay đổi mà không cần đầu vào mới) và Eventually consistent (tính nhất quán cuối cùng, dữ liệu sẽ đồng bộ sau một khoảng trễ) [1].

- Eventual Consistency (Tính nhất quán cuối cùng): Hệ thống chấp nhận việc các node bản sao có thể tạm thời không đồng bộ, nhưng cuối cùng sẽ nhất quán [1].

2.2 Tổng quan về MongoDB

MongoDB là một hệ quản trị cơ sở dữ liệu mã nguồn mở được phát triển lần đầu tiên vào năm 2007 bởi nhóm kỹ sư tại công ty 10gen (nay là MongoDB Inc.), được thiết kế để kết hợp các tính năng tốt nhất của cơ sở dữ liệu quan hệ và các kho lưu trữ khóa-giá trị (key-value stores). Nó cung cấp hiệu suất cao, tính khả dụng và khả năng mở rộng dễ dàng [3].

2.2.1 Mục đích của MongoDB

MongoDB là một hệ quản trị cơ sở dữ liệu được thiết kế để phát triển nhanh các ứng dụng web và cơ sở hạ tầng internet. Mô hình dữ liệu và các chiến lược lưu trữ được xây dựng cho thông lượng đọc và ghi cao, cùng khả năng mở rộng dễ dàng với cơ chế tự động chuyển đổi dự phòng. Lý do lớn nhất mà các nhà phát triển sử dụng MongoDB là mô hình dữ liệu trực quan của nó. MongoDB lưu trữ thông tin trong các tài liệu (documents) thay vì các hàng cho phép biểu diễn các cấu trúc dữ liệu phong phú, phân cấp [4] .

2.2.2 Ưu điểm của MongoDB so với các hệ quản trị CSDL khác

Đây là lợi thế cốt lõi và rõ ràng nhất của MongoDB so với cấu trúc bảng phẳng của SQL:

- Mô hình dữ liệu hướng tài liệu (Document-Oriented): Lưu trữ dữ liệu dưới dạng JSON/BSON giúp ánh xạ tự nhiên với các đối tượng trong mã nguồn lập trình, loại bỏ hoàn toàn sự phức tạp của các lớp chuyển đổi trung gian;

- Hiệu suất cao nhờ loại bỏ phép nối: Cấu trúc phân cấp cho phép nhúng dữ liệu và mảng ngay trong một bản ghi, giúp dữ liệu nằm liền kề nhau trên đĩa, cho phép truy xuất trọn vẹn thông tin chỉ với một thao tác đọc duy nhất;

- Tính linh hoạt của lược đồ: Không yêu cầu định nghĩa trước cấu trúc bảng, cho

Thiết kế và cài đặt cơ sở dữ liệu NoSQL cho hệ thống diễn đàn Hỏi-Đáp chuyên ngành với tính năng đánh giá và vinh danh.

phép tự do thêm trường mới hoặc lưu trữ các đối tượng đa hình (cấu trúc khác nhau) trong cùng một bộ sưu tập mà không cần chạy lệnh ALTER TABLE gây gián đoạn hệ thống;

- Khả năng mở rộng theo chiều ngang: Được thiết kế cốt lõi để tự động phân mảnh dữ liệu ra hàng loạt máy chủ giá rẻ, giúp giải quyết bài toán Big Data dễ dàng thay vì phụ thuộc vào việc nâng cấp phần cứng đắt đỏ như SQL;

- Truy vấn và Phân tích mạnh mẽ: Hỗ trợ đánh chỉ mục sâu trên mọi trường (bao gồm cả trong mảng và tài liệu con), hỗ trợ giao dịch ACID đa tài liệu và cung cấp khung tổng hợp (Aggregation Framework) để thực hiện các phép toán phân tích dữ liệu phức tạp.

2.2.3 Một số khái niệm cơ bản trong MongoDB

Một số khái niệm trong MongoDB có thể kể ra như:

- Document Data Model (Mô hình dữ liệu hướng tài liệu): MongoDB lưu trữ thông tin trong các tài liệu (documents) thay vì các hàng. Tài liệu là cấu trúc dữ liệu gồm các cặp khóa-giá trị, tương tự như JSON. Các tài liệu được nhóm lại thành các Collections (Bộ sưu tập), tương đương với bảng trong SQL nhưng không có lược đồ cố định [2];

- Document (Tài liệu): Là đơn vị dữ liệu cơ bản của MongoDB, tương đương với một hàng trong cơ sở dữ liệu quan hệ. Nó là một tập hợp các khóa được sắp xếp đi kèm với các giá trị[2];

- Collection (Bộ sưu tập): Là một nhóm các tài liệu. Các tài liệu trong cùng một collection có thể có cấu trúc khác nhau (dynamic schemas), không bắt buộc phải giống hệt nhau như trong bảng SQL [2];

- Database (Cơ sở dữ liệu): Một phiên bản MongoDB có thể chứa nhiều cơ sở dữ liệu độc lập, mỗi cơ sở dữ liệu chứa các collection riêng biệt [2];

- _id: Mọi tài liệu đều có một khóa đặc biệt là _id, giá trị của khóa này phải là duy nhất trong một collection. Nếu không được cung cấp, MongoDB sẽ tự động tạo một ObjectId cho nó;

- GridFS: Một giao thức để lưu trữ các tệp lớn (vượt quá giới hạn 16MB của tài

Thiết kế và cài đặt cơ sở dữ liệu NoSQL cho hệ thống diễn đàn Hỏi-Đáp chuyên ngành với tính năng đánh giá và vinh danh.

liệu) bằng cách chia nhỏ tệp thành các phần (chunks) và lưu trữ metadata riêng biệt.

2.2.4 Các kiểu dữ liệu trong MongoDB

Bảng 2.2: Các kiểu dữ liệu trong MongoDB

STT	Kiểu dữ liệu	Mô tả
1	Double	Kiểu số mặc định trong MongoDB shell (số thực dấu phẩy động 64-bit).
2	String	Chuỗi ký tự mã hóa UTF-8.
3	Object (Nhúng)	Một tài liệu nhúng hoàn chỉnh bên trong tài liệu cha.
4	Array	Mảng lưu trữ danh sách các giá trị hoặc đối tượng.
5	Binary data	Dữ liệu nhị phân (dùng để lưu chuỗi byte tùy ý, ví dụ: hình ảnh không dùng GridFS).
6	ObjectId	ID duy nhất 12-byte dùng làm khóa chính mặc định (_id).
7	Boolean	Giá trị đúng/sai.
8	Date	Lưu trữ ngày giờ dưới dạng số nguyên 64-bit (mili giây tính từ Unix epoch).
9	Null	Biểu thị giá trị rỗng hoặc trường không tồn tại.
10	Regular Expression	Biểu thức chính quy (Regex) dùng cho truy vấn so khớp mẫu.
11	JavaScript (Code)	Lưu trữ mã JavaScript (có thể có hoặc không có phạm vi - scope).
12	Integer (32-bit)	Số nguyên 32-bit.
13	Integer (64-bit)	Số nguyên 64-bit.
14	Timestamp	Kiểu dữ liệu đặc biệt dùng nội bộ cho việc sao chép (replication).
15	MinKey/MaxKey	Giá trị dùng nội bộ để so sánh (nhỏ hơn/lớn hơn tất cả các kiểu BSON khác).

2.2.5 Các thao tác CRUD trong MongoDB

Trong MongoDB, các thao tác CRUD (Create, Read, Update, Delete) được thực hiện thông qua các phương thức sau:

2.2.5.1 Thao tác Tạo (Create)

Nhóm lệnh dùng để thêm dữ liệu mới vào collection. Nếu _id không được cung cấp, MongoDB sẽ tự động tạo một ObjectId duy nhất cho tài liệu. Chúng bao gồm:

Thiết kế và cài đặt cơ sở dữ liệu NoSQL cho hệ thống diễn đàn Hỏi-Đáp chuyên ngành với tính năng đánh giá và vinh danh.

- insertOne: Thêm một tài liệu duy nhất.

```
db.movies.insertOne({"title": "Star Wars", "year": 1977})
```

- insertMany: Thêm một mảng gồm nhiều tài liệu cùng lúc (hiệu năng cao hơn thêm lẻ tẻ).

```
db.movies.insertMany([
  {"title": "Ghostbusters"},
  {"title": "E.T."}
])
```

2.2.5.2 Thao tác Đọc (Read)

Nhóm lệnh dùng để truy vấn và lấy dữ liệu từ collection.

- find: Trả về danh sách tất cả các tài liệu khớp với điều kiện tìm kiếm.

```
db.users.find({"username": "joe"})
```

- findOne: Chỉ trả về tài liệu đầu tiên tìm thấy khớp với điều kiện.

```
db.movies.findOne({"title": "Star Wars"})
```

2.2.5.3 Thao tác Cập nhật (Update)

Nhóm lệnh dùng để sửa đổi dữ liệu đã có.

- updateOne: Cập nhật tài liệu đầu tiên tìm thấy.

```
db.movies.updateOne(
  {"title": "Star Wars"},
  {"$set": {"reviews": []}}
)
```

- updateMany: Cập nhật toàn bộ các tài liệu khớp với điều kiện lọc.

Thiết kế và cài đặt cơ sở dữ liệu NoSQL cho hệ thống diễn đàn Hỏi-Đáp chuyên ngành với tính năng đánh giá và vinh danh.

```
db.users.updateMany(  
  {"birthday": "10/13"},  
  {"$set": {"gift": "Happy Birthday!"}}  
)
```

- replaceOne: Thay thế hoàn toàn nội dung một tài liệu bằng tài liệu mới (nhưng vẫn giữ nguyên _id).

```
db.users.replaceOne({"name": "joe"}, newDocument)
```

2.2.5.4 Thao tác Xóa (Delete)

Nhóm lệnh dùng để gỡ bỏ dữ liệu khỏi collection.

- deleteOne: Xóa tài liệu đầu tiên tìm thấy khớp với điều kiện.

```
db.movies.deleteOne({"_id": 4})
```

- deleteMany: Xóa toàn bộ các tài liệu khớp với điều kiện.

```
db.movies.deleteMany({"year": 1984})
```

2.2.6 Các toán tử cơ bản trong MongoDB

2.2.6.1 Các toán tử truy vấn

Bảng 2.3: Danh sách các toán tử truy vấn

STT	Toán tử	Mô tả	Ví dụ minh họa
1	\$eq/\$ne	Bằng/Không bằng.	{"role": {"\$ne": "admin"}}
2	\$gt/\$lt	Lớn hơn/Nhỏ hơn.	{"views": {"\$gt": 1000}}
3	\$gte \$lte	Lớn hơn hoặc bằng/ Nhỏ hơn hoặc bằng.	{"reputation": {"\$gte": 50}}
4	\$in	Giá trị nằm trong một danh sách cho trước.	{"status": {"\$in": ["Open", "Pending"]}}
5	\$or	Khớp với ít nhất một điều kiện.	{"\$or": [{"views": 0}, {"votes": 0}]}

Thiết kế và cài đặt cơ sở dữ liệu NoSQL cho hệ thống diễn đàn Hỏi-Đáp chuyên ngành với tính năng đánh giá và vinh danh.

STT	Toán tử	Mô tả	Ví dụ minh họa
6	\$and	Khớp với tất cả các điều kiện.	<code>{"\$and": [{"solved": true}, {"votes": {"\$gt": 5}}]}</code>
7	\$exists	Kiểm tra trường có tồn tại hay không.	<code>{"accepted_answer": {"\$exists": true}}</code>
8	\$all	Mảng chứa tất cả phần tử liệt kê.	<code>{"tags": {"\$all": ["mongodb", "nosql"]}}</code>
9	\$elemMatch	Tìm trong mảng các phần tử thỏa mãn điều kiện con.	<code>{"comments": {"\$elemMatch": {"user": "Hung", "likes": {"\$gt": 5}}}}</code>

2.2.6.2 Các toán tử cập nhật

Bảng 2.4: Danh sách các toán tử cập nhật

STT	Toán tử	Mô tả	Ví dụ minh họa
1	\$set	Gán giá trị mới cho trường.	<code>{"\$set": {"status": "Closed"}}</code>
2	\$unset	Xóa hoàn toàn một trường.	<code>{"\$unset": {"draft_content": 1}}</code>
3	\$inc	Tăng/Giảm giá trị số	<code>{"\$inc": {"votes": 1}}</code> (Tăng 1 vote)
4	\$push	Thêm phần tử vào cuối mảng.	<code>{"\$push": {"comments": {"text": "Hay!"}}}</code>
5	\$addToSet	Thêm vào mảng nhưng tránh trùng lặp	<code>{"\$addToSet": {"liked_by": "user123"}}</code>
6	\$pull	Xóa phần tử khỏi mảng.	<code>{"\$pull": {"comments": {"_id": "cmt01"}}}</code>

2.3 Xử lý dữ liệu nâng cao

2.3.1 Tổng quan về Aggregation Framework

Aggregation Framework là công cụ truy vấn nâng cao của MongoDB, cho phép chuyển đổi và kết hợp dữ liệu từ nhiều tài liệu để tạo ra thông tin mới không có sẵn trong từng tài liệu đơn lẻ. Nó tương đương với mệnh đề GROUP BY trong SQL nhưng linh hoạt hơn. Cơ chế hoạt động của nó dựa trên khái niệm Pipeline (Đường ống). Dữ liệu (các tài liệu) đi qua một chuỗi các bước xử lý (stages). Đầu ra của bước này sẽ là đầu vào của bước tiếp theo. Mỗi bước thực hiện một thao tác cụ thể trên các tài liệu [4]. Cấu trúc cơ bản của một câu lệnh pipeline như sau:

```
db.collection.aggregate([
    {$match: ...}, // Bước 1: Lọc dữ liệu
    {$group: ...}, // Bước 2: Nhóm dữ liệu
    {$sort: ...} // Bước 3: Sắp xếp kết quả
])
```

Bảng sau đây là danh sách các toán tử đại diện cho các giai đoạn phổ biến khi sử dụng công cụ truy vấn này.

Bảng 2.5: Bảng tổng hợp các toán tử (operators) đại diện cho các giai đoạn phổ biến nhất trong Aggregation Pipeline

STT	Tên Giai đoạn	Mô tả	Tương đương trong SQL
1	\$match	Lọc các tài liệu. Chỉ các tài liệu khớp với điều kiện mới được chuyển sang giai đoạn tiếp theo. Thường được đặt ở đầu pipeline để giảm lượng dữ liệu cần xử lý.	WHERE
2	\$project	Chỉ định các trường (fields) nào sẽ được giữ lại, loại bỏ, hoặc đổi tên trong tài liệu đầu ra. Có thể dùng để tạo trường mới từ tính toán.	SELECT
3	\$group	Nhóm các tài liệu dựa trên một khóa xác định (_id) và thực hiện các phép tính tổng hợp trên nhóm đó (ví dụ: đếm, tính tổng).	GROUP BY
4	\$unwind	"Mở rộng" một mảng. Nó tách một tài liệu chứa mảng kích thước N thành N tài liệu riêng biệt, mỗi tài liệu chứa một phần tử của mảng đó.	(Không có tương đương trực tiếp, giống JOIN với chính nó)
5	\$sort	Sắp xếp các tài liệu theo thứ tự nhất định.	ORDER BY
6	\$limit	Giới hạn số lượng tài liệu được chuyển sang giai đoạn tiếp theo.	LIMIT
7	\$skip	Bỏ qua một số lượng tài liệu nhất định.	OFFSET / SKIP

Thiết kế và cài đặt cơ sở dữ liệu NoSQL cho hệ thống diễn đàn Hỏi-Đáp chuyên ngành với tính năng đánh giá và vinh danh.

STT	Tên Giai đoạn	Mô tả	Tương đương trong SQL
8	\$out	Ghi kết quả của pipeline vào một collection mới (sẽ ghi đè nếu collection đã tồn tại). Phải là giai đoạn cuối cùng.	CREATE TABLE AS SELECT hoặc INSERT INTO
9	\$merge	(Mới từ v4.2) Ghi kết quả vào một collection (có thể là sharded). Linh hoạt hơn \$out vì có thể hợp nhất dữ liệu thay vì ghi đè hoàn toàn.	MERGE

Giai đoạn \$group là trái tim của việc tổng hợp số liệu. Trong cú pháp bắt buộc phải chỉ định trường `_id` để xác định tiêu chí nhóm. Các trường khác trong kết quả đầu ra sử dụng các Hàm tích lũy để tính toán giá trị từ nhiều tài liệu.

Bảng 2.6: Danh sách các hàm tích lũy trong giai đoạn \$group

STT	Hàm tích lũy	Mô tả	Ví dụ
1	\$sum	Tính tổng giá trị. Dùng \$sum: 1 để đếm số lượng tài liệu.	total: {\$sum: "\$price"}
2	\$avg	Tính giá trị trung bình.	averageScore: {\$avg: "\$score"}
3	\$min/\$max	Tìm giá trị nhỏ nhất/lớn nhất trong nhóm.	maxPrice: {\$max: "\$price"}
4	\$first/\$last	Lấy giá trị từ tài liệu đầu tiên hoặc cuối cùng trong nhóm (chỉ có ý nghĩa khi đi sau giai đoạn \$sort).	firstOrder: {\$first: "\$orderId"}
5	\$push	Thêm giá trị vào một mảng (giữ lại cả giá trị trùng lặp).	allTags: {\$push: "\$tags"}
6	\$addToSet	Thêm giá trị vào mảng nhưng chỉ giữ các giá trị duy nhất (loại bỏ trùng lặp).	uniqueTags: {\$addToSet: "\$tags"}

2.3.2 Tổng quan về toán tử cập nhật nguyên tử

Trong MongoDB, tất cả các cập nhật gửi đến máy chủ đều là nguyên tử (atomic) nhưng bị cô lập ở cấp độ một tài liệu. Điều này có nghĩa là một thao tác cập nhật sẽ thành công hoàn toàn hoặc thất bại hoàn toàn trên tài liệu đó; không bao giờ có trường

hợp chỉ một phần của tài liệu được cập nhật còn phần khác thì không[4].

Có hai phương pháp cập nhật:

- Cập nhật thay thế: Tải tài liệu về, sửa đổi trong ứng dụng và lưu đè lại toàn bộ tài liệu. Cách này chung chung nhưng tốn băng thông và kém hiệu quả hơn.

- Cập nhật theo toán tử (Modify by operator): sử dụng các toán tử đặc biệt (như \$set, \$inc) để chỉ sửa đổi các trường nhất định ngay trên máy chủ mà không cần gửi lại toàn bộ tài liệu. Cách này hiệu quả hơn và hỗ trợ tính nguyên tử tốt hơn.

Bảng 2.7: Các toán tử cập nhật

STT	Tên Toán tử	Mô tả chức năng
1	\$inc	Tăng (hoặc giảm) giá trị của một trường số một lượng nhất định.
2	\$set	Thiết lập giá trị cho một trường cụ thể (ghi đè hoặc tạo mới).
3	\$unset	Xóa bỏ một trường khỏi tài liệu.
4	\$rename	Đổi tên của một trường
5	\$push	Thêm một giá trị vào cuối mảng.
6	\$pushAll	Thêm một mảng các giá trị vào mảng (Lưu ý: Đã lỗi thời, nên dùng \$each).
7	\$addToSet	Thêm một giá trị vào mảng chỉ khi giá trị đó chưa tồn tại (tránh trùng lặp).
8	\$pop	Xóa phần tử đầu tiên hoặc cuối cùng của mảng.
9	\$pull	Xóa các giá trị khỏi mảng nếu khớp với một truy vấn quy định.
10	\$pullAll	Xóa nhiều giá trị cụ thể khỏi mảng.
11	\$	Toán tử vị trí: Cập nhật phần tử con trong mảng được tìm thấy bởi truy vấn (ví dụ: items.\$.qty).
12	\$each	Dùng kèm \$push và \$addToSet để thêm nhiều giá trị cùng lúc.
13	\$sort	Dùng kèm \$push, \$each, và \$slice để sắp xếp các phần tử mảng trước khi cắt bớt.

2.4 Lý thuyết về thiết kế lược đồ

2.4.1 Kỹ thuật Nhúng (Embedding)

Đây là kỹ thuật "Phi chuẩn hóa" (Denormalization). Nó thay thế khái niệm "hàng" (row) bằng mô hình linh hoạt hơn là "tài liệu" (document). Nó cho phép biểu diễn các mối quan hệ phân cấp phức tạp chỉ bằng một bản ghi duy nhất thông qua tài liệu nhúng và mảng. Nhúng giúp việc đọc dữ liệu nhanh hơn vì chỉ cần một truy vấn database là lấy được đầy đủ thông tin. Ngoài ra dữ liệu được nhúng thường ít khi bị thay đổi hay bị tác động nhiều. Tuy nhiên, việc ghi có thể chậm hơn nếu dữ liệu bị lặp lại ở nhiều nơi. Cách lưu trữ này phù hợp tự nhiên với cách các lập trình viên hướng đối tượng tư duy về dữ liệu của họ [2].

2.4.2 Kỹ thuật Tham chiếu (Referencing)

Đây là việc chia dữ liệu thành nhiều collection và tạo tham chiếu giữa chúng (giống khóa ngoại). Mỗi mảnh dữ liệu chỉ hiện hữu ở một nơi duy nhất. Kỹ thuật này được dùng khi:

- Dữ liệu phân cấp quá lớn: Tài liệu đưa ra khái niệm về số lượng quan hệ để quyết định: "Ít" thì nên Nhúng, "Nhiều" thì nên Tham chiếu. Ví dụ: Một bài viết có 50 bình luận (nhúng được), nhưng một người nổi tiếng có 5 triệu người theo dõi (phải tham chiếu).

- Dữ liệu thay đổi thường xuyên: Nếu thông tin ít khi thay đổi, nên nhúng. Nhưng nếu dữ liệu thay đổi liên tục và được sử dụng bởi nhiều tài liệu khác nhau, nên tham chiếu để tránh phải cập nhật hàng loạt.

- Extended Reference Pattern (Tham chiếu mở rộng): Đây là kỹ thuật lai. Bạn tham chiếu đến tài liệu khác nhưng đồng thời nhúng một số trường quan trọng thường xuyên được truy cập (ví dụ: lưu user_id và username trong đơn hàng để không phải join khi hiển thị danh sách đơn hàng). Điều này giúp giảm số lượng truy vấn join [5].

CHƯƠNG 3: HIỆN THỰC HÓA NGHIÊN CỨU

3.1 Phân tích yêu cầu và đặc tả chức năng

3.1.1 Mô tả

Đề tài "Thiết kế và cài đặt cơ sở dữ liệu NoSQL cho hệ thống diễn đàn Hỏi-Đáp chuyên ngành với tính năng đánh giá và vinh danh" được thực hiện nhằm mô phỏng một môi trường hỏi đáp lành mạnh thông qua cơ chế “trò chơi hóa” dựa trên Điểm uy tín (Reputation) và Huy chương (Medals) để kích thích mọi người có thể trao đổi kiến thức với nhau. Các quy tắc này chi phối trực tiếp đến cách thiết kế cơ sở dữ liệu và các thuật toán cập nhật điểm số.

Điểm uy tín là thước đo mức độ tin cậy của cộng đồng đối với người dùng. Hệ thống sử dụng cơ chế thưởng/phạt điểm dựa trên các hành động Vote và Chấp nhận câu trả lời theo bảng dưới đây:

Bảng 3.1: Cơ chế thưởng/phạt dựa trên các hành động

STT	Hành động	Đối tượng hưởng điểm	Thay đổi điểm	Ý nghĩa nghiệp vụ
1	Upvote Câu hỏi	Tác giả câu hỏi	+10	Thưởng cho việc đặt câu hỏi hay, hữu ích.
		Người Upvote	0	Khuyến khích hành vi đánh giá tích cực.
2	Downvote Câu hỏi	Tác giả câu hỏi	0	Không trừ điểm để khuyến khích người mới tự tin đặt câu hỏi.
		Người Downvote	-1	Phạt nhẹ để đảm bảo tính trách nhiệm, chống vote bừa bãi.
3	Upvote Câu trả lời	Tác giả câu trả lời	+10	Thưởng cho nội dung trả lời chất lượng, chính xác.
4	Downvote Câu trả lời	Tác giả câu trả lời	-2	Phạt nội dung sai, kém chất lượng.
		Người Downvote	-1	Đảm bảo tính trách nhiệm của người đánh giá.

Thiết kế và cài đặt cơ sở dữ liệu NoSQL cho hệ thống diễn đàn Hỏi-Đáp chuyên ngành với tính năng đánh giá và vinh danh.

STT	Hành động	Đối tượng hưởng điểm	Thay đổi điểm	Ý nghĩa nghiệp vụ
5	Chấp nhận câu trả lời	Tác giả câu trả lời	+15	Phân thưởng lớn nhất vì đã giải quyết triệt để vấn đề.
		Tác giả câu hỏi	+2	Thưởng vì đã giúp cộng đồng xác định đâu là giải pháp đúng.

Việc xếp hạng danh hiệu dựa trên tổng điểm uy tín tích lũy, hệ thống tự động phân loại người dùng thành các cấp bậc danh hiệu để vinh danh.

Bảng 3.2: Các mốc danh hiệu người dùng

STT	Tên Danh hiệu	Mức uy tín yêu cầu	Ghi chú
1	Học việc	50 điểm	Mốc đầu tiên, đánh dấu thành viên bắt đầu đóng góp tích cực.
2	Người hướng dẫn	100 điểm	Mốc trung bình, chứng minh thành viên đáng tin cậy.
3	Bậc thầy	300 điểm	Mốc cao cấp, dành cho chuyên gia có đóng góp lâu dài.

Khác với điểm uy tín, huy chương được tính dựa trên số lượng Upvote của từng câu trả lời cụ thể, nhằm ghi nhận chất lượng của từng bài đăng riêng lẻ.

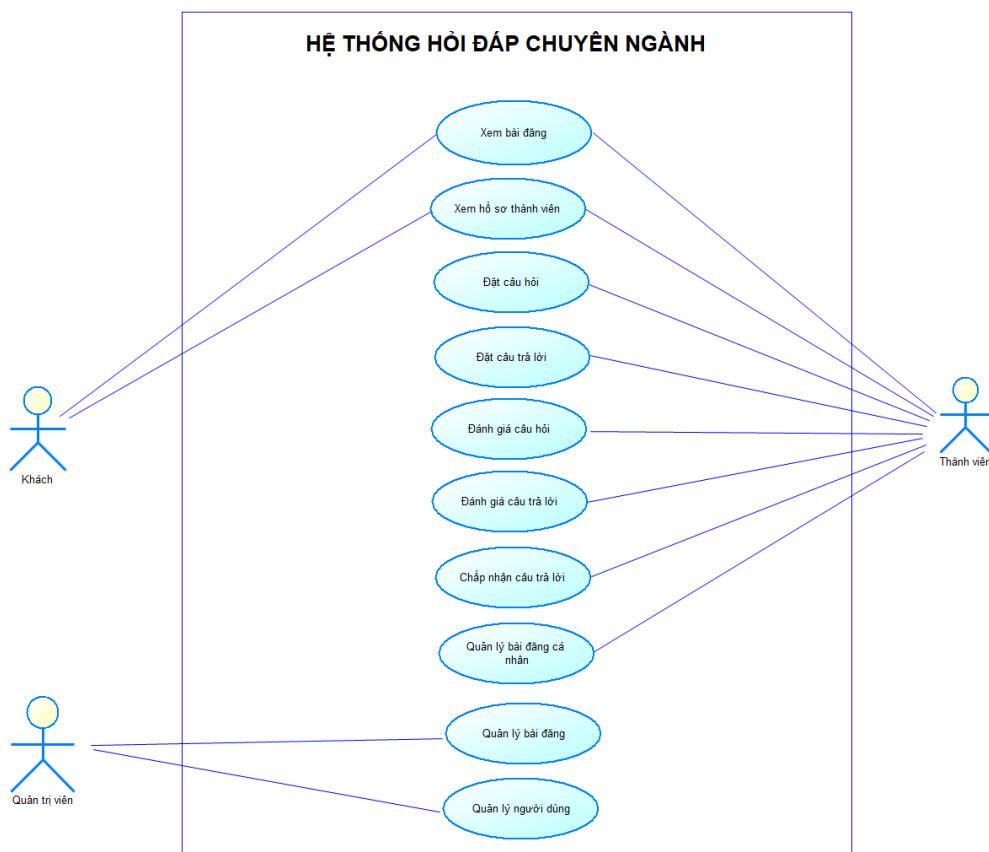
Bảng 3.3: Tiêu chí đạt huy chương cho câu trả lời

STT	Loại Huy chương	Điều kiện (Lượt Upvote)	Tương đương mức đánh giá
1	Huy chương Vàng	10 upvote	Câu trả lời Hữu ích (Nice Answer).
2	Huy chương Bạch kim	25 upvote	Câu trả lời Chất lượng tốt (Good Answer).
3	Huy chương Kim cương	100 upvote	Câu trả lời Xuất sắc (Great Answer).

3.1.2 Sơ đồ Use Case tổng quát

Hệ thống được thiết kế để phục vụ ba nhóm tác nhân chính: Khách, Thành viên và Quản trị viên.

3.1.2.1 Sơ đồ Use Case tổng quát



Hình 3.1: Sơ đồ Use Case tổng quát cho hệ thống

3.1.2.2 Danh sách tác nhân

Khách: Người dùng chưa đăng nhập, chỉ có quyền xem nội dung công khai, tác nhân này trong hệ thống hiện tại không gây ảnh hưởng cho cơ sở dữ liệu.

Thành viên: Người dùng đã đăng nhập, có toàn quyền tham gia hỏi, đáp, đánh giá và quản lý trang cá nhân.

Quản trị viên: Người quản lý hệ thống, có quyền can thiệp vào nội dung và tài khoản người dùng.

3.1.2.3 Đặc tả các Use Case chính

Dưới đây là đặc tả các chức năng nghiệp vụ cốt lõi liên quan đến luồng dữ liệu của đồ án.

Bảng 3.4: Tổng hợp các đặc tả use case của hệ thống

STT	Use Case và Tác nhân	Mô tả & Luồng xử lý chính	Hậu điều kiện (Post-conditions)
1	UC01: Xem bài đăng (Khách, Thành viên)	Người dùng chọn xem chi tiết câu hỏi. Hệ thống tải dữ liệu câu hỏi, danh sách câu trả lời, bình luận và tăng lượt xem.	view_count của câu hỏi tăng lên 1.
2	UC02: Xem hồ sơ (Khách, Thành viên)	Xem thông tin cá nhân và thành tích. Hệ thống tổng hợp điểm uy tín, thống kê số lượng bài đăng và danh sách huy chương theo thẻ.	Không có dữ liệu thay đổi.
3	UC03: Đặt câu hỏi (Thành viên)	Thành viên nhập Tiêu đề, Nội dung, Thẻ (Tags). Hệ thống kiểm tra hợp lệ và lưu trữ vào CSDL.	Document mới được tạo trong collection questions.
4	UC04: Đặt câu trả lời (Thành viên)	Thành viên gửi giải pháp cho một câu hỏi. Hệ thống lưu câu trả lời và liên kết với ID câu hỏi.	Document mới trong answers. answer_count trong câu hỏi tăng 1.
5	UC05: Đánh giá câu hỏi (Thành viên - trừ tác giả)	Người dùng Upvote/Downvote. Hệ thống kiểm tra trùng lặp, cập nhật điểm câu hỏi và điểm uy tín tác giả.	vote_score câu hỏi thay đổi. reputation tác giả thay đổi. Lưu vết người vote.
6	UC06: Đánh giá câu trả lời (Thành viên - trừ tác giả)	Tương tự UC05 nhưng áp dụng cho câu trả lời. Hệ thống cập nhật điểm tương ứng.	vote_score câu trả lời thay đổi. reputation tác giả thay đổi.
7	UC07: Chấp nhận câu trả lời (Chủ câu hỏi)	Tác giả đánh dấu một câu trả lời là giải pháp đúng nhất. Hệ thống cộng thưởng điểm lớn cho cả hai bên.	is_accepted = true. Cộng điểm reputation lớn cho người trả lời.

Thiết kế và cài đặt cơ sở dữ liệu NoSQL cho hệ thống diễn đàn Hỏi-Đáp chuyên ngành với tính năng đánh giá và vinh danh.

STT	Use Case và Tác nhân	Mô tả & Luồng xử lý chính	Hậu điều kiện (Post-conditions)
8	UC08: Quản lý bài cá nhân (Tác giả bài đăng)	Thành viên sửa hoặc xóa bài của mình. Nếu xóa, hệ thống chỉ ẩn bài (Soft Delete) để bảo toàn dữ liệu.	Nội dung cập nhật hoặc status chuyển sang deleted / hidden.
9	UC09: Quản lý bài đăng (Admin)	Quản trị viên xem và xử lý các bài vi phạm (Xóa/Ẩn/Khóa) dựa trên báo cáo.	status của bài đăng chuyển sang banned / hidden.
10	UC10: Quản lý người dùng (Admin)	Quản trị viên tìm kiếm và Khóa (Ban) tài khoản thành viên vi phạm quy định.	status của người dùng chuyển sang banned.

3.2 Thiết kế lược đồ cơ sở dữ liệu

Dựa vào sơ đồ use case và đặc tả chức năng, đề tài thiết kế lược đồ dữ liệu theo hướng phi chuẩn hóa nhằm tối ưu hóa tác vụ truy vấn cho các chức năng được miêu tả ở trên.

3.2.1 Mô hình dữ liệu

Hệ thống sử dụng 3 collection chính: users, questions và answers.

Collection users: lưu trữ thông tin tài khoản và các thông số thành tích vinh danh.

Cấu trúc được thể hiện như sau:

```
{  
  1. _id: <objectId>,  
  2. username: <string>,  
  3. password: <string>,  
  4. role: <string>,  
  5. email: <string>,  
  6. joined_at: <date>,  
  7. status: <string>,  
  8. reputation: <int>,  
  9. stats:
```



```
{
    9.1. question_count: <int>,
    9.2. answer_count: <int>,
    9.3. accepted_answer_count: <int>
},
10. medal_counts:
{
    10.1. gold: <int>,
    10.2. platinum: <int>,
    10.3. diamond: <int>
}
}
```

Collection questions: Lưu trữ câu hỏi. Đặc biệt có những thông tin tác giả và danh sách người vote. Cấu trúc được thể hiện như sau:

```
{
    1. _id: <objectId>,
    2. title: <string>,
    3. body: <string>,
    4. tags: <[string]>,
    5. created_at: <date>,
    6. status: <string>,
    7. author_info:
    {
        7.1. author_id: <objectId>,
        7.2. username: <string>,
    },
    8. vote_score: <int>,
    9. answer_count: <int>,
    10. voters: [
```

```
{  
    10.1. voter_id: <objectId>,  
    10.2. value: <int>  
}  
],  
}
```

Collection answers Lưu trữ câu trả lời, liên kết với câu hỏi thông qua question_id. Cấu trúc được thể hiện như sau:

```
{  
    1. _id: <objectId>,  
    2. question_id: <objectId>,  
    3. body: <string>,  
    4. created_at: <date>,  
    5. author_info:  
        {  
            5.1. author_id: <objectId>,  
            5.2. username: <string>,  
        },  
    6. vote_score: <int>,  
    7. is_accepted: <bool>,  
    8. voters: [  
        {  
            8.1. voter_id: <objectId>,  
            8.2. value: <int>  
        }  
    ],  
}
```

3.2.2 Chiến lược thiết kế lược đồ

Dựa vào phần 2.4, ta có thể giải thích được chiến lược thiết kế lược đồ trên như sau:

- Nhúng thông tin tác giả (Author Info) để tối ưu hóa hiệu năng đọc và loại bỏ hoàn toàn các phép nối (JOIN) tốn kém giữa các collection khi lượng truy cập tăng cao, hệ thống áp dụng kỹ thuật nhúng một tập con dữ liệu tác giả (gồm `_id`, `username`, `reputation`) trực tiếp vào document bài viết. Chiến lược này tận dụng đặc thù ít biến động của thông tin người dùng, giúp hệ thống chỉ cần thực hiện một truy vấn `find()` duy nhất là có thể trích xuất đầy đủ nội dung và thông tin cần thiết.

- Nhúng danh sách người đánh giá (Voters) nhằm đảm bảo tính nhất quán dữ liệu và ngăn chặn tình trạng spam vote, danh sách người đánh giá (voters) được thiết kế nhúng ngay bên trong document bài viết thay vì tách thành bảng riêng biệt như mô hình SQL. Giải pháp này cho phép sử dụng toán tử cập nhật nguyên tử để kiểm tra quyền vote và cập nhật dữ liệu, đồng thời tăng tốc độ xử lý do chỉ cần quét mảng cục bộ thay vì quét toàn bộ bảng dữ liệu lớn.

- Lưu trữ sẵn các giá trị tổng hợp như `vote_score` và `medal_counts` ngay trong document thay vì phải đếm lại số phần tử mảng mỗi khi hiển thị. Các trường này được cập nhật tự động theo thời gian thực thông qua toán tử `$inc` mỗi khi có tương tác mới, giúp chuyển tải trọng tính toán sang thao tác ghi, từ đó tối ưu hóa tuyệt đối tác vụ truy vấn.

3.3 Môi trường và công cụ thực hiện

Đồ án sử dụng các công cụ và môi trường kỹ thuật chuyên dụng nhằm hiện thực hóa các quy trình xử lý dữ liệu, đồng thời kiểm chứng tính hiệu quả của lược đồ cơ sở dữ liệu đã thiết kế.

Đối với giải pháp phần mềm, hệ thống vận hành trên nền tảng MongoDB Community Server phiên bản 8.2.1, đóng vai trò là cơ sở dữ liệu trung tâm chịu trách nhiệm lưu trữ, thực thi các truy vấn CRUD và xử lý các luồng tổng hợp dữ liệu. Bên cạnh đó, công cụ MongoDB Compass phiên bản 1.48.2 được sử dụng làm giao diện đồ họa (GUI) chính thức để quản lý và tương tác với hệ thống, hỗ trợ đắc lực trong việc

Thiết kế và cài đặt cơ sở dữ liệu NoSQL cho hệ thống diễn đàn Hỏi-Đáp chuyên ngành với tính năng đánh giá và vinh danh.

trực quan hóa dữ liệu JSON, xây dựng câu lệnh truy vấn, tích hợp sẵn môi trường dòng lệnh MongoDB Shell và phân tích hiệu năng thông qua tính năng Explain Plan.

Hệ thống được triển khai và đánh giá trên môi trường phần cứng máy tính cá nhân (Laptop) với cấu hình chi tiết như sau:

- CPU: 11th Gen Intel(R) Core(TM) i5-11400H @ 2.70GHz.
- RAM: 8.00 GB.
- Hệ điều hành: Windows 11 Home Single Language (64-bit).

3.4 Chiến lược sinh dữ liệu mẫu

Trong quá trình kiểm thử phần mềm, việc tạo bộ dữ liệu mẫu đóng vai trò quan trọng. Các phương pháp truyền thống như nhập liệu thủ công thường tốn kém thời gian và dễ xảy ra sai sót. Trong khi đó, các thư viện sinh dữ liệu tự động phổ biến (như Faker.js) lại hoạt động dựa trên cơ chế ngẫu nhiên, dẫn đến dữ liệu thiếu tính liên kết logic và nội dung thường vô nghĩa (dạng Lorem Ipsum).

Để khắc phục các hạn chế trên, đề án đã lựa chọn giải pháp sử dụng Mô hình Ngôn ngữ Lớn (LLM - cụ thể là Gemini) để hỗ trợ sinh dữ liệu. Dữ liệu không được sinh ra ngẫu nhiên mà tuân thủ nghiêm ngặt các quy tắc kiểm thử sau:

- Dữ liệu có ý nghĩa: Thay vì sử dụng các văn bản giả gây khó khăn cho việc hình dung sản phẩm, AI được yêu cầu sinh ra các nội dung câu hỏi và câu trả lời chuyên sâu về lĩnh vực CNTT. Điều này giúp dữ liệu có tính thực tế cao, mang lại trải nghiệm như một diễn đàn công nghệ đang vận hành thực sự.

- Tránh sai lệch số liệu: Đối với chức năng tính điểm, sự sai lệch giữa điểm số hiển thị và số lượng người vote thực tế là lỗi thường gặp khi nhập liệu thủ công. Giải pháp được áp dụng là yêu cầu AI tự động tổng hợp danh sách người vote (tạo lượt vote ảo, tức là giả lập các vote gộp từ hệ thống cũ), sau đó điền con số chính xác tuyệt đối vào trường điểm số, đảm bảo tính toàn vẹn dữ liệu.

- Tạo các tình huống thử nghiệm: Để kiểm chứng độ thông minh của các câu truy vấn, dữ liệu mẫu bao gồm các tình huống "bẫy" thay vì chỉ có các trường hợp lý tưởng. Cụ thể, hệ thống giả lập các trường hợp người dùng đồng hạng điểm số hoặc tài khoản có thành tích cao nhưng bị khóa, nhằm chứng minh khả năng xử lý logic nghiệp

vụ của hệ thống.

- Thông thường, các ObjectId trong MongoDB được sinh ngẫu nhiên tại thời điểm tạo các Document. Tuy nhiên, điều này gây khó khăn cho việc thiết lập các mối quan hệ tham chiếu giữa các Collection trong bộ dữ liệu kiểm thử. Do đó, thay vì để hệ thống tự sinh ID, dữ liệu mẫu sẽ có các chuỗi ObjectId được tạo thủ công theo quy luật (Giả sử user thứ nhất có ID kết thúc bằng ...0001, user thứ hai là ...0002,...). Điều đó giúp dễ dàng nhận diện và truy vấn đối tượng trong quá trình thực hiện, thuận tiện trong quá trình demo.

CHƯƠNG 4: KẾT QUẢ NGHIÊN CỨU

4.1 Triển khai dữ liệu mẫu

Dựa trên các tiêu chí ở mục 3.4, bộ dữ liệu mẫu được khởi tạo vào MongoDB như sau:

4.1.1 Quy trình khởi tạo

Hệ thống cơ sở dữ liệu được khởi tạo với tên gọi `csn_forumDB`. Quá trình khởi tạo dữ liệu không thực hiện thủ công mà được tự động hóa hoàn toàn thông qua MongoDB Shell được tích hợp sẵn trong MongoDB Compass.

Khi tạo một Collection mới, sử dụng câu lệnh `db.collection.insertMany([])` để thực hiện đồng thời hai tác vụ:

- Tự động khởi tạo Collection: Các collection `users`, `questions`, `answers` tự động được sinh ra ngay khi document đầu tiên được chèn vào.
- Nạp dữ liệu hàng loạt: Đưa toàn bộ dữ liệu mẫu vào hệ thống trong một lần thực thi duy nhất, giúp giảm thiểu thời gian kết nối và đảm bảo tính nguyên vẹn của dữ liệu.

```
>_MONGOSH
csn_forumDB> db.users.insertMany([
  {
    _id: '67500000000000000000000001',
    username: 'admin_root',
    password: 'Admin@123',
    role: 'admin',
    email: 'admin@example.com',
    joined_at: '2024-01-10T09:00:00Z',
    status: 'active',
    reputation: 0,
    stats: { question_count: 0, answer_count: 0, accepted_answer_count: 0 },
    medal_counts: { gold: 0, platinum: 0, diamond: 0 }
  },
  {
    _id: '67500000000000000000000002',
    username: 'Nguyễn Văn An',
    password: '123456789',
    role: 'user',
    email: 'nguyenvan@example.com',
    joined_at: '2024-01-10T09:00:00Z',
    status: 'active',
    reputation: 0,
    stats: { question_count: 0, answer_count: 0, accepted_answer_count: 0 },
    medal_counts: { gold: 0, platinum: 0, diamond: 0 }
  }
])
```

Hình 4.1: Nạp dữ liệu trên MongoDB Shell

```
>_MONGOSH
< {
  acknowledged: true,
  insertedIds: {
    '0': '67500000000000000000000001',
    '1': '67500000000000000000000002',
    '2': '67500000000000000000000003',
    '3': '67500000000000000000000004',
    '4': '67500000000000000000000005',
    '5': '67500000000000000000000006',
    '6': '67500000000000000000000007',
    '7': '67500000000000000000000008',
    '8': '67500000000000000000000009',
    '9': '6750000000000000000000000a'
  }
}
```

Hình 4.2: Thông báo kết quả thực thi thành công

4.1.2 Dữ liệu cho users

```
[
  {
    _id: '67500000000000000000000001',
    username: 'admin_root',
    password: 'Admin@123',
    role: 'admin',
    email: 'admin@example.com',
    joined_at: '2024-01-10T09:00:00Z',
    status: 'active',
    reputation: 0,
    stats: { question_count: 0, answer_count: 0, accepted_answer_count: 0 },
    medal_counts: { gold: 0, platinum: 0, diamond: 0 }
  },
  {
    _id: '67500000000000000000000002',
```

```
username: 'Nguyễn Văn An',
password: 'User@12345',
role: 'member',
email: 'nguyenvana@example.com',
joined_at: '2024-02-15T14:20:00Z',
status: 'active',
reputation: 115,
stats: { question_count: 5, answer_count: 6, accepted_answer_count: 3 },
medal_counts: { gold: 3, platinum: 1, diamond: 0 }
},
{
  _id: '675000000000000000000003',
  username: 'Trần Thị Bình',
  password: 'User@23456',
  role: 'member',
  email: 'tranthib@example.com',
  joined_at: '2024-03-02T08:45:00Z',
  status: 'active',
  reputation: 115,
  stats: { question_count: 5, answer_count: 6, accepted_answer_count: 3 },
  medal_counts: { gold: 3, platinum: 2, diamond: 1 }
},
{
  _id: '675000000000000000000004',
  username: 'Lê Quang Cường',
```



```
password: 'User@34567',
role: 'member',
email: 'lequangc@example.com',
joined_at: '2024-03-20T19:10:00Z',
status: 'active',
reputation: 82,
stats: { question_count: 2, answer_count: 4, accepted_answer_count: 2 },
medal_counts: { gold: 2, platinum: 1, diamond: 0 }
},
{
  _id: '675000000000000000000005',
  username: 'Phạm Minh Đức',
  password: 'User@45678',
  role: 'member',
  email: 'phamminhd@example.com',
  joined_at: '2024-04-05T11:30:00Z',
  status: 'active',
  reputation: 59,
  stats: { question_count: 2, answer_count: 3, accepted_answer_count: 1 },
  medal_counts: { gold: 1, platinum: 0, diamond: 0 }
},
{
  _id: '675000000000000000000006',
  username: 'Hoàng Nam Anh',
  password: 'User@56789',
```

```
role: 'member',

email: 'hoangname@example.com',

joined_at: '2024-05-12T16:05:00Z',

status: 'active',

reputation: 62,

stats: { question_count: 2, answer_count: 4, accepted_answer_count: 1 },

medal_counts: { gold: 1, platinum: 1, diamond: 0 }

},

{

  _id: '675000000000000000000007',

  username: 'Vũ Hoài Phúc',

  password: 'User@67890',

  role: 'member',

  email: 'vuhoaif@example.com',

  joined_at: '2024-06-01T09:15:00Z',

  status: 'active',

  reputation: 29,

  stats: { question_count: 0, answer_count: 3, accepted_answer_count: 0 },

  medal_counts: { gold: 1, platinum: 0, diamond: 0 }

},

{

  _id: '675000000000000000000008',

  username: 'Linh Chi',

  password: 'User@78901',

  role: 'member',
```

```
email: 'linhchig@example.com',
joined_at: '2024-06-18T22:40:00Z',
status: 'active',
reputation: 8,
stats: { question_count: 0, answer_count: 1, accepted_answer_count: 0 },
medal_counts: { gold: 0, platinum: 0, diamond: 0 }
},
{
  _id: '675000000000000000000009',
  username: 'Tuân Anh',
  password: 'User@89012',
  role: 'member',
  email: 'tuananh@example.com',
  joined_at: '2024-07-07T13:55:00Z',
  status: 'banned',
  reputation: 500,
  stats: { question_count: 1, answer_count: 1, accepted_answer_count: 0 },
  medal_counts: { gold: 0, platinum: 0, diamond: 0 }
},
{
  _id: '67500000000000000000000a',
  username: 'Khoái',
  password: 'User@90123',
  role: 'member',
  email: 'khoai@example.com',
```

```
joined_at: '2024-07-21T07:25:00Z',
status: 'banned',
reputation: 115,
stats: { question_count: 0, answer_count: 1, accepted_answer_count: 0 },
medal_counts: { gold: 0, platinum: 0, diamond: 0 }
}
]
```

4.1.3 Dữ liệu cho questions

```
[
{
  _id: '675100000000000000000001',
  title: 'Thiết kế schema MongoDB cho hệ thống hỏi đáp giống StackOverflow?',
  body: 'Em đang làm đồ án hệ thống hỏi đáp chuyên ngành, cần thiết kế schema MongoDB cho users, questions, answers. Nên nhúng (embed) hay tham chiếu (reference) để tối ưu cho việc truy vấn danh sách câu hỏi và câu trả lời?',
  tags: [ 'mongodb', 'schema-design', 'database' ],
  created_at: '2024-08-01T09:15:00Z',
  status: 'open',
  author_info: { author_id: '675000000000000000000002', username: 'Nguyễn Văn An' },
  vote_score: 2,
  answer_count: 3,
  voters: [
    { voter_id: '675000000000000000000003', value: 1 },
    { voter_id: '675000000000000000000004', value: 1 },
  ]
}
```

```
{ voter_id: '67500000000000000000000005', value: 1 },
  { voter_id: '67500000000000000000000006', value: -1 }
]
},
{
  _id: '67510000000000000000000002',
  title: 'Khi nào nên dùng MongoDB thay vì MySQL cho web dịch vụ?',
  body: 'Dự án web dịch vụ có nhiều lượt đọc, ít ghi, dữ liệu không quá phức tạp về
quan hệ. Trong trường hợp nào nên chọn MongoDB thay vì MySQL, và có các trade-
off gì về transaction, join, consistency?',
  tags: [ 'mongodb', 'mysql', 'architecture' ],
  created_at: '2024-08-03T14:20:00Z',
  status: 'open',
  author_info: { author_id: '67500000000000000000000003', username: 'Trần Thị Bình'
},
  vote_score: 1,
  answer_count: 2,
  voters: [
    { voter_id: '67500000000000000000000002', value: 1 },
    { voter_id: '67500000000000000000000004', value: 1 },
    { voter_id: '67500000000000000000000005', value: -1 }
  ]
},
{
  _id: '67510000000000000000000003',
```

title: 'So sánh ưu nhược điểm giữa mô hình quan hệ và NoSQL trong đồ án Cơ sở ngành?',

body: 'Thầy yêu cầu phân tích lý do vì sao chọn MongoDB cho đồ án Cơ sở ngành thay vì dùng hoàn toàn MySQL. Cần liệt kê ưu nhược điểm chính của từng mô hình và ví dụ tình huống áp dụng.',

tags: ['database', 'nosql', 'relational-db'],

created_at: '2024-08-05T08:45:00Z',

status: 'hidden',

author_info: { author_id: '675000000000000000000004', username: 'Lê Quang Cường' },

vote_score: 0,

answer_count: 0,

voters: [

{ voter_id: '675000000000000000000002', value: 1 },

{ voter_id: '675000000000000000000003', value: -1 }

]

},

{

_id: '675100000000000000000004',

title: 'Thiết kế index cho collection questions để tối ưu tìm kiếm theo tag?',

body: 'Em có collection questions với trường tags là mảng string. Khi search theo một hoặc nhiều tag thì nên tạo index như thế nào cho hiệu quả? Có nên dùng compound index với created_at không?',

tags: ['mongodb', 'indexing', 'performance'],

created_at: '2024-08-10T19:10:00Z',

status: 'open',

```
author_info: { author_id: '675000000000000000000002', username: 'Nguyễn Văn
An' },

vote_score: 3,

answer_count: 4,

voters: [

  { voter_id: '67500000000000000000000003', value: 1 },

  { voter_id: '67500000000000000000000004', value: 1 },

  { voter_id: '67500000000000000000000006', value: 1 }

]

},

{

  _id: '67510000000000000000000005',

  title: 'Lưu lịch sử chỉnh sửa câu hỏi trong MongoDB như thế nào?',

  body: 'Hệ thống hỏi đáp cần lưu lại lịch sử chỉnh sửa câu hỏi (title, body). Nên lưu
toàn bộ bản ghi cũ vào một collection riêng hay dùng mảng embedded trong document
chính?',

  tags: [ 'mongodb', 'audit-log', 'design' ],

  created_at: '2024-08-15T11:30:00Z',

  status: 'open',

  author_info: { author_id: '67500000000000000000000005', username: 'Phạm Minh
Đức' },

  vote_score: 2,

  answer_count: 1,

  voters: [

    { voter_id: '67500000000000000000000002', value: 1 },
```

```
{ voter_id: '675000000000000000000003', value: 1 }

],

{

  _id: '675100000000000000000006',

  title: 'Cách phân trang (pagination) hiệu quả với MongoDB cho danh sách câu hỏi?',

  body: 'Khi hiển thị danh sách câu hỏi mới nhất, em dùng skip/limit thì bị chậm khi dữ liệu lớn. Có giải pháp nào tốt hơn cho pagination trong MongoDB, ví dụ dùng _id hoặc created_at làm cursor?',

  tags: [ 'mongodb', 'pagination', 'performance' ],

  created_at: '2024-08-20T16:05:00Z',

  status: 'open',

  author_info: { author_id: '675000000000000000000006', username: 'Hoàng Nam Anh' },

  vote_score: 4,

  answer_count: 2,

  voters: [

    { voter_id: '000000000000000000000000', value: 3 },

    { voter_id: '675000000000000000000003', value: 1 },

  ]

},

{

  _id: '675100000000000000000007',

  title: 'Tính điểm reputation cho user dựa trên vote của câu hỏi và câu trả lời?',

  body: 'Em muốn thiết kế rule tính điểm reputation giống StackOverflow: khi câu trả lời được upvote, accepted answer, hoặc câu hỏi bị downvote thì cộng/trừ bao nhiêu
```


điểm là hợp lý cho đề án?',

tags: ['reputation', 'gamification', 'system-design'],

created_at: '2024-08-25T09:15:00Z',

status: 'open',

author_info: { author_id: '675000000000000000000002', username: 'Nguyễn Văn An' },

vote_score: 2,

answer_count: 3,

voters: [

{ voter_id: '675000000000000000000003', value: 1 },

{ voter_id: '675000000000000000000004', value: 1 },

{ voter_id: '675000000000000000000005', value: 1 },

{ voter_id: '675000000000000000000006', value: -1 }

]

},

{

_id: '675100000000000000000008',

title: 'Có nên cho phép admin tham gia vote và trả lời trong hệ thống hỏi đáp?',

body: 'Trong đặc tả hệ thống hiện tại, admin chỉ có quyền quản trị, không được đặt câu hỏi, trả lời hoặc vote. Thiết kế như vậy có bất lợi gì không so với việc để admin cũng là một user bình thường?',

tags: ['role', 'authorization', 'design-decision'],

created_at: '2024-09-01T22:40:00Z',

status: 'hidden',

author_info: { author_id: '675000000000000000000003', username: 'Trần Thị Bình' }

```
},  
  
  vote_score: 0,  
  
  answer_count: 0,  
  
  voters: [  
  
    { voter_id: '675000000000000000000002', value: 1 },  
  
    { voter_id: '675000000000000000000004', value: -1 }  
  
  ]  
  
},  
  
{  
  
  _id: '67510000000000000000000009',  
  
  title: 'Thiết kế chức năng ẩn (hidden) câu hỏi do vi phạm nội quy?',  
  
  body: 'Em muốn khi câu hỏi bị report nhiều lần thì tự động chuyển status sang  
hidden nhưng không xóa hẳn để giữ log. Có nên tạo thêm collection reports hay chỉ  
cần lưu số lần report trong questions?',  
  
  tags: [ 'moderation', 'status', 'system-design' ],  
  
  created_at: '2024-09-05T13:55:00Z',  
  
  status: 'open',  
  
  author_info: { author_id: '675000000000000000000004', username: 'Lê Quang  
Cường' },  
  
  vote_score: 3,  
  
  answer_count: 1,  
  
  voters: [  
  
    { voter_id: '675000000000000000000002', value: 1 },  
  
    { voter_id: '675000000000000000000005', value: 1 },  
  
    { voter_id: '675000000000000000000007', value: 1 }  
  ]  
}
```

```
]
},
{
  _id: '67510000000000000000000a',
  title: 'Cách tổ chức tag cho câu hỏi: dùng tiếng Việt hay tiếng Anh?',
  body: 'Trong hệ thống hỏi đáp chuyên ngành CNTT, nên chuẩn hóa tag theo tiếng Việt hay tiếng Anh để thuận tiện cho người dùng tìm kiếm và mở rộng sau này?',
  tags: [ 'tags', 'ux', 'best-practice' ],
  created_at: '2024-09-10T07:25:00Z',
  status: 'open',
  author_info: { author_id: '675000000000000000000005', username: 'Phạm Minh Đức' },
  vote_score: 2,
  answer_count: 4,
  voters: [
    { voter_id: '675000000000000000000002', value: 1 },
    { voter_id: '675000000000000000000003', value: 1 }
  ]
},
{
  _id: '67510000000000000000000b',
  title: 'Làm sao để demo tính năng full-text search trong MongoDB cho đồ án?',
  body: 'MongoDB có hỗ trợ full-text search cơ bản. Em muốn dựng demo tìm kiếm theo title và body câu hỏi, có highlight từ khóa thì cần cấu hình index và truy vấn như thế nào?'
```

```
tags: [ 'mongodb', 'full-text-search' ],

created_at: '2024-09-15T09:00:00Z',

status: 'open',

author_info: { author_id: '675000000000000000000006', username: 'Hoàng Nam
Anh' },

vote_score: 2,

answer_count: 2,

voters: [

  { voter_id: '67500000000000000000000002', value: 1 },

  { voter_id: '67500000000000000000000003', value: 1 },

  { voter_id: '67500000000000000000000004', value: 1 },

  { voter_id: '67500000000000000000000008', value: -1 }

]

},

{

  _id: '67510000000000000000000000c',

  title: 'Có nên lưu thông tin voters của câu hỏi trong mảng hay tách collection riêng?',

  body: 'Hiện em đang lưu danh sách người vote cho câu hỏi trong mảng voters
(voter_id, value). Khi số lượng vote lớn thì việc cập nhật có bị chậm không, có cần
tách riêng thành collection votes?',

  tags: [ 'mongodb', 'schema-design', 'scalability' ],

  created_at: '2024-09-20T10:30:00Z',

  status: 'open',

  author_info: { author_id: '67500000000000000000000002', username: 'Nguyễn Văn
An' },
```

```
    vote_score: 2,

    answer_count: 2,

    voters: [

        { voter_id: '675000000000000000000003', value: 1 },

        { voter_id: '675000000000000000000004', value: 1 }

    ]

},

{

    _id: '675100000000000000000000d',

    title: 'Thiết kế API lấy danh sách câu hỏi kèm số lượng câu trả lời?',

    body: 'Trong REST API cho frontend, em muốn endpoint trả về danh sách câu hỏi kèm số lượng câu trả lời tương ứng. Nên dùng aggregation lookup trên collection answers hay lưu sẵn answer_count trong questions?',

    tags: [ 'api-design', 'aggregation', 'mongodb' ],

    created_at: '2024-09-25T15:45:00Z',

    status: 'open',

    author_info: { author_id: '675000000000000000000003', username: 'Trần Thị Bình'

},

    vote_score: 3,

    answer_count: 1,

    voters: [

        { voter_id: '675000000000000000000002', value: 1 },

        { voter_id: '675000000000000000000004', value: 1 },

        { voter_id: '675000000000000000000005', value: 1 }

    ]

}
```

```
    },  
    {  
      _id: '675100000000000000000000e',  
      title: 'Xử lý user bị banned nhưng vẫn còn câu hỏi và câu trả lời cũ?',  
      body: 'Khi user bị banned, hệ thống nên xử lý các câu hỏi và câu trả lời cũ như thế nào? Có nên ẩn toàn bộ nội dung của họ hay chỉ đánh dấu trạng thái user là banned để giữ lịch sử?',  
      tags: [ 'moderation', 'user-status' ],  
      created_at: '2024-10-01T08:20:00Z',  
      status: 'open',  
      author_info: { author_id: '6750000000000000000000009', username: 'Tuấn Anh' },  
      vote_score: 1,  
      answer_count: 2,  
      voters: [  
        { voter_id: '6750000000000000000000002', value: 1 },  
        { voter_id: '6750000000000000000000003', value: 1 },  
        { voter_id: '6750000000000000000000004', value: -1 }  
      ]  
    },  
    {  
      _id: '675100000000000000000000f',  
      title: "Gợi ý cấu trúc demo dữ liệu 'vừa đủ' cho đồ án hệ thống hỏi đáp?",  
      body: 'Đồ án yêu cầu có dữ liệu demo nhưng không cần quá lớn. Khoảng bao nhiêu users, questions, answers là hợp lý để thể hiện được tính năng tìm kiếm, phân trang, vote và thống kê?',
```

```
tags: [ 'demo-data', 'system-design' ],
created_at: '2024-10-05T09:50:00Z',
status: 'open',
author_info: { author_id: '675000000000000000000002', username: 'Nguyễn Văn
An' },
vote_score: 4,
answer_count: 1,
voters: [
  { voter_id: '675000000000000000000003', value: 1 },
  { voter_id: '675000000000000000000004', value: 1 },
  { voter_id: '675000000000000000000005', value: 1 },
  { voter_id: '675000000000000000000006', value: 1 }
]
}
```

4.1.4 Dữ liệu cho answers

```
[
  {
    _id: '675200000000000000000001',
    question_id: '675100000000000000000001',
    body: 'Bạn có thể thiết kế schema dạng lai...',
    created_at: '2024-08-01T10:00:00Z',
    author_info: { author_id: '675000000000000000000003', username: 'Trần Thị Bình'
  },
  vote_score: 15, // Đã khớp: 12 + 1 + 1 + 1 = 15
}
```

```
is_accepted: true,

voters: [

  { voter_id: '000000000000000000000000', value: 12 }, // Sửa vote ảo lên 12 cho
khớp

  { voter_id: '675000000000000000000002', value: 1 },

  { voter_id: '675000000000000000000004', value: 1 },

  { voter_id: '675000000000000000000005', value: 1 }

],

},

{

  _id: '675200000000000000000002',

  question_id: '675100000000000000000001',

  body: 'Nếu quy mô đồ án nhỏ...',

  created_at: '2024-08-01T10:30:00Z',

  author_info: { author_id: '675000000000000000000004', username: 'Lê Quang
Cường' },

  vote_score: 0,

  is_accepted: false,

  voters: [

    { voter_id: '675000000000000000000002', value: 1 },

    { voter_id: '675000000000000000000003', value: -1 }

  ]

},

{

  _id: '675200000000000000000003',
```



```
question_id: '67510000000000000000000000000001',
body: 'Trong đồ án Cơ sở ngành...',
created_at: '2024-08-01T11:00:00Z',
author_info: { author_id: '67500000000000000000000000000006', username: 'Hoàng Nam Anh' },
vote_score: 1,
is_accepted: false,
voters: [ { voter_id: '67500000000000000000000000000005', value: 1 } ]
},
{
  _id: '67520000000000000000000000000004',
  question_id: '67510000000000000000000000000002',
  body: 'MongoDB phù hợp khi dữ liệu mang tính document...',
  created_at: '2024-08-03T15:00:00Z',
  author_info: { author_id: '67500000000000000000000000000004', username: 'Lê Quang Cường' },
  vote_score: 135,
  is_accepted: true,
  voters: [
    { voter_id: '67500000000000000000000000000002', value: 134 },
    { voter_id: '67500000000000000000000000000003', value: 1 }
  ]
},
{
  _id: '67520000000000000000000000000005',
```

```
question_id: '675100000000000000000002',
body: 'Nếu hệ thống của bạn cần báo cáo phức tạp...',
created_at: '2024-08-03T15:30:00Z',
author_info: { author_id: '675000000000000000000002', username: 'Nguyễn Văn
An' },
vote_score: 0,
is_accepted: false,
voters: [
  { voter_id: '675000000000000000000003', value: 1 },
  { voter_id: '675000000000000000000004', value: -1 }
],
{
  _id: '675200000000000000000006',
  question_id: '675100000000000000000004',
  body: 'Với trường tags là mảng...',
  created_at: '2024-08-11T08:00:00Z',
  author_info: { author_id: '675000000000000000000002', username: 'Nguyễn Văn
An' },
  vote_score: 3,
  is_accepted: true,
  voters: [
    { voter_id: '675000000000000000000003', value: 1 },
    { voter_id: '675000000000000000000004', value: 1 },
    { voter_id: '675000000000000000000006', value: 1 }
```

```
]
},
{
  _id: '675200000000000000000007',
  question_id: '675100000000000000000004',
  body: 'Nếu bạn thường lọc theo tag...',
  created_at: '2024-08-11T08:30:00Z',
  author_info: { author_id: '675000000000000000000003', username: 'Trần Thị Bình'
},
  vote_score: 1,
  is_accepted: false,
  voters: [
    { voter_id: '675000000000000000000002', value: 1 },
    { voter_id: '675000000000000000000005', value: 1 },
    { voter_id: '675000000000000000000007', value: -1 }
  ]
},
{
  _id: '675200000000000000000008',
  question_id: '675100000000000000000004',
  body: 'Đừng quên phân tích profiler...',
  created_at: '2024-08-11T09:00:00Z',
  author_info: { author_id: '675000000000000000000005', username: 'Phạm Minh
Đức' },
  vote_score: 1,
```

```
is_accepted: false,

voters: [ { voter_id: '675000000000000000000002', value: 1 } ]

},

{

  _id: '67520000000000000000000009',

  question_id: '67510000000000000000000004',

  body: 'Trong đồ án, bạn có thể minh họa...',

  created_at: '2024-08-11T09:30:00Z',

  author_info: { author_id: '67500000000000000000000007', username: 'Vũ Hoài Phúc'

},

  vote_score: 0,

  is_accepted: false,

  voters: []

},

{

  _id: '6752000000000000000000000a',

  question_id: '67510000000000000000000005',

  body: 'Cách đơn giản là giữ câu hỏi hiện tại...',

  created_at: '2024-08-15T13:00:00Z',

  author_info: { author_id: '67500000000000000000000006', username: 'Hoàng Nam Anh' },

  vote_score: 1,

  is_accepted: false,

  voters: [

    { voter_id: '67500000000000000000000002', value: 1 },
```

```
{ voter_id: '675000000000000000000003', value: 1 },  
  
  { voter_id: '675000000000000000000006', value: -1 }  
  
]  
  
},  
  
{  
  
  _id: '67520000000000000000000b',  
  
  question_id: '675100000000000000000006',  
  
  body: 'Thay vì dùng skip/limit...',  
  
  created_at: '2024-08-20T17:00:00Z',  
  
  author_info: { author_id: '675000000000000000000003', username: 'Trần Thị Bình'  
},  
  
  vote_score: 135,  
  
  is_accepted: true,  
  
  voters: [  
  
    { voter_id: '000000000000000000000000', value: 135 }  
  
  ]  
  
},  
  
{  
  
  _id: '67520000000000000000000c',  
  
  question_id: '675100000000000000000006',  
  
  body: 'Trong MongoDB, bạn có thể kết hợp sort...',  
  
  created_at: '2024-08-20T17:30:00Z',  
  
  author_info: { author_id: '675000000000000000000002', username: 'Nguyễn Văn  
An' },  
  
  vote_score: 3,
```

```
is_accepted: false,

voters: [

  { voter_id: '000000000000000000000000', value: 3 },

  { voter_id: '6750000000000000000000003', value: 1 },

  { voter_id: '6750000000000000000000005', value: -1 }

]

},

{

  _id: '675200000000000000000000d',

  question_id: '6751000000000000000000007',

  body: 'Bạn có thể định nghĩa rule đơn giản...',

  created_at: '2024-08-25T10:00:00Z',

  author_info: { author_id: '6750000000000000000000003', username: 'Trần Thị Bình'

},

  vote_score: 5,

  is_accepted: true,

  voters: [

    { voter_id: '000000000000000000000000', value: 3 },

    { voter_id: '6750000000000000000000005', value: 1 },

    { voter_id: '6750000000000000000000006', value: 1 }

  ]

},

{

  _id: '675200000000000000000000e',

  question_id: '6751000000000000000000007',
```

```
body: 'Đối với câu hỏi, bạn có thể cho upvote +5...',
created_at: '2024-08-25T10:30:00Z',
author_info: { author_id: '675000000000000000000004', username: 'Lê Quang
Cường' },
vote_score: 2,
is_accepted: false,
voters: [
  { voter_id: '000000000000000000000000', value: 3 },
  { voter_id: '67500000000000000000000003', value: -1 },
]
},
{
  _id: '675200000000000000000000f',
  question_id: '67510000000000000000000007',
  body: 'Quan trọng là bạn mô tả rõ rule...',
  created_at: '2024-08-25T11:00:00Z',
  author_info: { author_id: '675000000000000000000006', username: 'Hoàng Nam
Anh' },
  vote_score: 4,
  is_accepted: false,
  voters: [
    { voter_id: '67500000000000000000000003', value: 1 },
    { voter_id: '000000000000000000000000', value: 3 }
  ]
},
```

Thiết kế và cài đặt cơ sở dữ liệu NoSQL cho hệ thống diễn đàn Hỏi-Đáp chuyên ngành với tính năng đánh giá và vinh danh.

```
{
  _id: '67520000000000000000000010',
  question_id: '67510000000000000000000009',
  body: 'Bạn nên tạo collection reports riêng...',
  created_at: '2024-09-05T15:00:00Z',
  author_info: { author_id: '67500000000000000000000004', username: 'Lê Quang Cường' },
  vote_score: 4,
  is_accepted: true,
  voters: [
    { voter_id: '00000000000000000000000000', value: 3 },
    { voter_id: '6750000000000000000000000005', value: 1 }
  ]
},
{
  _id: '6752000000000000000000000011',
  question_id: '675100000000000000000000000a',
  body: 'Đề hệ thống chuyên nghiệp...',
  created_at: '2024-09-10T09:00:00Z',
  author_info: { author_id: '67500000000000000000000005', username: 'Phạm Minh Đức' },
  vote_score: 4,
  is_accepted: true,
  voters: [
    { voter_id: '00000000000000000000000000', value: 3 },
    { voter_id: '6750000000000000000000000005', value: 1 }
  ]
}
```



```
{ voter_id: '675000000000000000000003', value: 1 },  
  
],  
  
,  
  
{  
  
  _id: '6752000000000000000000012',  
  
  question_id: '67510000000000000000000a',  
  
  body: 'Bạn có thể thêm một bước hướng dẫn...',  
  
  created_at: '2024-09-10T09:20:00Z',  
  
  author_info: { author_id: '6750000000000000000000002', username: 'Nguyễn Văn  
An' },  
  
  vote_score: 5,  
  
  is_accepted: false,  
  
  voters: [  
  
    { voter_id: '0000000000000000000000000', value: 3 },  
  
    { voter_id: '6750000000000000000000003', value: 1 },  
  
    { voter_id: '6750000000000000000000006', value: 1 },  
  
  ],  
  
},  
  
{  
  
  _id: '6752000000000000000000013',  
  
  question_id: '67510000000000000000000a',  
  
  body: 'Trong đề án, bạn có thể minh họa...',  
  
  created_at: '2024-09-10T09:40:00Z',  
  
  author_info: { author_id: '6750000000000000000000007', username: 'Vũ Hoài Phúc'  
},
```

```
    vote_score: 3,

    is_accepted: false,

    voters: [

      { voter_id: '000000000000000000000000', value: 3 },

      { voter_id: '6750000000000000000000005', value: 1 },

      { voter_id: '6750000000000000000000009', value: -1 }

    ]

  },

  {

    _id: '67520000000000000000000014',

    question_id: '675100000000000000000000a',

    body: 'Tag có thể lưu dạng slug không dấu...',

    created_at: '2024-09-10T10:00:00Z',

    author_info: { author_id: '6750000000000000000000008', username: 'Linh Chi' },

    vote_score: 3,

    is_accepted: false,

    voters: [

      { voter_id: '000000000000000000000000', value: 3 }

    ]

  },

  {

    _id: '67520000000000000000000015',

    question_id: '675100000000000000000000b',

    body: 'Bạn có thể bật text index...',

    created_at: '2024-09-15T10:00:00Z',
```

```
author_info: { author_id: '675000000000000000000006', username: 'Hoàng Nam Anh' },
```

```
vote_score: 4,
```

```
is_accepted: true,
```

```
voters: [
```

```
  { voter_id: '000000000000000000000000', value: 3 },
```

```
  { voter_id: '675000000000000000000003', value: 1 },
```

```
]
```

```
},
```

```
{
```

```
  _id: '67520000000000000000000016',
```

```
  question_id: '675100000000000000000000b',
```

```
  body: 'Đề highlight từ khóa trên giao diện...',
```

```
  created_at: '2024-09-15T10:30:00Z',
```

```
  author_info: { author_id: '675000000000000000000002', username: 'Nguyễn Văn An' },
```

```
  vote_score: 4,
```

```
  is_accepted: false,
```

```
  voters: [
```

```
    { voter_id: '000000000000000000000000', value: 3 },
```

```
    { voter_id: '675000000000000000000003', value: 1 },
```

```
    { voter_id: '675000000000000000000005', value: 1 },
```

```
    { voter_id: '675000000000000000000008', value: -1 }
```

```
  ]
```

```
},
```



```
voters: [
  { voter_id: '000000000000000000000000', value: 3 },
  { voter_id: '6750000000000000000000003', value: 1 },
  { voter_id: '6750000000000000000000005', value: 1 }
]
},
{
  _id: '6752000000000000000000001b',
  question_id: '675100000000000000000000e',
  body: 'Trên giao diện, bạn có thể hiển thị nhãn...',
  created_at: '2024-10-01T09:30:00Z',
  author_info: { author_id: '6750000000000000000000009', username: 'Tuấn Anh' },
  vote_score: 3,
  is_accepted: false,
  voters: [
    { voter_id: '000000000000000000000000', value: 3 },
    { voter_id: '6750000000000000000000003', value: -1 },
    { voter_id: '675000000000000000000000a', value: 1 }
  ]
},
{
  _id: '6752000000000000000000001c',
  question_id: '675100000000000000000000f',
  body: 'Một bộ demo hợp lý là khoảng 10 users...',
  created_at: '2024-10-05T11:00:00Z',
```

```
author_info: { author_id: '675000000000000000000000a', username: 'Khoái' },  
vote_score: 4,  
is_accepted: false,  
voters: [  
  { voter_id: '0000000000000000000000000000', value: 3 },  
  { voter_id: '6750000000000000000000000003', value: 1 },  
]]
```

4.1.5 Các trường hợp được tạo trong dữ liệu mẫu

Bộ dữ liệu mẫu được xây dựng gồm:

- Collection Users: 10 tài liệu (documents);
- Collection Questions: 15 tài liệu;
- Collection Answers: 28 tài liệu;

Chúng không chỉ nhằm mục đích hiển thị mà còn chứa các kịch bản kiểm thử để đánh giá độ chính xác của các truy vấn nghiệp vụ.

Về xử lý xếp hạng đồng hạng: Bộ dữ liệu mẫu có thiết lập một tình huống tranh chấp vị trí dẫn đầu giữa các thành viên. Cụ thể, hai người dùng Nguyễn Văn An và Trần Thị Bình được gán cùng mức điểm uy tín là 115, đồng thời hai câu trả lời của Lê Quang Cường và Trần Thị Bình cũng sở hữu số lượt bình chọn cao nhất bằng nhau là 135.

Về việc lọc theo loại tài khoản: một trường hợp về tài khoản vi phạm đã được đưa vào dữ liệu mẫu. Người dùng Tuấn Anh được thiết lập mức điểm uy tín kỷ lục là 500 điểm – cao nhất toàn hệ thống – nhưng lại mang trạng thái là "banned" (bị khóa). Khi thực thi truy vấn trích xuất "Top 5 thành viên uy tín", hệ thống đã loại bỏ hoàn toàn tài khoản này khỏi danh sách kết quả nhờ bộ lọc trạng thái (\$match), nhường vị trí dẫn đầu cho các thành viên hợp lệ có điểm số thấp hơn.

Về các trường dữ liệu ảo (giả lập hệ thống đã vận hành lâu năm): Dữ liệu mẫu có tạo "Vote gộp" để kiểm thử hiệu năng xử lý số liệu lớn. Tại các câu trả lời có mã định

Thiết kế và cài đặt cơ sở dữ liệu NoSQL cho hệ thống diễn đàn Hỏi-Đáp chuyên ngành với tính năng đánh giá và vinh danh.

danh kết thúc bằng ...004 và ...00b, hệ thống ghi nhận một lượt bình chọn đặc biệt từ ID hệ thống (000...000) với giá trị lần lượt là 3 điểm 134 điểm. Hệ thống vẫn tính toán và hiển thị chính xác tổng điểm bình chọn mà không cần phải truy vấn và lưu trữ hàng trăm document người dùng ảo.

Về đảm bảo tính nhất quán dữ liệu: Tại câu trả lời mã ...001, trường điểm số hiển thị (vote_score) có giá trị là 15, hoàn toàn trùng khớp với tổng giá trị của các phần tử trong mảng người bình chọn (12 + 1 + 1 + 1). Ngoài ra các dữ liệu khác đều được tạo theo nguyên tắc này.

4.2 Kết quả và thực hiện các chức năng Quản lý (CRUD)

Phần này trình bày việc hiện thực hóa các chức năng cơ bản của diễn đàn: Đăng bài, Xem bài và Quản lý nội dung.

4.2.1 Đăng câu hỏi mới (UC03)

Để đảm bảo dữ liệu bài viết mới được lưu trữ đồng thời với việc ghi nhận thành tích cho thành viên, hệ thống thực hiện chuỗi thao tác liên tục sau:

- Bước 1: Thực hiện lệnh insertOne để khởi tạo Document câu hỏi mới vào collection questions với trạng thái mặc định là "open":

```
db.questions.insertOne({
  _id: "675100000000000000000099",
  title: "Demo MongoDB Indexing",
  body: "Làm sao tối ưu truy vấn?",
  tags: ["mongodb", "performance"],
  created_at: new Date(),
  status: "open",
  author_info: {
    author_id: "675000000000000000000002",
    username: "Nguyễn Văn An"
  },
  vote_score: 0,
  answer_count: 0,
  voters: []
})
```


Thiết kế và cài đặt cơ sở dữ liệu NoSQL cho hệ thống diễn đàn Hỏi-Đáp chuyên ngành với tính năng đánh giá và vinh danh.

- Bước 2: Thực thi lệnh `updateOne` với toán tử `$inc` vào collection `users` để tăng số lượng câu hỏi (`question_count`) của tác giả lên 1 đơn vị.

```
db.users.updateOne(
  { _id: "67500000000000000000000000000002"},
  { $inc: { "stats.question_count": 1 } }
)
```

Kết quả sau khi thực hiện: câu hỏi mới được chèn, số câu hỏi trong `users` đã tăng từ 5 lên 6.

```
//1. hệ thống thông báo chèn thành công
{
  acknowledged: true,
  insertedId: '6751000000000000000000000000099'
}

//2. Kiểm tra lại user Nguyễn Văn An
{
  _id: '67500000000000000000000000000002',
  username: 'Nguyễn Văn An',
  ...
  stats: {
    question_count: 6, //số câu hỏi tăng từ 5 lên 6
    answer_count: 6,
    accepted_answer_count: 3
  },
  ...
}
```

4.2.2 Đăng câu trả lời (UC04)

Nghệ vụ này yêu cầu sự đồng bộ dữ liệu cao giữa 3 thực thể (Users, Questions, Answers). Hệ thống xử lý theo các bước sau:

- Bước 1: Thêm nội dung câu trả lời vào collection `answers`, liên kết với câu hỏi gốc thông qua khóa ngoại `question_id`:

Thiết kế và cài đặt cơ sở dữ liệu NoSQL cho hệ thống diễn đàn Hỏi-Đáp chuyên ngành với tính năng đánh giá và vinh danh.

```
//2. Kiểm tra lại user (Trần Thị Bình)
{
  _id: '675000000000000000000003',
  username: 'Trần Thị Bình',
  ...
  stats: {
    question_count: 5,
    answer_count: 7, //Số câu trả lời tăng từ 6 lên 7
    accepted_answer_count: 3
  },
  ...
}

//Kiểm tra lại câu hỏi (câu hỏi có ID ...099 vừa tạo ở mục 4.2.1)
{
  _id: '675100000000000000000099',
  ...
  answer_count: 1, //Đã tăng từ 0 lên 1
  ...
}
```

4.2.3 Xem chi tiết bài đăng (UC01)

Khi người dùng truy cập vào một bài viết cụ thể, hệ thống cần trích xuất dữ liệu từ hai nguồn để hiển thị trọn vẹn nội dung thảo luận:

- Bước 1: Truy vấn collection questions theo _id để lấy nội dung câu hỏi chính.

```
db.questions.findOne({ _id: "675100000000000000000001" })
```

- Bước 2: Truy vấn collection answers dựa trên khóa ngoại question_id để lấy toàn bộ danh sách các câu trả lời liên quan.

```
db.answers.find({ question_id: "675100000000000000000001" })
```

Kết quả trả về : hiện thông tin đầy đủ câu hỏi có _id và câu trả lời có mã câu hỏi như truy vấn:

```
//1. Kết quả câu hỏi cần tìm
{
  _id: '67510000000000000000000000000001',
  title: 'Thiết kế schema MongoDB cho hệ thống hỏi đáp giống StackOverflow?',
  body: 'Em đang làm đồ án hệ thống hỏi đáp chuyên ngành, cần thiết kế schema MongoDB cho users, questions, answers. Nên nhúng (embed) hay tham chiếu (reference) để tối ưu cho việc truy vấn danh sách câu hỏi và câu trả lời?',
  ...
}

//2. Kết quả danh sách câu trả lời
{
  _id: '67520000000000000000000000000001',
  question_id: '67510000000000000000000000000001',
  body: 'Bạn có thể thiết kế schema dạng lại...',
  ...
}
{
  _id: '67520000000000000000000000000002',
  question_id: '67510000000000000000000000000001',
  body: 'Nếu quy mô đồ án nhỏ...',
  ...
}
{
  _id: '67520000000000000000000000000003',
  question_id: '67510000000000000000000000000001',
  body: 'Trong đồ án Cơ sở ngành...',
  ...
}
```

4.2.4 Sửa bài đăng (UC08)

Chức năng này cho phép tác giả cập nhật nội dung bài viết, đồng thời đảm bảo tính sở hữu của tác giả bài đăng.

- Bước 1: Thực hiện lệnh updateOne để tìm kiếm bài viết theo _id, kết hợp điều kiện kiểm tra author_info.author_id để cho chỉ chính tác giả mới có quyền chỉnh sửa.

Thiết kế và cài đặt cơ sở dữ liệu NoSQL cho hệ thống diễn đàn Hỏi-Đáp chuyên ngành với tính năng đánh giá và vinh danh.

- Bước 2: Sử dụng toán tử \$set để cập nhật các trường nội dung thay đổi và ghi nhận thời gian chỉnh sửa (updated_at).

Câu truy vấn cụ thể như sau:

```
db.questions.updateOne(
  {
    _id: "675100000000000000000001",
    "author_info.author_id": "675000000000000000000002"
  },
  {
    $set: {
      title: "[Đã giải quyết] Thiết kế schema MongoDB?",
      updated_at: new Date()
    }
  }
)
```

Kết quả trả về: trả về một document thông báo đã cập nhật thành công

```
//1. Hệ thống thông báo cập nhật thành công
{
  acknowledged: true,
  insertedId: null,
  matchedCount: 1,
  modifiedCount: 1,
  upsertedCount: 0
}
//2. Kiểm tra lại câu hỏi
{
  _id: '675100000000000000000001',
  title: '[Đã giải quyết] Thiết kế schema MongoDB?', //Cập nhật lại tiêu đề
  ...
  updated_at: 2026-01-03T09:04:48.465Z //Thêm thời gian cập nhật lại câu hỏi
}
```


4.3.1 Cơ chế bình chọn đồng thời

Khi một hành động đánh giá diễn ra, hệ thống thực hiện cập nhật song song trên hai collection: Tại Collection questions/answers thực hiện việc tăng/giảm điểm vote_score và thêm ID người vote vào mảng voters để ngăn chặn việc vote trùng lặp. Tại Collection người dùng users thực hiện việc cộng/trừ điểm uy tín (reputation) cho tác giả bài viết để ghi nhận đóng góp. Các thao tác như sau:

4.3.2 Đánh giá câu hỏi (UC05)

Khi người dùng thực hiện hành động Vote, hệ thống sẽ cập nhật song song trên hai collection để đảm bảo tính công bằng:

- Bước 1: Tại collection questions, hệ thống sử dụng toán tử \$inc để tăng điểm vote_score và \$push để thêm ID người vote vào mảng voters. Việc này giúp ngăn chặn hành vi spam vote (một người vote nhiều lần).

```
db.questions.updateOne(
  { _id: "675100000000000000000002" },
  {
    $inc: { vote_score: 1 },
    $push: { voters: { voter_id: "675000000000000000000003", value: 1 } }
  }
)
```

- Bước 2: Tại collection users, tiến hành cộng điểm thưởng vào trường reputation của tác giả câu hỏi nhằm khích lệ đóng góp chất lượng.

```
db.users.updateOne(
  { _id: "675000000000000000000003" },
  { $inc: { reputation: 10 } }
)
```

Kết quả : cập nhật được lượt vote đồng thời cộng điểm cho tác giả câu hỏi.

```
//1. Ghi nhận lượt vote mới trong mảng voters
{
  _id: '675100000000000000000002',
  ...
}
```

Thiết kế và cài đặt cơ sở dữ liệu NoSQL cho hệ thống diễn đàn Hỏi-Đáp chuyên ngành với tính năng đánh giá và vinh danh.

```
voters: [  
  ...  
  {voter_id: '675000000000000000000002', value: 1}  
]  
}  
  
//2. Kiểm tra điểm tác giả  
{  
  _id: '675000000000000000000003', //tác giả có câu hỏi được vote  
  ...  
  reputation: 125, //điểm uy tín cộng 10 điểm, từ 115 lên 125  
  ...  
}
```

4.3.3 Đánh giá câu trả lời (UC06)

Quy trình thực hiện tương tự như đánh giá câu hỏi, nghiệp vụ này xử lý sự tương tác với các giải pháp được đưa ra (các câu trả lời):

- Bước 1: Cập nhật tăng điểm vote_score và lưu vết người đánh giá vào document câu trả lời tương ứng trong collection answers.

```
db.answers.updateOne(  
  { _id: "675200000000000000000001" },  
  {  
    $inc: { vote_score: 1 },  
    $push: { voters: { voter_id: "675000000000000000000002", value: 1 } }  
  }  
)
```

- Bước 2: Cộng điểm uy tín (reputation) cho thành viên đã đưa ra câu trả lời đó. Mức điểm thưởng ở đây (10 điểm) được thiết lập cao hơn so với câu hỏi để khuyến khích việc giải quyết vấn đề.

```
db.users.updateOne(  
  { _id: "675000000000000000000003" },  
  { $inc: { reputation: 10 } }  
)
```


Kết quả: cập nhật được lượt vote đồng thời tác giả câu trả lời nhận được điểm.

```
//1. Ghi nhận lượt vote trong mảng voters
{
  _id: '67520000000000000000000000000001',
  ...
  voters: [
    ...
    {
      voter_id: '67500000000000000000000000000002',
      value: 1
    }
  ]
}

//2. Kiểm tra tác giả
{
  _id: '67500000000000000000000000000003', //tác giả có câu trả lời được vote
  ...
  reputation: 135, //cộng 10 điểm, từ 125 (điểm sau khi cập nhật ở 4.3.2) lên 135
  ...
}
```

4.3.4 Chấp nhận câu trả lời (UC07)

Người hỏi xác nhận giải pháp đúng nhất. Hệ thống đánh dấu câu trả lời và thưởng điểm lớn (15 điểm) cho người trả lời. Các bước thực hiện như sau:

- Bước 1: Cập nhật trạng thái câu trả lời được chấp nhận chưa:

```
db.answers.updateOne(
  { _id: "67520000000000000000000000000001" },
  { $set: { is_accepted: true } }
)
```

- Bước 2: Cập nhật điểm cho user:

```
db.users.updateOne(
  { _id: "67500000000000000000000000000003" },
  {
    $inc: {
```

Thiết kế và cài đặt cơ sở dữ liệu NoSQL cho hệ thống diễn đàn Hỏi-Đáp chuyên ngành với tính năng đánh giá và vinh danh.

```
        reputation: 15,
        "stats.accepted_answer_count": 1
    }
}
)
```

Kết quả sau khi truy vấn: đã chuyển trạng thái `is_accepted` từ `false` thành `true`, đồng thời cộng điểm cho người trả lời.

```
//1. Kiểm tra trạng thái
{
  _id: '67520000000000000000000000000001',
  ...
  is_accepted: true, //Câu trả lời được chấp nhận
  ...
}
{
  _id: '67500000000000000000000000000003', //tác giả có câu trả lời được chấp nhận
  ...
  reputation: 150, //cộng 15 điểm, từ 135 (điểm sau khi cập nhật ở 4.3.3) lên 150
  ...
}
```

4.4 Kết quả và thực hiện nghiệp vụ Vinh danh

Nghiệp vụ này sử dụng Aggregation Framework (Pipeline xử lý dữ liệu) để thực hiện các thống kê phức tạp, xếp hạng thành viên và tự động trao huy hiệu dựa trên dữ liệu lịch sử hoạt động.

4.4.1 Top 5 thành viên uy tín nhất

Hệ thống lấy danh sách 5 thành viên có trạng thái hoạt động (active) với điểm uy tín (reputation) cao nhất. Trong trường hợp bằng điểm, ưu tiên thành viên tham gia hệ thống sớm hơn (joined_at). Truy vấn được thực hiện như sau:

```
db.users.aggregate([
  { $match: { status: "active" } },
  { $sort: { reputation: -1, joined_at: 1 } }, // Ưu tiên điểm cao, sau đó đến ngày gia nhập
  { $limit: 5 },
])
```

Thiết kế và cài đặt cơ sở dữ liệu NoSQL cho hệ thống diễn đàn Hỏi-Đáp chuyên ngành với tính năng đánh giá và vinh danh.

```
{
  $project: {
    _id: 0,
    username: 1,
    reputation: 1,
    medal_counts: 1
  }
}
```

Kết quả trả về: trả về danh sách top 5 thành viên có điểm uy tín cao nhất dựa theo điểm uy tín và ngày gia nhập:

```
{
  username: 'Nguyễn Văn An',
  reputation: 115,
  medal_counts: {
    gold: 3,
    platinum: 1,
    diamond: 0
  }
}
{
  username: 'Trần Thị Bình',
  reputation: 115,
  medal_counts: {
    gold: 3,
    platinum: 2,
    diamond: 1
  }
}
{
  username: 'Lê Quang Cường',
  reputation: 82,
  medal_counts: {
    gold: 2,
    platinum: 1,
```

Thiết kế và cài đặt cơ sở dữ liệu NoSQL cho hệ thống diễn đàn Hỏi-Đáp chuyên ngành với tính năng đánh giá và vinh danh.

```
    diamond: 0
  }
}
{
  username: 'Hoàng Nam Anh',
  reputation: 62,
  medal_counts: {
    gold: 1,
    platinum: 1,
    diamond: 0
  }
}
{
  username: 'Phạm Minh Đức',
  reputation: 59,
  medal_counts: {
    gold: 1,
    platinum: 0,
    diamond: 0
  }
}
```

4.4.2 Vinh danh thành viên giải quyết vấn đề xuất sắc nhất

Nhiệm vụ đặt được là vinh danh thành viên có nhiều câu trả lời được chấp nhận (accepted_answer_count) nhất. Nếu có nhiều người cùng đạt số lượng cao nhất, hiển thị tất cả. Truy vấn được thực hiện như sau:

```
db.users.aggregate([
  { $match: { status: "active" } },
  {
    $group: {
      _id: "$stats.accepted_answer_count",
      candidates: { $push: { username: "$username", count: "$stats.accepted_answer_count" } }
    }
  },
  { $sort: { _id: -1 } }, // Sắp xếp theo số lượng giảm dần
  { $limit: 1 },        // Lấy nhóm cao nhất
])
```

Thiết kế và cài đặt cơ sở dữ liệu NoSQL cho hệ thống diễn đàn Hỏi-Đáp chuyên ngành với tính năng đánh giá và vinh danh.

```
{ $unwind: "$candidates" },
{
  $project: {
    _id: 0,
    username: "$candidates.username",
    accepted_count: "$candidates.count"
  }
}
]);
```

Kết quả trả về: Trả về thành viên có số câu trả lời được chấp nhận nhiều nhất (trường hợp ở đây là 2 người đồng điểm)

```
{ "username": "Nguyễn Văn An", "accepted_count": 3 }
{ "username": "Trần Thị Bình", "accepted_count": 3 }
```

4.4.3 Vinh danh thành viên hoạt động chăm chỉ nhất (Nhiều câu trả lời)

Hệ thống vinh danh thành viên đóng góp nhiều câu trả lời (answer_count) nhất. Hiện thị tất cả nếu đồng hạng. Truy vấn được thực hiện như sau:

```
db.users.aggregate([
  { $match: { status: "active" } },
  {
    $group: {
      _id: "$stats.answer_count",
      candidates: { $push: { username: "$username", count: "$stats.answer_count" } }
    }
  },
  { $sort: { _id: -1 } }, { $limit: 1 }, { $unwind: "$candidates" },
  {
    $project: {
      _id: 0,
      username: "$candidates.username",
      answer_count: "$candidates.count"
    }
  }
]);
```

Thiết kế và cài đặt cơ sở dữ liệu NoSQL cho hệ thống diễn đàn Hỏi-Đáp chuyên ngành với tính năng đánh giá và vinh danh.

Kết quả trả về: Trả về thành viên có số câu trả lời nhiều nhất (trường hợp ở đây là 2 người đồng điểm)

```
{ "username": "Nguyễn Văn An", "answer_count": 6 }  
{ "username": "Trần Thị Bình", "answer_count": 6 }
```

4.4.4 Vinh danh thành viên đặt vấn đề nhiều nhất (Nhiều câu hỏi)

Hệ thống thực hiện vinh danh thành viên đặt nhiều câu hỏi (question_count) nhất hệ thống. Cú pháp như sau:

```
db.users.aggregate([  
  { $match: { status: "active" } },  
  {  
    $group: {  
      _id: "$stats.question_count",  
      candidates: { $push: { username: "$username", count: "$stats.question_count" } }  
    }  
  },  
  { $sort: { _id: -1 } },  
  { $limit: 1 },  
  { $unwind: "$candidates" },  
  {  
    $project: {  
      _id: 0,  
      username: "$candidates.username",  
      question_count: "$candidates.count"  
    }  
  }  
]);
```

Kết quả trả về: Kết quả trả về thành viên đặt nhiều câu hỏi nhất (Trường hợp ở đây là hai người đồng điểm)

```
{ "username": "Nguyễn Văn An", "question_count": 5 }  
{ "username": "Trần Thị Bình", "question_count": 5 }
```

4.4.5 Vinh danh theo huy chương

Nhiệm vụ là vinh danh thành viên sở hữu số lượng theo loại huy chương nhiều nhất (Trong trường hợp này là vinh danh theo medal_counts.gold). Cú pháp như sau:

```
db.users.aggregate([
  { $match: { status: "active" } },
  {
    $group: {
      _id: "$medal_counts.gold",
      candidates: { $push: { username: "$username", gold_medals: "$medal_counts.gold" } }
    }
  },
  { $sort: { _id: -1 } },
  { $limit: 1 },
  { $unwind: "$candidates" },
  {
    $project: {
      _id: 0,
      username: "$candidates.username",
      gold_medals: "$candidates.gold_medals"
    }
  }
]);
```

Kết quả trả về: Kết quả trả về thành viên sở hữu số huy chương vàng (Trường hợp ở đây là hai người đồng điểm)

```
{ "username": "Nguyễn Văn An", "gold_medals": 3 },
{ "username": "Trần Thị Bình", "gold_medals": 3 }
```

4.4.6 Câu hỏi được thảo luận sôi nổi nhất

Trường hợp này sẽ tìm các câu hỏi đang mở (status: open) có số lượng câu trả lời (answer_count) nhiều nhất. Cú pháp như sau:

Thiết kế và cài đặt cơ sở dữ liệu NoSQL cho hệ thống diễn đàn Hỏi-Đáp chuyên ngành với tính năng đánh giá và vinh danh.

```
db.questions.aggregate([
  { $match: { status: "open" } },
  {
    $group: {
      _id: "$answer_count",
      questions: { $push: { title: "$title", author: "$author_info.username", answers:
"$answer_count" } }
    }
  },
  { $sort: { _id: -1 } },
  { $limit: 1 },
  { $unwind: "$questions" },
  {
    $project: {
      _id: 0,
      title: "$questions.title",
      author: "$questions.author",
      answers: "$questions.answers"
    }
  }
]);
```

Kết quả trả về: Trả về câu hỏi ở trạng thái open có số câu trả lời nhiều nhất (Trong trường hợp này có 2 câu hỏi có số câu trả lời như nhau)

```
{
  "title": "Thiết kế index cho collection questions để tối ưu tìm kiếm theo tag?",
  "author": "Nguyễn Văn An",
  "answers": 4
},
{
  "title": "Cách tổ chức tag cho câu hỏi: dùng tiếng Việt hay tiếng Anh?",
  "author": "Phạm Minh Đức",
  "answers": 4
}
```


4.4.7 Câu trả lời chất lượng nhất

Yêu cầu đặt được là tìm câu trả lời có điểm bình chọn (vote_score) cao nhất trong toàn bộ hệ thống. Cú pháp như sau:

```
db.answers.aggregate([
  {
    $group: {
      _id: "$vote_score",
      answers: { $push: { author: "$author_info.username", vote_score: "$vote_score", body: "$body" } }
    },
  },
  { $sort: { _id: -1 } }, { $limit: 1 },
  { $unwind: "$answers" },
  {
    $project: {
      _id: 0,
      author: "$answers.author",
      vote_score: "$answers.vote_score",
      body: "$answers.body" // Hiển thị body để kiểm chứng
    }
  }
]);
```

Kết quả trả về: Trả về câu trả lời có điểm bình chọn cao nhất (Trường hợp ở đây có 2 câu trả lời đồng điểm)

```
{
  "author": "Lê Quang Cường",
  "vote_score": 135,
  "body": "MongoDB phù hợp khi dữ liệu mang tính document..."
},
{
  "author": "Trần Thị Bình",
  "vote_score": 135,
  "body": "Thay vì dùng skip/limit..."
}
```

4.4.8 Chuyên gia lĩnh vực MongoDB

Nhiệm vụ là tìm thành viên có tổng điểm vote cao nhất từ các câu hỏi thuộc chủ đề (tag) "mongodb". Cú pháp như sau:

```
db.questions.aggregate([
  { $match: { tags: "mongodb" } },
  {
    $group: {
      _id: "$author_info.username",
      total_votes: { $sum: "$vote_score" }
    }
  },
  {
    $group: {
      _id: "$total_votes",
      users: { $push: { username: "$_id", total_votes: "$total_votes" } }
    }
  },
  { $sort: { _id: -1 } },
  { $limit: 1 },
  { $unwind: "$users" },
  {
    $project: {
      _id: 0,
      username: "$users.username",
      total_votes: "$users.total_votes"
    }
  }
]);
```

Kết quả trả về: Trả về user có điểm vote cao nhất theo tag "mongodb".

```
{ "username": "Nguyễn Văn An", "total_votes": 7 }
```

//Giải thích: An có 3 câu hỏi tag mongodb với điểm lần lượt là 2, 3, 2. Tổng = 7).

4.4.9 Phân loại danh hiệu người dùng

Nhiệm vụ là Phân loại danh hiệu cho Top 3 thành viên active dựa trên điểm uy tín. Danh hiệu được phân loại như sau:

- Điểm uy tín > 300: Bạc thầy
- Điểm uy tín > 100: Người hướng dẫn
- Điểm uy tín > 50: Học việc
- Còn lại: Thành viên mới

Cú pháp thực hiện như sau:

```
db.users.aggregate([
  { $match: { status: "active" } },
  { $sort: { reputation: -1 } },
  {
    $project: {
      _id: 0,
      username: 1,
      reputation: 1,
      danh_hieu: {
        $switch: {
          branches: [
            { case: { $gte: ["$reputation", 300] }, then: "Bạc thầy" },
            { case: { $gte: ["$reputation", 100] }, then: "Người hướng dẫn" },
            { case: { $gte: ["$reputation", 50] }, then: "Học việc" }
          ],
          default: "Thành viên mới"
        }
      }
    }
  }
]);
```

Kết quả trả về : trả về danh sách user có kèm theo danh hiệu.

```
{ "username": "Trần Thị Bình", "reputation": 150, "danh_hieu": "Người hướng dẫn" }
{ "username": "Nguyễn Văn An", "reputation": 115, "danh_hieu": "Người hướng dẫn" }
{ "username": "Lê Quang Cường", "reputation": 82, "danh_hieu": "Học việc" }
{ "username": "Hoàng Nam Anh", "reputation": 62, "danh_hieu": "Học việc" }
{ "username": "Phạm Minh Đức", "reputation": 59, "danh_hieu": "Học việc" }
{ "username": "Vũ Hoài Phúc", "reputation": 29, "danh_hieu": "Thành viên mới" }
{ "username": "Linh Chi", "reputation": 8, "danh_hieu": "Thành viên mới" }
```

Mặc dù thành viên Tuấn Anh sở hữu 500 điểm uy tín và đạt được danh hiệu bậc thầy, nhưng hệ thống không hiển thị trong danh sách vinh danh. Lý do là vì câu truy vấn đã áp dụng bộ lọc { \$match: { status: "active" } } để loại bỏ các tài khoản bị khóa (banned) do vi phạm quy tắc cộng đồng.

CHƯƠNG 5: KẾT LUẬN VÀ HƯỚNG PHÁT TRIỂN

Sau quá trình nghiên cứu và thực hiện, đồ án "Thiết kế và cài đặt cơ sở dữ liệu NoSQL cho hệ thống diễn đàn Hỏi-Đáp chuyên ngành" đã thực hiện được các vấn đề sau:

5.1 Về mặt lý thuyết

Đề tài đã phân tích và so sánh được sự khác biệt cơ bản giữa mô hình quan hệ (RDBMS) và phi quan hệ (NoSQL), từ đó làm rõ lý do lựa chọn MongoDB cho các hệ thống yêu cầu hiệu năng đọc cao và cấu trúc dữ liệu linh hoạt, giúp nắm vững các kỹ thuật thiết kế lược đồ (Schema Design) đặc thù của NoSQL như nhúng (Embedding) để tối ưu hóa việc đọc và tham chiếu (Referencing) để đảm bảo tính nhất quán cho dữ liệu lớn.

5.2 Về mặt thực tiễn và kỹ thuật:

Đồ án đã thiết kế thành công lược đồ dữ liệu với cấu trúc lưu trữ tối ưu cho các đối tượng Users, Questions và Answers, giải quyết được bài toán hiệu năng thông qua kỹ thuật nhúng thông tin tác giả và danh sách người bình chọn ngay trong document. Trên cơ sở cấu trúc đó, đồ án đã ứng dụng các toán tử cập nhật nguyên tử (Atomic Updates), đảm bảo tính chính xác của dữ liệu ngay cả khi có lượng lớn tương tác đánh giá cùng lúc. Đặc biệt, hệ thống vinh danh tự động đã được hiện thực hóa thông qua việc khai thác sức mạnh của Aggregation Framework để xây dựng các pipeline xử lý phức tạp, cho phép tự động tính toán xếp hạng và trao huy chương mà không cần can thiệp thủ công. Nhìn chung, các kết quả đạt được đã chứng minh tính khả thi và hiệu quả vượt trội của việc ứng dụng công nghệ NoSQL trong xây dựng các hệ thống mạng xã hội hoặc diễn đàn trao đổi trực tuyến hiện đại.

5.3 Đánh giá ưu điểm và nhược điểm

Về ưu điểm:

- Hiệu năng đọc vượt trội: Nhờ áp dụng chiến lược Nhúng (Embedding) thông tin tác giả và danh sách người vote vào trong document bài viết, hệ thống giảm thiểu tối đa các thao tác nối bảng (JOIN). Điều này giúp tốc độ tải trang chi tiết bài đăng nhanh hơn đáng kể so với mô hình quan hệ truyền thống.

Thiết kế và cài đặt cơ sở dữ liệu NoSQL cho hệ thống diễn đàn Hỏi-Đáp chuyên ngành với tính năng đánh giá và vinh danh.

- Đảm bảo tính toàn vẹn dữ liệu: Việc sử dụng các toán tử nguyên tử (Atomic Operators) như \$inc, \$push cho phép hệ thống xử lý chính xác các thao tác Đánh giá (Vote) ngay cả khi có nhiều người dùng tương tác cùng lúc mà không cần cơ chế khóa (Locking) phức tạp.

- Linh hoạt trong mở rộng cấu trúc: Lược đồ dữ liệu của MongoDB cho phép dễ dàng thêm các trường mới (như huy chương, thống kê) mà không làm gián đoạn hệ thống hay phải chạy các lệnh ALTER TABLE nặng nề như trong SQL.

- Khả năng xử lý thống kê mạnh mẽ: Aggregation Framework đã chứng minh được sức mạnh trong việc xử lý các logic nghiệp vụ phức tạp (như xếp hạng, cấp huy chương tự động) ngay tại phía Database, giảm tải cho phía Server ứng dụng.

Về nhược điểm, hạn chế:

- Dư thừa dữ liệu (Data Redundancy): Do áp dụng kỹ thuật phi chuẩn hóa (Denormalization), thông tin người dùng (tên, avatar) bị lặp lại ở nhiều nơi (trong collection questions, answers). Điều này dẫn đến khó khăn khi cần cập nhật thông tin (ví dụ: khi user đổi tên, hệ thống phải cập nhật hàng loạt document liên quan).

- Độ phức tạp của câu truy vấn: Các câu lệnh Aggregation (để thống kê huy chương, xếp hạng) có cú pháp phức tạp và khó bảo trì hơn so với câu lệnh SQL thông thường. Việc viết và tối ưu hóa các pipeline này đòi hỏi kiến thức chuyên sâu.

- Giới hạn kích thước Document: Mặc dù chưa chạm ngưỡng trong đề án này, nhưng việc nhúng mảng voters vào trong bài viết có nguy cơ vượt quá giới hạn 16MB của MongoDB nếu số lượng người vote quá lớn (hàng triệu lượt).

- Chưa có giao diện người dùng: Đề án hiện tại mới chỉ dừng lại ở mức độ xử lý dữ liệu (Backend/Database), chưa xây dựng được giao diện (Frontend) để người dùng cuối trải nghiệm trực quan.

5.4 Hướng phát triển

Do giới hạn về mặt thời gian và phạm vi của đề án cơ sở ngành, hệ thống hiện tại vẫn còn một số hạn chế. Để hoàn thiện và đưa sản phẩm vào ứng dụng thực tế, các hướng phát triển tiếp theo được đề xuất như sau:

- Phát triển Giao diện người dùng (Frontend): Xây dựng ứng dụng Web hoặc

Thiết kế và cài đặt cơ sở dữ liệu NoSQL cho hệ thống diễn đàn Hỏi-Đáp chuyên ngành với tính năng đánh giá và vinh danh.

Mobile (sử dụng ReactJS, VueJS hoặc Flutter) để kết nối với cơ sở dữ liệu đã thiết kế, giúp người dùng cuối có thể tương tác trực quan với hệ thống.

- Tối ưu hóa Tìm kiếm (Full-text Search): Tích hợp các công cụ tìm kiếm mạnh mẽ như Elasticsearch hoặc sử dụng tính năng Atlas Search của MongoDB để hỗ trợ tìm kiếm câu hỏi theo ngữ nghĩa, từ khóa gợi ý và sửa lỗi chính tả, thay vì chỉ tìm kiếm chính xác theo thẻ (Tag) như hiện tại.

- Tính năng Thời gian thực (Real-time): Sử dụng MongoDB Change Streams để lắng nghe các thay đổi trong cơ sở dữ liệu và gửi thông báo tức thời đến người dùng khi câu hỏi của họ được trả lời hoặc được upvote.

DANH MỤC TÀI LIỆU THAM KHẢO

- [1] A. Meier and M. Kaufmann, *SQL & NoSQL Databases: Models, Languages, Consistency Options and Architectures for Big Data Management*. Wiesbaden: Springer Fachmedien Wiesbaden, 2019. doi: 10.1007/978-3-658-24549-8.
- [2] S. Bradshaw, E. Brazil, and K. Chodorow, “MongoDB: the Definitive Guide”.
- [3] “MongoDB là gì? Định nghĩa đầy đủ và chi tiết về MongoDB.” Accessed: Jan. 02, 2026. [Online]. Available: <https://compaclass.com/blog/p/mongodb-la-gi-inh-nghia-ay-u-va-chi-tiet-ve-mongodb-gnBNS>
- [4] K. Banker, *MongoDB in Action: Covers MongoDB Version 3.0*. New York: Manning Publications Co. LLC, 2016.
- [5] R. Copeland, *MongoDB applied design patterns*. Sebastopol, CA: O'Reilly Media, Inc, 2013.