**MINAR Tạp chí Quốc tế về Khoa học và Công nghệ**

**Ứng dụng ISSN: 2717-8234**

**Loại bài viết: Bài báo nghiên cứu**

|  |  |  |
| --- | --- | --- |
| **Ngày nhận: 04/06/2022** | **Ngày nhận được: 19/06/2022** | **Công bố : 01/09/2022** |

**SO SÁNH GIỮA NOSQL VÀ RDBMS: STORAGE VÀ RETRIEVA**

**Sameera Abbas FADHEL 1**

**Đại học Mosul, Iraq**

**Enas Ali JAMEEL 2**

**Đại học Mosul, Iraq**

**Trừu tượng**

Các hệ thống quản lý cơ sở dữ liệu quan hệ (RDBMS) nổi lên như một giải pháp cho việc lưu trữ dữ liệu trong những thập kỷ qua. Tất cả các hệ thống và ứng dụng lưu trữ dữ liệu đều sử dụng RDBMS ở trung tâm của hệ thống để lưu trữ và truy xuất dữ liệu. Trong vài năm qua, một mô hình lưu trữ dữ liệu mới, có tên là Hệ thống quản lý dữ liệu không quan hệ (NOSQL: Not Only Structured Query Language), đã xuất hiện để tạo ra các hệ thống lưu trữ dữ liệu ít phức tạp hơn và giải quyết tình trạng suy giảm hiệu suất khối lượng lớn dữ liệu của các hệ thống đã sử dụng. Trong công việc này, một nghiên cứu so sánh được thực hiện giữa MySQL như một ví dụ về RDBMS và Monogodb như một ví dụ về các hệ thống NOSQL sử dụng luồng và tài nguyên máy để chỉ ra sự khác biệt cho các nhà phát triển chọn một trong những mô hình này cho ứng dụng của họ. Kết quả cho thấy hiệu suất của NoSQL kém hơn MySQL đối với các tập dữ liệu nhỏ và ít thao tác cơ sở dữ liệu, chẳng hạn như vài nghìn bản ghi và hàng trăm thao tác mỗi ngày. Tuy nhiên, với sự ra đời của các luồng và khối lượng dữ liệu, hiệu suất của Mongodb vượt qua Ngôn ngữ truy vấn có cấu trúc của tôi (MySQL). Ngoài ra, kết quả đã chỉ ra rằng Mongodb yêu cầu sử dụng nhiều bộ nhớ và tài nguyên CPU hơn MySQL để hoàn thành nhiệm vụ của họ. Cuối cùng, hình ảnh được lưu dưới dạng dữ liệu byte bên trong cả hai nền tảng. Quá trình lưu trữ dữ liệu này bên trong Mongodb nhanh hơn nền tảng MySQL.

Từ khóa: Mongodb, NoSQL, Hệ quản trị cơ sở dữ liệu quan hệ (RDBMS), MySQL, Sử dụng CPU,Threading( Phân luồng).

**Giới thiệu**

Dữ liệu là dầu mới [1], một câu được sử dụng rộng rãi trong vài năm qua. Vào năm 2020, đã có báo cáo rằng 2,5 triệu byte dữ liệu đã được tạo trong các cơ sở hàng ngày. Hơn nữa, ước tính vào năm 2025, thế giới sẽ có 175 zettabyte dữ liệu. Với sự ra đời của Internet vạn vật (IoT) và các ứng dụng của chúng, hàng tỷ thiết bị sẽ tạo ra một lượng dữ liệu khổng lồ hàng ngày, chẳng hạn như điện thoại thông minh, đồng hồ thông minh, máy tính, máy tính bảng, kính thông minh và thậm chí cả thiết bị gia dụng [3]. Tất cả dữ liệu này phải được chuyển từ vị trí này sang vị trí khác để phân tích( analytics) và phân giải ( analysis). Tuy nhiên, trước khi xử lý lượng dữ liệu khổng lồ này, dữ liệu này phải được lưu trữ. Cơ sở dữ liệu đã chiếm ưu thế trong khu vực lưu trữ. Tuy nhiên, với việc giới thiệu lượng dữ liệu khổng lồ này, cấu trúc cơ sở dữ liệu cổ điển, được đặt tên là RDBMS, đã tiết lộ nhiều vấn đề về hiệu suất [4]. Để giải quyết những vấn đề này, các hệ thống và mô hình cơ sở dữ liệu khác nhau đã được đề xuất trong vài năm qua. Một trong những mô hình đã trở nên phổ biến này là NoSQL.

Ý tưởng cho rằng “dữ liệu là dầu mới” có liên quan đến những điểm tương đồng trong cách hai nguồn tài nguyên trở nên có giá trị. Cũng giống như dầu mỏ, bản thân dữ liệu thô không có giá trị; đúng hơn, giá trị được tạo ra khi nó được thu thập nhanh chóng, đầy đủ, chính xác và được kết nối với các dữ liệu liên quan khác.

NoSQL đã được đề xuất để xử lý khối lượng dữ liệu theo các phương pháp hiệu quả hơn. Vai trò và cấu trúc của nó không tuân theo mô hình quan hệ của RDBMS cổ điển. Trong RDBMS, các "bảng" quan hệ được sử dụng để ghi dữ liệu. Mỗi bảng có một khóa không trùng lặp đặc biệt để xác định từng hàng "bản ghi dữ liệu". Các khóa này được tận dụng để tạo kết nối và liên kết giữa các quan hệ. Điều này có nghĩa là trước khi lưu trữ bất kỳ bản ghi dữ liệu nào trong hệ thống RDBMS, sơ đồ cơ sở dữ liệu phải được tạo. Ngoài ra, chuẩn hóa dữ liệu phải được thực hiện trên dữ liệu để tạo sơ đồ cơ sở dữ liệu mà không có dữ liệu trùng lặp. Các vai trò này không được tuân theo trong các hệ thống NoSQL. Hơn nữa, NoSQL không đáp ứng tính nguyên tử, tính nhất quán, tính cô lập và độ bền (ACID: Atomicity, Consistency, Isolation, Durability) [5,6]. NoSQL áp dụng các mô hình dữ liệu khác nhau, chẳng hạn như cột, khóa-giá trị, tài liệu và biểu đồ. Mô hình dữ liệu tài liệu đã chiếm ưu thế trong cách tiếp cận NoSQL. Theo cách này, cơ sở dữ liệu là một tập hợp các khối tài liệu độc lập. Mỗi tài liệu lưu trữ sơ đồ và dữ liệu riêng của nó. Như đã đề cập, không có sơ đồ cơ sở dữ liệu nào được sử dụng. Điều này có nghĩa là các tài liệu được chia sẻ có thể có các lược đồ và cấu trúc khác nhau. Các tài liệu phải được mã hóa để lưu trữ trong mô hình NoSQL. Các phương pháp mã hóa khác nhau có thể được tận dụng, chẳng hạn như ngôn ngữ đánh dấu mở rộng (XML: extensible markup language), ký hiệu đối tượng JavaScript (JSON: JavaScript object notation), JSON nhị phân (BSON: Binary JSON: Binary JavaScript Object Notation), Ngôn ngữ tuần tự hóa dữ liệu mà con người có thể đọc được (YAML: human-readable data-serialization language) và định dạng nhị phân(binary format) [7]. Mô hình lưu trữ dữ liệu dựa trên tài liệu này cho phép hệ thống NoSQL dễ dàng được phân chia theo số lượng máy chủ trong trung tâm dữ liệu và đám mây. Câu hỏi nổi lên là, nên chọn mô hình cơ sở dữ liệu nào cho ứng dụng của bạn? Ứng dụng có yêu cầu NoSQL hoặc RDBMS không?

Trong công trình này, một nghiên cứu so sánh giữa mô hình NoSQL và RDBMS được tiến hành. Mongodb đã được chọn làm ví dụ về mô hình NoSQL. Nó đã được chọn vì nó là hệ thống NoSQL phổ biến nhất trên thế giới [8]. MySQL đã được chọn làm ví dụ về hệ thống DBMS. Hai hệ thống này đã được so sánh theo nhiều thước đo hiệu suất khác nhau. Đầu tiên, tác động của các yêu cầu phân luồng và song song đối với các hệ thống này sẽ được nghiên cứu. Số lượng luồng khác nhau sẽ được tạo để thực hiện các thao tác truy xuất dữ liệu đơn giản và phức tạp trên các hệ thống này. Thứ hai, trình kết nối Python cho hai hệ thống này sẽ được điều tra. Python sẽ được tận dụng để tạo các luồng và cố gắng kết nối với cơ sở dữ liệu. Python đã được tận dụng vì nó đã thống trị trong lĩnh vực khoa học dữ liệu và máy học trong vài năm qua. Cuối cùng, việc sử dụng CPU và sử dụng bộ nhớ sẽ được đo để hiển thị tải của các hệ thống này trên tài nguyên máy tính. Trong nghiên cứu thử nghiệm này, các câu hỏi sau cần được trả lời:

* Nền tảng nào nhanh hơn khi sử dụng luồng?
* Nền tảng nào tải ít tài nguyên hệ thống hơn, chẳng hạn như bộ nhớ và CPU?
* Nền tảng nào có thể xử lý lượng dữ liệu khổng lồ nhanh hơn?
* Trình kết nối Python có phải là lựa chọn tốt cho các hệ thống này không?
* Làm thế nào để cả hai hệ thống xử lý các cấu trúc dữ liệu khác nhau, chẳng hạn như hình ảnh?

Phần còn lại của bài viết này được tổ chức như sau; trong phần tiếp theo, một số công trình và thử nghiệm đã được thực hiện trong lĩnh vực so sánh hệ thống cơ sở dữ liệu sẽ được tổng quan. Phần 3 giới thiệu phương pháp đã được sử dụng và các công cụ được sử dụng. Phần 4, tổng quan các kết quả thu được. Cuối cùng, kết luận của bài viết này trong phần 5.

**2. Các công việc liên quan:**

Tất cả quá trình chuyển đổi từ RDBMS sang NoSQL đã phát triển mạnh mẽ trong vài năm qua [Bansal et al. 2021]. Các chi tiết kỹ thuật và phát triển khác nhau được đề xuất để chuyển đổi các bộ dữ liệu quan hệ cấu trúc thành các mô hình định hướng tài liệu [Abdelhedi et al.2022, Namdeo et al. 2012]. Các nhà nghiên cứu khác đã cố gắng thiết kế một cố vấn cho lược đồ thiết kế để chuyển từ RDBMS sang NoSQL [Dabowsa et al. 2021]. Tuy nhiên, tại sao họ lại chuyển đổi từ RDBMS sang các hệ thống NoSQL?

So sánh cơ sở dữ liệu quan hệ và cơ sở dữ liệu NoSQL đã thu hút nhiều nhà nghiên cứu trong những năm qua. Những so sánh này đã cố gắng đo lường thời gian cần thiết để thực hiện thao tác và kiểm soát dữ liệu cho cả hai nền tảng. Như ví dụ về các hệ thống đã thống trị như RDBMS, MySQL và Mongodb để so sánh trong các bài báo nghiên cứu. Ngoài ra nó còn thúc đẩy các bộ dữ liệu khác nhau phát triển. Ví dụ, trong [Jose et al. 2021], các tác giả đã cố gắng so sánh Mongo và MySQL để truy vấn dữ liệu trên hai bộ dữ liệu khác nhau. Họ đã chỉ ra rằng trong các tình huống khác nhau thời gian cần thiết để chọn đơn giản, chọn với điều kiện, cập nhật và chèn cho Mongodb ít hơn MySQL. Trong [Reichardt et al. 2021], các tác giả đã so sánh bốn hệ thống cơ sở dữ liệu, ba nền tảng NoSQL "Mongodb, Redis, Cassandra" và một nền tảng RDBMS "MySQL". Tác giả đã tận dụng thư viện trình điều khiển Python để kết nối với các nền tảng cơ sở dữ liệu. Tác giả đã báo cáo rằng các nền tảng NoSQL nhanh hơn MySQL để cập nhật, chèn, đọc và ghi cho các bộ dữ liệu nhỏ. Tuy nhiên, MySQL hoạt động tốt hơn cho các giá trị lớn "giá trị 300K". Trong [Pereira et al. 2018], các tác giả đã so sánh ba nền tảng NoSQL khác nhau '; Mongodb, Rethnkdb và Couchbase. Tác giả đã báo cáo rằng hiệu suất của Monogodb và Couchbase vượt qua Rethinkdb. Hơn nữa, các tác giả đã chỉ ra rằng Monogodb vượt qua Couchbase cho các hoạt động đa luồng. Tuy nhiên, Couchbase đã ghi được kết quả betere để chọn với điều kiện "GET with ID". Trong [Győrödi et al. 2020], các tác giả đã so sánh hiệu suất của MySQL và nền tảng Couchbase NoSQL. Các hoạt động CRUD đã được điều tra. Các kết quả được báo cáo trong công việc đã chỉ ra rằng MySQL hoạt động tốt hơn cho các hoạt động nhỏ và dữ liệu có cấu trúc. Tuy nhiên, đối với các hoạt động phức tạp và lớn, NoSQL hoạt động tốt hơn. Một so sánh khác giữa MySQL và Mongodb đã được báo cáo trong [Matallah et al. 2021], tác giả đã sử dụng Yahoo Cloud Serving Benchmark [Cooper et al. 2010]. Tác giả báo cáo rằng thời gian thực hiện hoạt động Mongodb ít hơn MySQL cho tất cả các loại hoạt động.

Trong [Sánchez-de-Madariaga et al. 2018], hồ sơ dữ liệu y tế đã được sử dụng để so sánh RDBMS và NoSQL. Các tác giả báo cáo rằng thời gian thực thi Mongodb thấp hơn MySQL cho tất cả các loại hoạt động. Trong [Das et al. 2019], các chỉ số tải và quét dữ liệu đã được sử dụng để so sánh các nền tảng MySQL và NoSQL HBase. Tác giả đã báo cáo rằng HBase hoạt động tốt hơn trong các số liệu này so với MySQL. Trong [Yassine et al. 2018], quá trình di chuyển từ MySQL sang Monogdb đã được điều tra. Hiệu suất của các hoạt động phức tạp đã được so sánh giữa hai nền tảng, Mongodb vượt trội hơn Mysql về các hoạt động phức tạp. Trong [Chakraborty et al 2021], một bộ dữ liệu của bộ gen COVID-19 đã được sử dụng cho mục đích so sánh giữa MySQL và Mongodb. Hai hoạt động chính đã được sử dụng cho mục đích so sánh, tải dữ liệu và các yêu cầu phức tạp. Mongodb đã vượt trội hơn MySQL cho cả hai hoạt động .

Với sự ra đời của điện toán đám mây và các dịch vụ của họ, cơ sở dữ liệu như một dịch vụ đã được giới thiệu bởi các nhà cung cấp đám mây khác nhau. RDBMS và NoSQL đã được thiết kế để hoạt động với các nền tảng đám mây khác nhau. Trong [Shareef et al. 2022], một cuộc khảo sát và so sánh các hệ thống cơ sở dữ liệu dựa trên đám mây khác nhau đã được xem xét. Khả năng mở rộng của các hệ thống này đã được tổng quan. Trong [Zaman et al. 2021], một so sánh giữa Azure SQL và Atlas Mongodb đã được tiến hành. Kết quả đã chỉ ra rằng thời gian tải, truy xuất và tải lên dữ liệu cho Azure ít hơn nhiều so với Atlas. Những kết quả này cho thấy quá trình lựa chọn nền tảng đám mây là một quá trình quan trọng . Công việc này khác với các tác phẩm khác trong ba nếp gấp chính. Đầu tiên, lưu trữ và truy xuất các loại dữ liệu đặc biệt, chẳng hạn như hình ảnh. Thứ hai, trình kết nối cơ sở dữ liệu Python sẽ được sử dụng để tải dữ liệu và thực hiện các yêu cầu. Bằng cách này, hiệu suất của cơ sở dữ liệu với giao diện của chúng đã được đo lường. Cuối cùng, tài nguyên của các yêu cầu, chẳng hạn như việc sử dụng bộ nhớ và sử dụng CPU bằng cách sử dụng các luồng khác nhau để truy cập cơ sở dữ liệu song song sẽ được đo lường.

**3. Chi tiết thử nghiệm.**

Để so sánh nền tảng của hai cơ sở dữ liệu, một tập dữ liệu phải được sử dụng. Tập dữ liệu từ hồ sơ của sinh viên và lớp học của họ đã được tạo. Hơn nữa, thông tin của 500 sinh viên với 20 giảng viên đã được thêm vào để tạo giản đồ cơ sở dữ liệu.



Hình 1 Thiết kế lược đồ cơ sở dữ liệu .RDBMS

Hình 1 cho thấy sơ đồ được thiết kế. Bảng đầu tiên bao gồm thông tin của 500 sinh viên. Bảng thứ hai bao gồm thông tin của 20 giảng viên. Bảng thứ ba có thông tin điểm của những học sinh này. Bảng này bao gồm 20.000 bản ghi. Bảng thứ tư bao gồm thông tin của các lớp. Bảng thứ năm bao gồm thông tin về chương trình giảng dạy. Bảng này được tạo ra từ mối quan hệ giữa chương trình giảng dạy, sinh viên và giảng viên. Để điền vào các bảng này, một trang chủ đã được tạo và mỗi học sinh đã được yêu cầu điền điểm cho tất cả các lớp đã tham dự. Sau đó, dữ liệu đã được trích xuất từ các bảng của cơ sở dữ liệu thành các tệp csv. Hơn nữa, dữ liệu này đã được xuất từ MySQL để tải vào cơ sở dữ liệu mới. Ngoài ra, các tệp này đã được phân tích để tạo tài liệu dữ liệu cho từng sinh viên, sinh viên tốt nghiệp, lớp học và giảng viên. Các tài liệu này đã được chuyển đổi thành tệp JSON để tải vào Mongodb.

Mongodb phiên bản 5.0.6 đã được sử dụng cho cơ sở dữ liệu NoSQL và MySQL server phiên bản 8.0.28 đã được sử dụng cho RDBMS. PC được sử dụng trong thử nghiệm này là Intel i5 thế hệ 10 với RAM 8G và bộ nhớ 1 Tbyte. Windows 10 đã được sử dụng làm hệ điều hành cho PC. Python 3.6 đã được sử dụng để kết nối với cơ sở dữ liệu. Để xuất tập dữ liệu từ cơ sở dữ liệu được tạo đầu tiên, công cụ xuất của MySQL đã được sử dụng. Để tạo lược đồ cho MySQL, công cụ máy khách MySQL đã được sử dụng. Cuối cùng, để kết nối cơ sở dữ liệu với Python, hai thư viện trình kết nối đã được tải xuống. Trình kết nối đầu tiên là Pymongo kết nối với cơ sở dữ liệu Mongodb. Mặt khác, trình kết nối Mysql-connector-python-8.0.28 đã được sử dụng để kết nối Python với MySQL.

Bảng 1 . Thời gian tải của cả hai nền tảng

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
|  | **Nền tảng** | |  | **Thời gian** | |
|  |  |  |  |  |  |
|  | **MySQL** |  |  | 11 phút |  |
|  |  |  |  |  |  |
|  | **Mongodb** | |  | 5 phút | |
|  |  |  |  |  |  |

Phiên bản MySQL đã xuất của các tệp và tệp JSON để tải chúng vào cơ sở dữ liệu mới được tạo đã được sử dụng. Bảng 1 cho thấy thời gian cần thiết để tải cả hai tệp vào cơ sở dữ liệu mới. Quan sát từ bảng rằng thời gian cần thiết để tải các tệp JSON vào Mongodb nhỏ hơn thời gian cần thiết để tải cùng một dữ liệu vào MySQL vì tệp được nhập từ MySQL có các câu lệnh SQL cần được thực thi từng cái một để tạo cơ sở dữ liệu giống như bản gốc. Tuy nhiên, trong Mongodb, tệp JSON chỉ được sử dụng để tạo tài liệu cho mỗi mục nhập trong tệp. Ngoài ra, các tệp JSON cho tất cả dữ liệu "sinh viên, giảng viên, lớp học và lớp" đã được chèn vào cùng một bộ sưu tập mà không cần tạo lược đồ mới. Đây không phải là trường hợp của MySQL yêu cầu tạo một lược đồ mới cho mỗi bảng và chèn dữ liệu của nó vào một quy trình riêng biệt.

**3.2 Hoạt động và yêu cầu**

Để so sánh hiệu suất của hai nền tảng bằng cách sử dụng tập dữ liệu được tạo, bốn hoạt động khác nhau đã được tận dụng. Các tiểu mục sau đây giới thiệu các hoạt động này.

**A- Chọn hoặc Tìm**

Thao tác đầu tiên là tìm kiếm hoặc "chọn" yêu cầu. Ba phiên bản đã được sử dụng từ cuộc điều tra này. Đầu tiên là một lệnh tìm đơn giản phổ biến trên bảng đánh dấu bao gồm 20.000 bản ghi. Thao tác chọn này đã truy xuất bản ghi giới hạn ở một số lượng bản ghi khác nhau. Trong hình thức thứ hai của cuộc điều tra này, một điều kiện đã được thêm vào để chỉ chọn điểm của các sinh viên trong cùng một chuyên ngành. Phiên bản cuối cùng là sử dụng 10 luồng để truy xuất các dấu cho mười chuyên ngành khác nhau từ cùng một cơ sở dữ liệu.

Bảng 2 . Các yêu cầu chọn lọc được thực hiện

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  | **Hoạt động** | |  | **Nền tảng** | |  |  | **Truy vấn** | |
|  |  |  |  |  |  |  |  |  |  |
|  | **Cập nhật** |  |  | MySQL |  |  |  | Select \* from marks LIMIT |  |
|  | **Đơn giản** |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
|  | **Cập nhật** | |  | MySQL | |  |  | Select \* from marks where “student\_id” like | |
|  | **Phức tạp** | |  |  |  |  |  | {%num%} | |
|  |  | |  |  | |  |  |  | |
|  | **Cập nhật** |  |  | Mongodb |  |  |  | db.marks.find.limit(). explain() |  |
|  | **Đơn giản** |  |  |  |  |  |  |  |  |
|  |  |  |  |  | |  |  |  | |
|  | **Cập nhật** | |  | Mongodb | |  |  | db.marks.findOne({“student\_id”: | |
|  | **Phức tạp** | |  |  |  |  |  | {$regex:”num”}}) | |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |

20 luồng khác nhau đã được tạo bằng thư viện luồng trong Python. Mỗi luồng tạo một kết nối khác bằng cách sử dụng thư viện trình kết nối cho từng nền tảng. Thời gian được ghi lại khi tất cả các luồng lấy dữ liệu từ cơ sở dữ liệu. Ngoài ra, việc sử dụng CPU đã được ghi lại cho từng hệ thống khi sử dụng các luồng này. Bảng 2 cho thấy các câu hỏi đã được sử dụng trong hoạt động đầu tiên.

**B- Truy vấn cập nhật**

Loại truy vấn được sử dụng thứ hai là lệnh cập nhật. Ba phiên bản cập nhật đã được sử dụng cho quá trình so sánh.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  |  | |  | | --- | | Bảng 3 . Các yêu cầu cập nhật được thực hiện | |  | |
|  |  |  |
| **Hoạt động** | **Nền tảng** | **Truy vấn** |
|  |  |  |
|  |  |  |
| **Cập nhật** | MySQL | update marks set marks =50 where “student\_id”=xx |
| **Đơn giản** |  |  |
|  |  |  |
| **Cập nhật** | MySQL | update marks set marks =50 where “ student\_id” like {%num%} |
| **Phức tạp** |  |  |
|  |  |  |
| **Cập nhật** | Mongod | db.marks.explain().update({“studnet\_id”:xx},{$set:{”mark”:50}}) |
| **Đơn giản** | b |  |
|  |  |  |
| **Cập nhật** | Mongod | db.marks.explain().update({“studnet\_id”:{$regex:num}},{$set:{”mark |
| **Phức tạp** | b | ”:50}, {$multi:true}}) |
|  |  |  |

Truy vấn cập nhật đầu tiên là cập nhật một trường chọn trong bản ghi duy nhất trong bảng đánh dấu. Phiên bản thứ hai là cập nhật tất cả các điểm cho tất cả các sinh viên trong cùng một chuyên ngành. Cuối cùng, phiên bản cuối cùng là chỉ cập nhật một bản ghi cho 20 sinh viên sử dụng chủ đề. Các luồng đã được tạo và định cấu hình như trong yêu cầu chọn. Bảng 3 cho thấy các yêu cầu cập nhật được thực hiện.

**C- Xóa hoạt động**

Thao tác thứ ba là quá trình xóa dữ liệu. Như trong phần chọn và cập nhật, thao tác này bao gồm ba phiên bản.

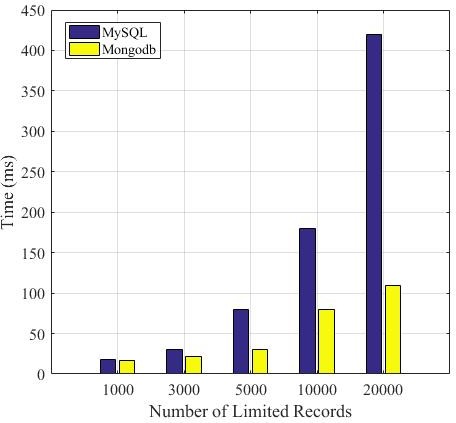
|  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  |  |  |  |  |  |  | |  |  | | --- | --- | |  | Bảng 4 . Các yêu cầu xóa được thực hiện | | |
|  |  |  |  |  |  |  |  |  |
|  | **Hoạt động** | |  | **Nền tảng** | |  | **Truy vấn** | |
|  |  |  |  |  |  |  |  |  |
|  | **Cập nhật** |  |  | MySQL |  |  | Delete from marks where “student\_id”=xx |  |
|  | **Đơn giản** |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |
|  | **Xóa** | |  | MySQL | |  | Delete from marks where “ student\_id” like {%num%} | |
|  | **Nhiều** | |  |  |  |  |  |  |
|  |  | |  |  | |  |  | |
|  | **Cập nhật** |  |  | Mongodb |  |  | db.marks.explain().deleteOne({“studnet\_id”:xx},{$set:{”mark”: |  |
|  | **Đơn giản** |  |  |  |  |  | 50}}) |  |
|  |  |  |  |  | |  |  |  |
|  | **Xóa** | |  | Mongodb | |  | db.marks.explain().deleteMany({“studnet\_id”:{$regex:num}},{$ | |
|  | **Nhiều** | |  |  |  |  | set:{”mark”:50}, {$multi:true}}){$multi:true}}) | |
|  |  |  |  |  |  |  |  |  |

Trong loại đầu tiên, một bản ghi sẽ bị xóa. Trong phiên bản thứ hai, nhiều máy ghi âm được chọn cho quá trình xóa. Cuối cùng, 20 luồng được sử dụng để xóa một 20 bản ghi. Bảng 4 cho thấy các yêu cầu được thực hiện đối với hai hệ thống.

**4. Kết quả**

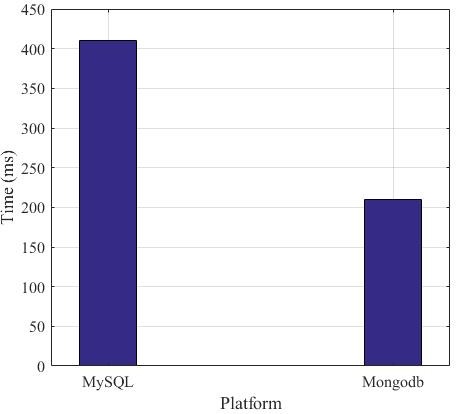
Phần này gồm 4 mục chính. Mỗi mục thảo luận kết quả thu được từ các câu hỏi trong từng trường hợp.

**4.1 Chọn trường hợp**



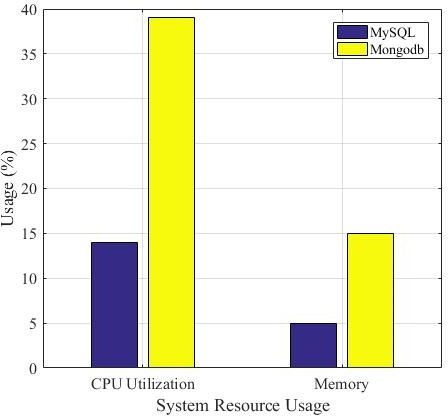
**Hình 2 .Thao tác Simple Select**

Hình 2 cho thấy thời gian thực hiện thao tác chọn đơn giản sử dụng một luồng với số lượng bản ghi được truy xuất khác nhau. Các quan sát cho thấy thời gian là gần giống như nhau cho cả hai hệ thống với bản ghi nhỏ. Tuy nhiên, Tuy nhiên, với số lượng Mongodb cao hơn ghi lại thời gian thực hiện ít hơn.



**Hình 3 .So sánh luồng của Simple Select**

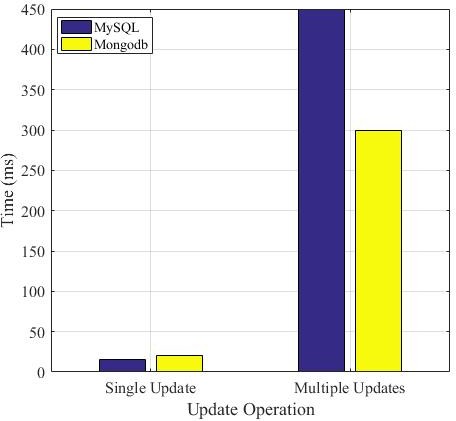
Hình 3 cho thấy thời gian thực hiện được ghi lại cho truy vấn compex đối với hai nền tảng. Quan sát cho thấy Mongodb ghi lại thời gian truy xuất ít hơn MySQL.



**Hình 4 .Tài nguyên hệ thống cho luồng của Simple Select**

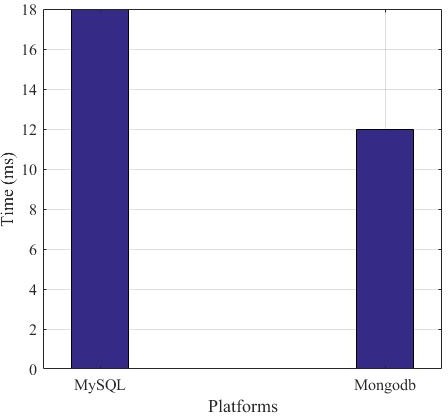
Cuối cùng, hình 4 cho thấy cách cả hai hệ thống hoạt động với số lượng luồng khác nhau. Tuy nhiên, việc quan sát Mongodb có ít thời gian thực hiện hơn. Trong hình 4, quan sát về Mongodb ta thấy đã sử dụng thêm 25% mức sử dụng CPU trong số tất cả các luồng được tạo. Ngoài ra, việc sử dụng bộ nhớ của Mongodb so với máy chủ MySQL cho các luồng này được hiển thị trong hình 4. Mức sử dụng bộ nhớ của Mongodb cao hơn 10% khi sử dụng các luồng này. Điều này có nghĩa là có sự đánh đổi giữa tài nguyên hệ thống và thời gian thực thi cho hai nền tảng này. Điều đáng nói là không so sánh lựa chọn phức tạp vì nó tương tự như nhiều bản cập nhật được hiển thị trong phần tiếp theo.

**4.2 Trường hợp cập nhật**



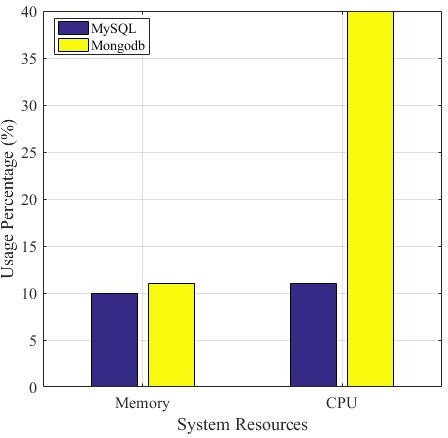
**Hình 5 .So sánh cập nhật**

Hình 5 cho thấy thời gian thực hiện cập nhật bản ghi bằng cả hai nền tảng. Quan sát cho thấy một bản cập nhật của MySQL đã ghi lại thời gian tốt hơn. Tuy nhiên, khi số lượng bản ghi tăng lên, thời gian của Mongodb sẽ giảm đi so với MySQL.

****

**Hình 6 .So sánh luồng cho quá trình cập nhật**

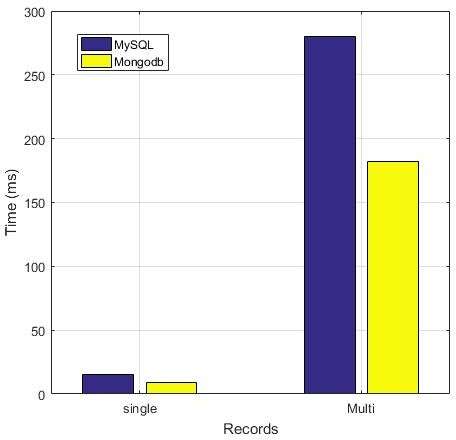
Hình 6 cho thấy thời gian bản ghi cập nhật sử dụng các luồng. Các kết nối luồng tốn ít thời gian hơn trong Mongodb so với MySQL.



**Hình 7 .Sử dụng tài nguyên hệ thống trong Cập nhật**

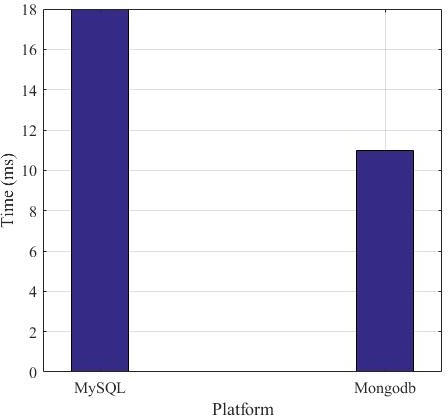
Tuy nhiên, hình 7 cho thấy mức sử dụng bộ nhớ và CPU của cả Mongodb và MySQL. Quan sát việc cập nhật, mức sử dụng bộ nhớ của cả hai hệ thống là gần như nhau. Tuy nhiên, đối với việc sử dụng CPU, Mongodb đã tăng mức sử dụng CPU lên tới 40% so với MySQL chỉ tăng 11%.

**4.3. Trường hợp xóa dữ liệu**



**Hình 8 .So sánh việc xóa giữa hai nền tảng**

Hình 8 cho thấy thời gian cần thiết để xóa dữ liệu trong cả hai hệ thống. Việc quan sát để xóa một tài liệu dễ hơn xóa một bản ghi trong một bảng.



**Hình 9 .So sánh luồng khi xóa**

Thời gian thực hiện Mongodb tốt hơn MySQL ngay cả đối với một bản ghi. Hơn nữa, đối với số lượng bản ghi, Mongodb đã ghi lại thời gian thực hiện thấp hơn. Đối với luồng, Mongodb cũng đã ghi lại thời gian tốt hơn như trong Hình 9. Tuy nhiên, đối với tài nguyên hệ thống, các kết quả tương tự như hiển thị đối với các yêu cầu cập nhật dữ liệu đã được ghi lại.

**4.4 Lưu trữ hình ảnh**

Hệ thống được xây dựng bằng cách thêm hình ảnh của sinh viên và giảng viên vào cơ sở dữ liệu đã được cải thiện. Để lưu hình ảnh trong cơ sở dữ liệu, hệ thống lưu trữ những hình ảnh này trực tiếp trong các thư mục hệ thống tệp của hệ điều hành và lưu liên kết dưới dạng “varchar” trong một cột trong bảng trong cơ sở dữ liệu RDBMS cổ điển. Điều tương tự cũng có thể xảy ra trong NoSQL. Tuy nhiên, hình ảnh có thể được lưu trữ dưới dạng dữ liệu byte bên trong Mongodb và có thể được lưu trữ dưới dạng “blob”. Có một kiểu dữ liệu được gọi là longblob có thể được tận dụng trong MySQL. Các hình ảnh đã được lưu bên trong Mongodb và MySQL.

**Bảng 5 . So sánh chèn và truy xuất**

|  |  |  |
| --- | --- | --- |
| **Hoạt động** | **Nền tảng** | **Thời gian** |
| **Chèn** | MySQL | 110  ms |
| **Truy xuất** | MySQL | 87  ms |
| **Chèn** | Mongodb | 90  ms |
| **Truy xuất** | Mongodb | 77  ms |

Bảng 5 cho thấy thời gian cần thiết để lưu một hình ảnh và truy xuất nó cho cả hai hệ thống. Thời gian quan sát được cao đối với cả hai nền tảng. Tuy nhiên, nó tốt hơn trong Mongodb. Phương pháp này không được khuyến khích cho cả hai hệ thống.

**5. Kết luận:**

Trong công trình này, một nghiên cứu so sánh đã được tiến hành giữa các nền tảng RDBMS và NoSQL cho các hoạt động dữ liệu khác nhau. Một bộ dữ liệu mới đã được thu thập từ các sinh viên đại học. Luồng và tài nguyên hệ thống đã được tận dụng làm thước đo hiệu suất cho quá trình so sánh. Hơn nữa, hình ảnh cũng đã được sử dụng như một kiểu dữ liệu phức tạp để chèn và truy xuất. Dữ liệu cho thấy Mongodb vượt qua MySQL về các loại dữ liệu phức tạp và khối lượng dữ liệu lớn. Tuy nhiên, Mongodb tiêu tốn nhiều tài nguyên hệ thống hơn so với nền tảng MySQL cho bộ nhớ và CPU. Điều này gây ra vấn đề cho các nhà phát triển tận dụng và cho thuê tài nguyên từ các hệ thống đám mây. Quan sát để phát triển NoSQL Mongodb dễ dàng hơn MySQL vì nhà phát triển không yêu cầu học ngôn ngữ SQL. Hơn nữa, không cần lược đồ, chuẩn hóa dữ liệu. Ngoài ra, việc nhập và xuất dữ liệu dễ dàng hơn với NoSQL vì nó xử lý các tài liệu.

**6. Tài liệu tham khảo:**

1. Humby, C. (2006). Data is the new oil. Proc. ANA Sr. Marketer’s Summit. Evanston, IL, USA.
2. Data Statistics. (october, 2021). How much data is creted everyday.

<https://seedscientific.com/how-much-data-is-created-every-day>

1. Masoud, M, Jaradat, Y, Manasrah, A, Jannoud, I. (2019). Sensors of smart devices in the internet of everything (IoE) era: big opportunities and massive doubts. *Journal of Sensors.*
2. Kumawat, D and Pavate, A. (2018). Correlation of NOSQL & SQL Database*. IOSR J. Comput. Eng.(IOSR-JCE),* 18, 70-74.
3. Truica, C, Radulescu, F, Boicea and A, Bucur, I. (2015). Performance evaluation for CRUD operations in asynchronously replicated document oriented database. *In 2015 20th International Conference on Control Systems and Computer Science,* 191-196.
4. Gupta, S and Rani, R. (2016). *Data Transformation and Query Analysis of Elasticsearch and CouchDB Document Oriented Databases*. PhD diss.
5. Meier, A and Kaufmann. (2019). M. SQL & NoSQL Databases: Models, Languages, Consistency Options and Architectures for Big Data Management; Springer Vieweg: Wiesbaden, Germany.
6. DB engines. (2022). Database system ranking. <https://db-engines.com/en/ranking_trend>
7. Bansal, N, Soni, K and Sachdeva, S. (2021). Journey of Database Migration from RDBMS to NoSQL Data Stores. In International Conference on Big Data Analytics, 159 - 177 .
8. Abdelhedi, F, Jemmali, R and Zurfluh, G. (2022). Relational Databases Ingestion into a

NoSQL Data Warehouse. *arXiv preprint arXiv:2203.06949.*

1. Namdeo, B and Suman U. (2012). Schema design advisor model for RDBMS to NoSQL database migration. *International Journal of Information Technology*, 13 (1), 277 - 286.
2. Dabowsa, N, Maatuk, A, Elakeili, S and Ali, A. (2021). Converting Relational Database to Document-Oriented NoSQL Cloud Database. *In 2021 IEEE 1st International Maghreb Meeting of the Conference on Sciences and Techniques of Automatic Control and Computer Engineering MI-STA*, 381-386.
3. Jose, B and Abraham S. (2020). Performance analysis of NoSQL and relational databases with MongoDB and MySQL. *Materials today: PROCEEDINGS 24* , 2036-2043.
4. Reichardt, M, Gundall, M and Schotten, H. (2021). Benchmarking the Operation Times of NoSQL and MySQL Databases for Python Clients. *In IECON 2021–47th Annual Conference of the IEEE Industrial Electronics Society*, 1-8.
5. Pereira, D, Morais, W and Freitas E. (2018). NoSQL real-time database performance comparison. *International Journal of Parallel, Emergent and Distributed Systems* , 2, 144-156.
6. Győrödi, C, Dumşe-Burescu, D, Zmaranda, D, Győrödi, R, Gabor, G, and Pecherle, G. (2020). Performance Analysis of NoSQL and Relational Databases with CouchDB and MySQL for Application’s Data Storage. *Applied Sciences*, 23 (8524).
7. Matallah, H, Belalem, G and Bouamrane, K. (2021). Comparative study between the MySQL relational database and the MongoDB NoSQL database. *International Journal of Software Science and Computational Intelligence (IJSSCI)*, 3, 38-63.
8. Cooper, B, Silberstein, A, Tam, E, Ramakrishnan, R and Sears, R. (2010). Benchmarking cloud serving systems with YCSB. *In Proceedings of the 1st ACM symposium on Cloud computing* ,143-154.
9. Sánchez-de-Madariaga, R, Muñoz, A, Castro, A, Moreno, O and Pascual, M. (2018). Executing complexity-increasing queries in relational (MySQL) and NoSQL (MongoDB and EXist) size-growing ISO/EN 13606 standardized EHR databases. *Journal of Visualized Experiments*, e57439.
10. Das, N, Paul, S, Sarkar, B and Chakrabarti, S. (2019). NoSQL overview and performance testing of HBase over multiple nodes with MYSQL. *In Emerging technologies in data mining and information security*, 269-279.
11. Yassine, F and Awad, M. (2018). Migrating from SQL to NOSQL Database: Practices and Analysis. *In 2018 International Conference on Innovations in Information Technology (IIT)* , 58- 62.
12. Chakraborty, S, Paul, S and Hasan, KM. (2021). Performance comparison for data retrieval from nosql and sql databases: a case study for covid-19 genome sequence dataset. *In 2021 2nd International Conference on Robotics, electrical and signal processing techniques (ICREST)*, 324-328.
13. Shareef. T, Sharif. K, and Rashid. B. (2022). A Survey of Comparison Different Cloud Database Performance: SQL and NoSQL. *Passer Journal of Basic and Applied Sciences*, 4(1), 45-57.
14. Zaman. F, Khuhro. M, Kumar, K, Mirbahar, N, Khan, M and Kalhoro A. (2021). Comparative Case Study Difference Between Azure Cloud SQL and Atlas MongoDB NoSQL Database. *International Journal*, 9(7).