**HỆ QUẢN TRỊ CƠ SỞ DỮ LIỆU**

**ĐỀ TÀI 21:**

**SO SÁNH NoSQL MongoDB VỚI SQL DB**

**Nhóm 12 :**

**Dương Hồng Phong**

**Trần Đăng Tiến**

**Từ Hữu Trung**

**Trịnh Minh Triết**

**SO SÁNH NoSQL MongoDB VỚI SQL DB**

Zachary Parke

The University of Alabama Center for Advanced Public Safety

Tuscaloosa, AL 35487-0290

(205) 348-6363

[zparker@cs.ua.edu](mailto:zparker@cs.ua.edu)

Scott Poe

The University of Alabama Center for Advanced Public Safety

Tuscaloosa, AL 35487-0290

(205) 348-6363

[spoe@cs.ua.edu](mailto:spoe@cs.ua.edu)

Susan V. Vrbsky

The University of Alabama Department of Computer Science Tuscaloosa,

AL 35487-0290

(205) 348-6363

[vrbsky@cs.ua.edu](mailto:vrbsky@cs.ua.edu)

**TRỪU TƯỢNG**

Các giải pháp cơ sở dữ liệu NoSQL ngày càng trở nên phổ biến trong một thế giới hiện đang bị chi phối bởi các cơ sở dữ liệu quan hệ SQL. Cơ sở dữ liệu NoSQL được thiết kế để cung cấp các giải pháp cơ sở dữ liệu cho khối lượng lớn dữ liệu không có cấu trúc. Tuy nhiên, lợi thế (hoặc bất lợi) của việc sử dụng cơ sở dữ liệu NoSQL cho dữ liệu có cấu trúc và không nhất thiết phải là "Big", là không rõ ràng. Không có nhiều nghiên cứu so sánh hiệu suất xử lý một lượng nhỏ dữ liệu có cấu trúc trong cơ sở dữ liệu NoSQL với cơ sở dữ liệu quan hệ truyền thống. Trong bài báo này, chúng tôi so sánh một trong những giải pháp NoSQL, MongoDB, với cơ sở dữ liệu quan hệ SQL tiêu chuẩn, SQL Server. Chúng tôi so sánh hiệu suất, về thời gian chạy, của hai cơ sở dữ liệu này cho một cơ sở dữ liệu có cấu trúc với quy mô khiêm tốn. Kết quả cho thấy MongoDB hoạt động tốt hoặc tốt hơn so với cơ sở dữ liệu quan hệ, ngoại trừ khi các chức năng tổng hợp được sử dụng.

# Categories and Subject Descriptors

H.2.4 [Information Systems]: Systems – *relational databases*

# General Terms

Performance

# Keywords

MongoDB, NoSQL, performance, relational, structured data, SQL

1. **GIỚI THIỆU**

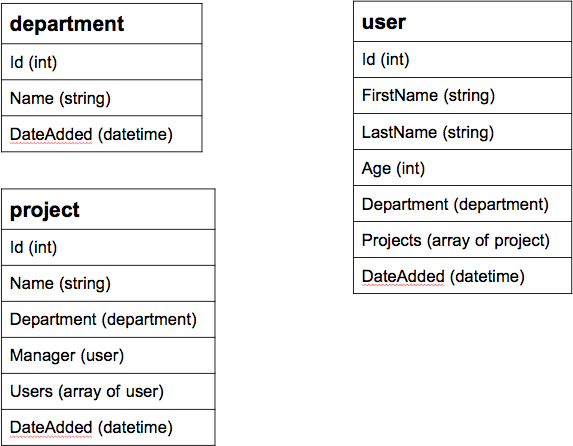
Triển khai cơ sở dữ liệu phổ biến nhất hiện nay dựa trên mô hình quan hệ [1] sử dụng SQL làm ngôn ngữ truy vấn. Tuy nhiên, các giải pháp cơ sở dữ liệu NoSQL (Not Only SQL) [2,3] đang trở nên nổi bật hơn khi một lượng lớn dữ liệu phát triển nhanh chóng đang được thu thập ngày nay. Dữ liệu này thường là phi cấu trúc, phức tạp và không phù hợp tốt với mô hình quan hệ. Ví dụ về loại dữ liệu này là bản ghi điện thoại thông minh trong đó vị trí được phát mỗi vài giây, nguồn cấp dữ liệu từ máy quay video trong không gian công cộng và thậm chí số lượng khổng lồ các trang và tài liệu trên web. Tuy nhiên, cũng có những tồn tại. Giấy phép làm bản sao kỹ thuật số hoặc khó khăn của một phần hoặc tất cả các công việc này cho các cá nhân hoặc lớp học sử dụng được cấp mà không cần lệ phí cung cấp các bản sao không được thực hiện hoặc phân phối lợi nhuận hoặc lợi thế thương mại và các bản sao chịu thông báo này và trích dẫn đầy đủ trên trang đầu tiên. Để sao chép theo cách khác, để xuất bản lại, đăng trên máy chủ hoặc phân phối lại vào danh sách, yêu cầu sự cho phép cụ thể trước và/hoặc phí. ACMSE'13, ngày 4-6 tháng 4 năm 2013, Savannah, GA, Hoa Kỳ. Bản quyền 2013 ACM 978-1-4503-1901-0/13/04... $ 15,00 là một số lượng lớn các cơ sở dữ liệu có kích thước khiêm tốn với dữ liệu có cấu trúc vẫn cần được lưu trữ và xử lý trong một cơ sở dữ liệu. Hiện đang có nhiều thảo luận trong thế giới cơ sở dữ liệu về việc liệu các cơ sở dữ liệu NoSQL có thay thế các cơ sở dữ liệu quan hệ hay không. Với việc chuyển sang cơ sở dữ liệu NoSQL và sự sẵn có của cơ sở dữ liệu NoSQL mã nguồn mở, một câu hỏi tự nhiên cho một nhà thiết kế cơ sở dữ liệu là có nên chọn cơ sở dữ liệu NoSQL hay không. Nếu cơ sở dữ liệu là phi cấu trúc và cực kỳ lớn, thì cơ sở dữ liệu NoSQL là một lựa chọn tốt. Tuy nhiên, không rõ ràng nếu sử dụng cơ sở dữ liệu NoSQL thay vì cơ sở dữ liệu quan hệ là một giải pháp khả thi cho một cơ sở dữ liệu có kích thước khiêm tốn với dữ liệu có cấu trúc. Dữ liệu trong mô hình quan hệ thường được thể hiện bằng lược đồ cơ sở dữ liệu [1], để nắm bắt ngữ nghĩa của cơ sở dữ liệu. Các đối tượng trong cơ sở dữ liệu có cùng số lượng đặc điểm, loại và định dạng được nhóm lại với nhau, làm cho nó có cấu trúc dữ liệu. Mô hình quan hệ được xây dựng trên giả định này của dữ liệu có cấu trúc, với dữ liệu của nó được lưu trữ trong các hàng và cột của một bảng, theo đó mỗi hàng có cùng số lượng và loại cột dữ liệu. Các bảng trong cơ sở dữ liệu quan hệ thường được chuẩn hóa, dẫn đến việc tạo ra nhiều bảng. Truy vấn các bảng đó đòi hỏi phải tìm nạp và kết hợp thông tin từ nhiều bảng khác nhau. Kết hợp thông tin dựa trên giá trị phù hợp cho khóa chính và khóa nước ngoài trên nhiều bảng trong mô hình quan hệ yêu cầu sử dụng thao tác tham gia. Lược đồ càng lớn và càng nhiều bảng cần được tham gia, cơ sở dữ liệu quan hệ càng mất nhiều thời gian để tìm nạp dữ liệu. NoSQL có thể giúp xử lý các dữ liệu không có cấu trúc. Dữ liệu có thể được bán cấu trúc, sao cho các đối tượng dữ liệu tương tự có thể được nhóm lại với nhau, nhưng các đối tượng có thể có các đặc điểm khác nhau. Thông tin sơ đồ cũng có thể được trộn lẫn với các giá trị dữ liệu trong dữ liệu bán cấu trúc, chẳng hạn như được tìm thấy trong dữ liệu XML. Dữ liệu không có cấu trúc có thể thuộc bất kỳ loại nào và có thể không có định dạng. Dữ liệu này không thể được thể hiện bằng bất kỳ loại lược đồ nào, chẳng hạn như các trang web trong HTML. Các tính năng điển hình của cơ sở dữ liệu SQL, chẳng hạn như các thuộc tính AXIT, đòi hỏi một lượng chi phí nhất định, và được nới lỏng hoặc loại bỏ trong cơ sở dữ liệu NoSQL để tối đa hóa hiệu suất. Nhiều cơ sở dữ liệu NoSQL tổ chức dữ liệu thành các cặp khóa-giá trị. Khóa được sử dụng để xác định duy nhất một mục dữ liệu cụ thể và giá trị có thể là một từ đơn giản, một số hoặc một cấu trúc phức tạp với ngữ nghĩa duy nhất. Sự phát triển của các truy vấn phức tạp hơn, không có ngôn ngữ truy vấn tiêu chuẩn và có giới hạn đối với các hoạt động. Cụ thể, không có hoạt động tham gia. Tuy nhiên, nhìn chung việc chế biến đơn giản hơn, giá cả phải chăng hơn và linh hoạt hơn. Trong bài báo này, chúng tôi so sánh một trong những giải pháp NoSQL, MongoDB [4,5], với một cơ sở dữ liệu quan hệ SQL tiêu chuẩn, SQL Server. Chúng tôi chạy thử nghiệm sử dụng cơ sở dữ liệu có cấu trúc kích thước khiêm tốn để xác định hiệu suất của cơ sở dữ liệu quan hệ với cơ sở dữ liệu NoSQL. Chúng tôi lưu ý rằng chúng tôi không xem xét bất kỳ thuộc tính ACID nào trong nghiên cứu của chúng tôi. Chúng tôi so sánh ba khía cạnh hiệu suất chính của các cơ sở dữ liệu này: tốc độ chèn, tốc độ cập nhật và tốc độ vận hành. Kết quả từ các thí nghiệm của chúng tôi cung cấp cái nhìn sâu sắc về khả năng tồn tại của việc sử dụng cơ sở dữ liệu NoSQL cho dữ liệu có cấu trúc có kích thước khiêm tốn. Bài báo này được sắp xếp như sau: Trong Phần 2, chúng tôi thảo luận về các công việc liên quan. Chúng tôi trình bày thiết lập thử nghiệm của chúng tôi trong Phần 3 và trong Phần 4 chúng tôi mô tả kết quả của các thử nghiệm của chúng tôi. Kết luận và công việc trong tương lai xuất hiện trong Phần 5.

1. **CÔNG VIỆC LIÊN QUAN**

Có nhiều blog, bài báo trắng và bình luận trực tuyến về lợi ích của cơ sở dữ liệu NoSQL so với cơ sở dữ liệu SQL, nhưng ít bài báo học thuật so sánh NoSQL với SQL. Thay vào đó, đã có một số bài báo gần đây đánh giá các cơ sở dữ liệu NoSQL khác nhau có sẵn [6-8]. Trong [6] các tác giả thực hiện một nghiên cứu điển hình về 14 cơ sở dữ liệu NoSQL khác nhau về các tính năng như mô hình dữ liệu của họ, khả năng truy vấn, kiểm soát đồng thời, phân vùng và cơ hội sao chép. Khuyến nghị của họ là sử dụng cơ sở dữ liệu NoSQL cho các hoạt động rất nhanh chóng và đơn giản cho các bộ dữ liệu rất lớn. Cơ sở dữ liệu NoSQL được so sánh với cơ sở dữ liệu SQL trong [9]. Chi phí do xử lý giao dịch trực tuyến được dự định không liên quan đến SQL, mà thay vào đó là các thành phần khác. Tác giả xem xét bốn thành phần trên của việc ghi nhật ký, khóa, chốt và quản lý bộ đệm. Loại bỏ chi phí chung liên quan đến một hoặc nhiều trong số này có thể cung cấp tốc độ tăng của cả hai. Tác giả kết luận rằng người ta có thể cải thiện cơ sở dữ liệu SQL để cạnh tranh với cơ sở dữ liệu NoSQL. Khi sử dụng triển khai NoSQL MongoDB, không có sơ đồ hoặc bảng cơ sở dữ liệu. MongoDB [4,5] thay vào đó sử dụng một "bộ sưu tập" tương tự như một bảng và "tài liệu" tương tự như hàng, để lưu trữ dữ liệu và thông tin sơ đồ. MongoDB tự động tạo khóa chính (id) để xác định duy nhất từng tài liệu. ID và tài liệu tương tự về mặt khái niệm với một cặp khóa-giá trị. MongoDB cố gắng giữ hầu hết dữ liệu trong bộ nhớ nên các truy vấn đơn giản mất ít thời gian hơn bằng cách loại bỏ nhu cầu lấy dữ liệu từ đĩa cứng. Một trong những điều này là một khi bộ dữ liệu trở nên lớn hơn bộ nhớ có sẵn, thì MongoDB sẽ phải bắt đầu truy vấn đĩa cứng để tìm kết quả. Vì MongoDB là một triển khai NoSQL, nó có khả năng mở rộng cao và không yêu cầu định nghĩa sơ đồ cứng nhắc. Điều này đôi khi có thể làm cho các truy vấn khó viết hơn vì người dùng có thể không biết chính xác phần nào của tài liệu cần truy vấn. Không cần phải có một hoạt động tham gia trong MongoDB. Lưu trữ dữ liệu trong MongoDB có thể được thực hiện theo một trong hai cách [4]. Cách thứ nhất là lồng các tài liệu vào nhau. Tùy chọn này có thể hoạt động cho các mối quan hệ một-một hoặc một-một-nhiều. Lưu ý rằng tùy chọn này không thể được sử dụng cho các mối quan hệ nhiều-nhiều vì nó có thể gây ra vòng lặp lồng nhau vô hạn. Tùy chọn thứ hai là lưu trữ một tham chiếu đến tài liệu khác thay vì lồng toàn bộ tài liệu. Với tùy chọn thứ hai này MongoDB sẽ chỉ cần truy xuất tài liệu tham chiếu khi người dùng yêu cầu dữ liệu bên trong tài liệu tham chiếu. Nhược điểm của tính năng này là MongoDB không có cách nào để truy xuất một đối tượng dựa trên tham chiếu. Điều này có nghĩa là người dùng phải xác định phương pháp riêng của họ để truy xuất dữ liệu trên tham chiếu. Nó được thực hiện như truy vấn bộ sưu tập các đối tượng trong MongoDB cho tham chiếu dựa trên khóa chính của nó làm cho MongoDB tương tự trong hành vi và mục đích của nó để tham gia SQL. Một số vấn đề chính với MongoDB và các cơ sở dữ liệu NoSQL khác là thiếu nhiều tính năng mà các cơ sở dữ liệu quan hệ tiêu chuẩn có. MongoDB chỉ cung cấp các hoạt động nguyên tử trong một tài liệu duy nhất. Một tính năng chính khác mà MongoDB thiếu là hầu hết các hàm tổng hợp; MongoDB không có nhiều hàm tổng hợp đơn giản được xây dựng trong cơ sở dữ liệu như các cơ sở dữ liệu quan hệ tiêu chuẩn. MongoDB có một cách để vượt qua điều này mặc dù và đó là cho phép người dùng để xác định chức năng của họ thông qua phương pháp MapReduce [4]. MapReduce là một sự trừu tượng hóa lập trình được đề xuất trong [10] được sử dụng ngày nay để xử lý dữ liệu lớn. MapReduce bao gồm hai giai đoạn: giai đoạn Bản đồ trong đó người dùng chỉ định tính toán được áp dụng cho tất cả đầu vào và giai đoạn Giảm, trong đó đầu ra từ giai đoạn Bản đồ được tổng hợp bằng cách sử dụng tính toán do người dùng chỉ định khác. Ví dụ cổ điển được sử dụng để minh họa MapReduce là đếm từ, trong đó sự xuất hiện của mỗi từ trong một tài liệu nhất định được tính toán. Trong giai đoạn Bản đồ, đối với mỗi từ gặp phải trong tài liệu, một cặp khóa-giá trị được phát ra bao gồm từ là khóa và 1 là giá trị, ví dụ <xin chào, 1>. Trong giai đoạn Giảm, tất cả các lần xuất hiện của cùng một từ được kết hợp và các giá trị của chúng là ‘1’ được tổng, cho kết quả đếm từ. Trong phần tiếp theo, chúng tôi mô tả các thử nghiệm của chúng tôi để nghiên cứu hiệu suất của SQL và MongoDB xem xét chèn, cập nhật và chọn các hoạt động yêu cầu tham gia và MapReduce.

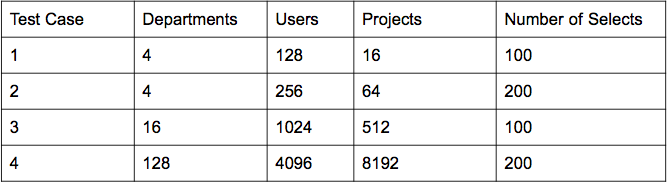
1. **THỰC HÀNH**

Để thiết lập ban đầu, chúng tôi đã cài đặt cả MongoDB và Microsoft SQL Server Express trên máy chạy bộ xử lý Intel i7 quad core 3.4GHz với 8GB DDR3 Ram. Cả hai cơ sở dữ liệu đều được cài đặt và lưu trữ dữ liệu trên SSD (Ổ đĩa thể rắn) để đọc và ghi nhanh nhất có thể. Chúng tôi đã viết cả hai ứng dụng thử nghiệm trong C# sử dụng Visual Studio 2012 dưới tiêu đề. Net 4.0 framework. Đối với ứng dụng SQL Server, chúng tôi đã sử dụng khung thực thể để chèn và liên kết dữ liệu đối tượng, nhưng đối với các bản cập nhật và truy vấn, chúng tôi đã sử dụng SQL thẳng để kết quả sẽ không bị sai lệch. Về phía MongoDB, chúng tôi đã sử dụng phiên bản 1.7 mới nhất của trình điều khiển MongoDB C# và tất cả các truy vấn đều được viết trực tiếp bằng trình soạn thảo truy vấn của MongoDB. Các bài kiểm tra của chúng tôi bao gồm bốn bài kiểm tra riêng biệt với 100 lần chạy cho mỗi bài kiểm tra. Hình 1 cho thấy sơ đồ đối tượng được sử dụng cho các bài kiểm tra. Như thể hiện trong Hình 1, dữ liệu bao gồm 3 bảng/bộ sưu tập đại diện cho bộ phận, dự án và người dùng. Có mối quan hệ giữa các đối tượng dữ liệu này, vì người dùng có một bộ phận cụ thể và một loạt các dự án. Tương tự, một dự án được liên kết với một bộ phận cụ thể, một người quản lý là người dùng và cũng là một mảng người dùng của dự án. Trong việc triển khai mô hình quan hệ của cơ sở dữ liệu trong Hình 1, các truy vấn liên quan đến nhiều hơn một bảng yêu cầu thao tác tham gia. Ví dụ: một truy vấn liên quan đến Người dùng và Bộ phận yêu cầu sự tham gia giữa người dùng và các bảng bộ phận dựa trên giá trị chung cho Bộ phận và id. Trong MongoDB, một truy vấn như vậy yêu cầu sao chép dữ liệu Bộ phận trong bộ sưu tập Người dùng hoặc truy xuất dữ liệu dựa trên khóa. Chúng tôi cũng lưu ý rằng việc triển khai cơ sở dữ liệu trong Hình 1 yêu cầu 3 bảng cho mỗi đối tượng dữ liệu, cũng như một bảng bổ sung thể hiện mối quan hệ M:N giữa Người dùng và các đối tượng Dự án. Bảng bổ sung này thay thế mảng dự án trong bảng Người dùng và mảng người dùng trong bảng Dự án.



Hình 1. Sơ đồ cơ sở dữ liệu

Hình 2 minh họa số lượng các Bộ phận, Người dùng và Dự án cho mỗi trường hợp thử nghiệm trong bốn trường hợp. Tổng số lượng tuples được sử dụng trong các ca kiểm tra dao động từ 148 cho ca kiểm tra 1 đến 12.416 cho ca kiểm tra 4. Tổng số lượng tuples được sử dụng trong các ca kiểm tra dao động từ 148 cho ca kiểm tra 1 đến 12.416 cho ca kiểm tra 4.

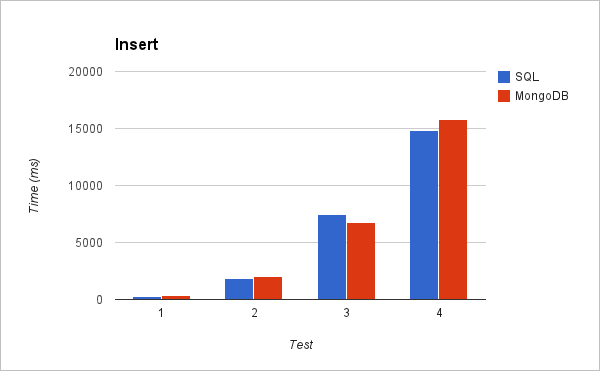


Hình 2. Các ca kiểm tra

Đối với các thử nghiệm, chúng tôi đã xây dựng các ứng dụng bảng điều khiển riêng biệt trong C# cố gắng sử dụng cùng một mã cho cả SQL và MongoDB khi có thể. Mã cho các chèn chỉ khác nhau trong các tài liệu tham khảo của họ cho các cơ sở dữ liệu khác nhau. Các phương pháp để thực hiện cập nhật và truy vấn rất khác nhau giữa hai triển khai. Cơ sở dữ liệu quan hệ sử dụng cú pháp SQL trong khi cơ sở dữ liệu MongoDB sử dụng JavaScript Object Notation (JSON). Các chức năng tổng hợp thậm chí còn khác nhau nhiều hơn giữa hai triển khai do độ phức tạp bổ sung của việc sử dụng các chức năng MapReduce trong MongoDB. Đối với phần Chèn trong thử nghiệm của chúng tôi, chúng tôi đã chèn từng mục dữ liệu như trong Hình 2. Các mối quan hệ giữa các đối tượng dữ liệu, chẳng hạn như Bộ phận trong đối tượng Người dùng được thực hiện trong MongoDB bằng cách lưu trữ một tham chiếu đến các tài liệu khác thay vì lồng ghép toàn bộ tài liệu. Ví dụ: tham chiếu đến một Bộ phận đã được lưu trữ trong đối tượng Người dùng. Đối với phần Cập nhật trong bài kiểm tra của chúng tôi, chúng tôi đã sử dụng ba loại cập nhật khác nhau. Loại đầu tiên liên quan đến việc cập nhật tên của một phần mười các bộ phận dựa trên khóa chính của họ. Loại thứ hai cập nhật họ của bất kỳ người dùng nào có tên trùng với tên đã chọn (tên được chọn cho bài kiểm tra của chúng tôi là Jacob). Loại thứ ba cập nhật tên của một phần tư dự án dựa trên khóa của chúng. Chúng tôi đã thử nghiệm từng trường hợp trong số ba loại cập nhật này trên cả bốn trường hợp thử nghiệm. Chúng tôi cũng đã thử nghiệm bảy truy vấn Select khác nhau. Các lựa chọn được chia thành hai loại: đơn giản và phức tạp. Các truy vấn đơn giản chỉ liên quan đến việc chọn dữ liệu của một loại đối tượng. Các truy vấn phức tạp liên quan đến nhiều loại đối tượng, các truy vấn lồng nhau và các hàm tổng hợp. Số lần chạy lựa chọn cho các ca kiểm tra 1 và 3 là 100 và số lần chạy lựa chọn cho các ca kiểm tra 2 và 4 là 200. Bốn lựa chọn đơn giản khác nhau mà chúng tôi đã tạo ra cho các thử nghiệm như sau. Lựa chọn đầu tiên được lấy từ một bộ phận bằng khóa chính của nó. Thứ hai là truy xuất một bộ phận bằng một tên được chọn ngẫu nhiên. Tất cả các phòng ban đều có tên duy nhất và tên được đảm bảo khớp với một kết quả trong bộ sưu tập của phòng ban. Lựa chọn thứ ba lấy lại người dùng bằng khóa chính của nó. Lựa chọn thứ tư truy xuất tất cả người dùng có tên trùng với tên được chọn ngẫu nhiên; tên được chọn ngẫu nhiên này có thể không khớp với bất kỳ người dùng nào trong bộ sưu tập. Ba lựa chọn phức tạp khác nhau được sử dụng trong các thử nghiệm liên quan đến hoạt động tham gia và như sau. Lựa chọn phức tạp đầu tiên đã truy xuất tất cả các bộ phận có chứa một hoặc nhiều người dùng có tên được chọn ngẫu nhiên. Vì tên đầu tiên được chỉ định ngẫu nhiên, điều này có thể hoặc không thể trả lại bất kỳ kết quả phù hợp nào. Lựa chọn phức tạp thứ hai đã truy xuất tất cả người dùng làm việc trên bất kỳ dự án nào trong ba dự án được chọn ngẫu nhiên. Vì người dùng được chỉ định ngẫu nhiên cho các dự án, truy vấn này cũng có thể không trả về bất kỳ kết quả phù hợp hợp hợp lệ nào. Lựa chọn phức hợp cuối cùng trả về độ tuổi trung bình của người dùng làm việc trên bất kỳ dự án nào trong số ba dự án được chọn ngẫu nhiên, và rõ ràng, chứa trung bình hàm tổng hợp (). Truy vấn cuối cùng này có thể trở về 0 nếu không có người dùng nào được chỉ định cho các dự án.

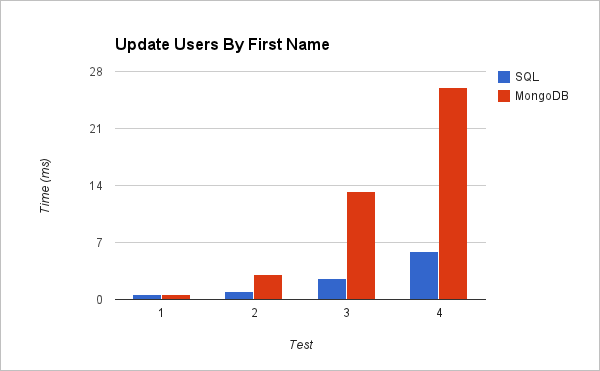
1. **KẾT QUẢ**

Kết quả của tất cả các thí nghiệm của chúng tôi xuất hiện trong Hình 3 đến 13. Mỗi số liệu so sánh thời gian trung bình, tính bằng mili giây, của việc triển khai SQL so với triển khai MongoDB của công việc đã cho cho cho mỗi trong số 4 trường hợp Kiểm tra. 13. Mỗi số liệu so sánh thời gian trung bình, tính bằng mili giây, của việc triển khai SQL so với triển khai MongoDB của công việc đã cho cho cho mỗi trong số 4 trường hợp Kiểm tra.

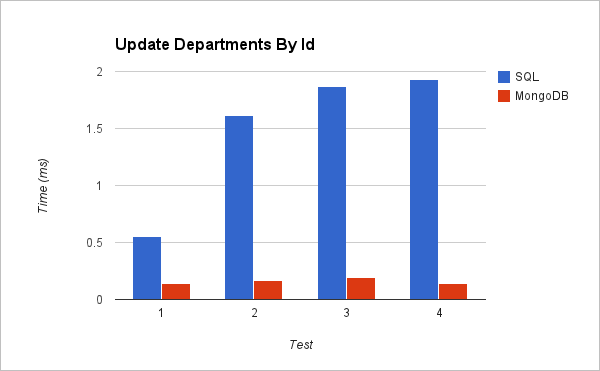


Hình 3. Chèn tốc độ

Hình 3 cho thấy sự khác biệt về tốc độ chèn giữa hai hệ thống cơ sở dữ liệu. Không có cơ sở dữ liệu nào thực hiện tốt hơn cơ sở dữ liệu còn lại. SQL nhanh hơn MongoDB trong Test Case 1, 2 và 4, trong khi MongoDB nhanh hơn SQL trong Test Case 3. Như đã lưu ý trước đó, các chèn trong MongoDB đã thực hiện các mối quan hệ giữa các đối tượng dữ liệu bằng cách sử dụng tham chiếu. Điều này là không yêu cầu chèn dữ liệu bổ sung trong MongoDB. MongoDB chậm hơn 34% đối với Test Case 1 so với SQL và chậm hơn 7% đối với Test Case 2 và 4, trong khi SQL chậm hơn 10% đối với Test Case 3 so với MongoDB. Hình 4, 5 và 6 trình bày thời gian tính bằng ms cho mỗi bản cập nhật khác nhau mà chúng tôi đã chạy. Đối với bản cập nhật trong Hình 4, chúng tôi đã thực hiện cập nhật cho người dùng dựa trên tên và tên không được lập chỉ mục theo bất kỳ cách nào. Trong Hình 4, MongoDB thực hiện tính nhất quán kém hơn SQL Server cho mỗi bốn Test Case. Vì Hình 4 là bản cập nhật trên một cột không được lập chỉ mục và bản chất phi cấu trúc của MongoDB buộc bất kỳ truy vấn không lập chỉ mục nào phải thực hiện tra cứu phức tạp trên mỗi mục dữ liệu, nên MongoDB mất nhiều thời gian hơn để chạy các bản cập nhật dữ liệu lớn hơn SQL. Đối với Test Case 1, MySQL và MongoDB có hiệu năng gần giống nhau. Khi kích thước của cơ sở dữ liệu tăng lên, thời gian để MongoDB cập nhật Người dùng bằng Tên của họ chậm hơn từ 3 lần so với SQL cho Test Case 2, đến hơn 5 lần đối với MongoDB cho Test Case 3.

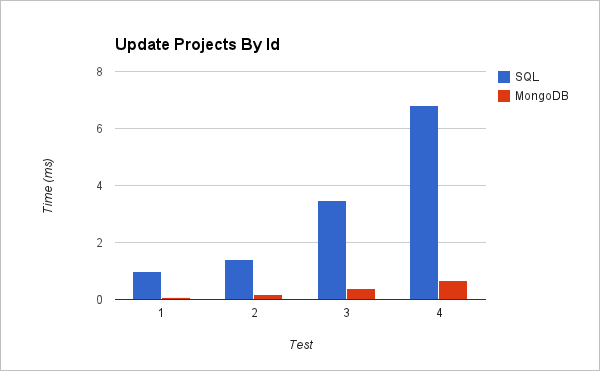


Hình 4. Cập nhật - Tên



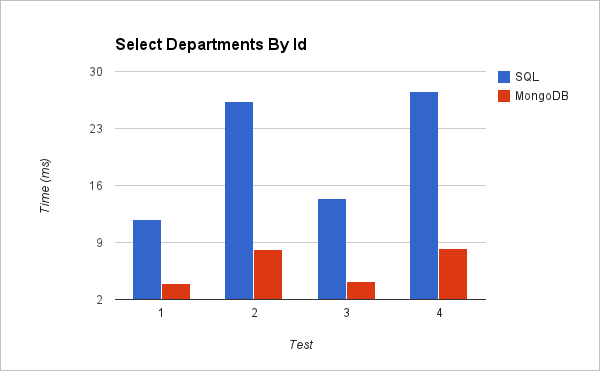
Hình 5. Cập nhật - ID phòng ban

MongoDB vượt trội SQL trong tất cả các trường hợp mà bản cập nhật liên quan đến việc sử dụng khóa chính, như thể hiện trong Hình 5 và 6. Bản cập nhật trong Hình 5 liên quan đến việc cập nhật Bộ phận bằng ID bộ phận của nó, trong khi bản cập nhật trong Hình 6 liên quan đến việc cập nhật Dự án bằng ID dự án. Sự thay đổi về thời gian cập nhật là nhỏ trong tất cả các ca kiểm tra, đặc biệt là đối với MongoDB, nhỏ hơn 0,2 ms đối với tất cả bốn ca kiểm tra. SQL Server dao động từ mức thấp 0,55 ms đối với Test Case 1 đến 1,93 ms đối với Test Case 2. Trong Hình 6, SQL Server chậm hơn 13 lần so với MongoDB cho Test Case 4 khi cập nhật theo Department ID. Chúng tôi tin rằng khoảng cách hiệu suất này là do MongoDB có một chỉ mục được xây dựng sẵn trên khóa chính của tài liệu nhanh hơn chỉ mục cụm khóa chính của SQL Server.

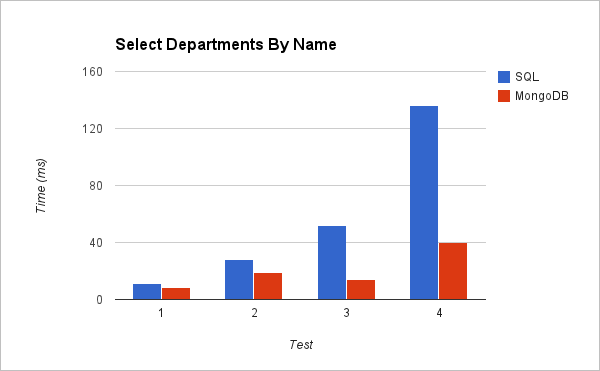


Hình 6. Cập nhật - ID dự án

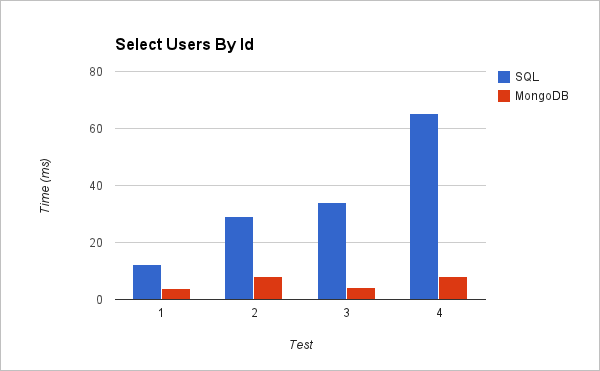
Hình 7 đến 13 minh họa thời gian thực hiện các truy vấn được chọn, với Hình 7-10 hiển thị kết quả cho các truy vấn đơn giản và Hình 11-13 hiển thị kết quả cho các truy vấn phức tạp. Các lựa chọn đơn giản nằm trên khóa chính của bảng (tài liệu) hoặc một trong các giá trị khác. Phức hợp chọn các mối quan hệ liên quan đến nhóm, M:N và tập hợp. Chúng tôi lưu ý rằng mặc dù kích thước của các bảng tăng lên đối với các ca kiểm tra 1-4, mỗi bài kiểm tra có một số lượng lựa chọn thay đổi. Test Case 2 có số lần chọn cao hơn Test 1, trong khi Test Case 3 có số lần chọn tương tự như Test 1, nhưng có nhiều dữ liệu hơn. Tương tự, các lựa chọn cho Test Case 2 và 4 là giống nhau, nhưng Test Case 4 có nhiều dữ liệu hơn. Do đó, không phải lúc nào cũng hữu ích để so sánh thời gian chạy của các bài kiểm tra với nhau, mà thay vào đó là so sánh thời gian chạy của MongoDB với SQL. Trong Hình 7, thời gian để chọn các Bộ phận theo ID là khoảng cao hơn 3,5 lần đối với SQL so với MySQL cho tất cả các Test Case. Mặc dù kích thước của dữ liệu tăng từ các Trường hợp Thử nghiệm 1 lên 4, thời gian chạy cho thử nghiệm này cũng bị ảnh hưởng bởi số lượng lựa chọn, như được chứng minh bởi sự giảm thời gian chạy cho Trường hợp 3 so với Trường hợp 2. Hình 8 minh họa thời gian để chọn các Bộ phận theo Tên. SQL chậm hơn 1,5 lần so với MongoDB cho Test Case 1 và 2, và chậm hơn hơn 3 lần so với MongoDB cho Test Case 3 và 4. Có một xu hướng tương tự trong Hình 9, trong đó các Khoa được chọn theo Tên, nhưng nó kịch tính hơn. MongoDB nhanh hơn SQL 3 lần đối với Test Case 1 và 2 và nhanh hơn 8 lần đối với Test Case 3 và 4. Điều đáng chú ý là trong Hình 8 và 9, thời gian chạy cho SQL tăng lên khi lượng dữ liệu tăng lên, trong khi thời gian chạy cho MongoDB có liên quan đến lượng dữ liệu cũng như số lượng các lựa chọn được thực hiện. Các ca kiểm tra 1 và 3 có 100 lựa chọn, trong khi các ca kiểm tra 2 và 4 có 200 lựa chọn, và có thời gian chạy cao hơn.



Hình 7. Chọn - ID phòng ban

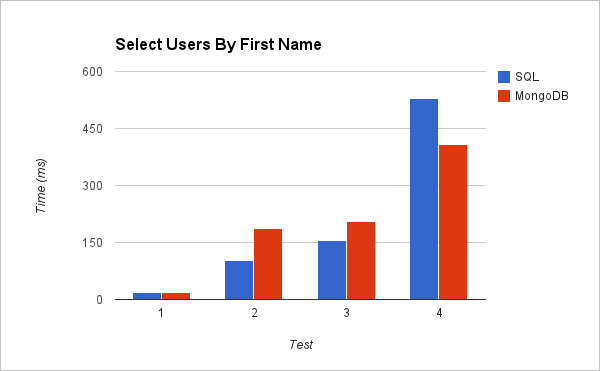


Hình 8. Chọn - Tên bộ phận



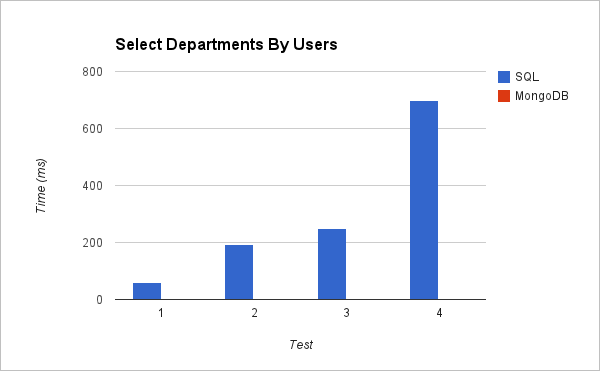
Hình 9. Chọn - ID người dùng

Truy vấn trong Hình 10 liên quan đến việc chọn Người dùng dựa trên Tên của họ. Các kết quả trong Hình 10 phù hợp với các kết quả trong Hình 4 cho các ca bệnh thử nghiệm 1-3, trong đó Người dùng được cập nhật dựa trên Tên của họ. SQL có thời gian chạy nhanh hơn MongoDB cho Test cases 1-3 vì nó có thể chọn thuộc tính cá nhân trong bảng quan hệ dễ dàng hơn so với việc định vị giá trị trong bộ sưu tập của MongoDB. Tuy nhiên, khi dữ liệu phát triển đủ lớn trong Test Case 4, thời gian chạy của SQL cao hơn 23% so với thời gian chạy của MongoDB. Thời gian xử lý SQL của truy vấn này trở nên chậm hơn so với tra cứu của MongoDB khi lượng dữ liệu tăng lên. Hình 7-9 minh họa rằng MongoDB vượt trội SQL trong tất cả các bài kiểm tra với một số lượng lớn. Chúng tôi tin rằng điều này là do sự kết hợp của chỉ mục được sử dụng bởi MongoDB và việc sử dụng bộ nhớ của nó. MongoDB sử dụng các tệp được ánh xạ bộ nhớ để lưu trữ tất cả các tài liệu của nó trong bộ nhớ thay vì trên đĩa (miễn là hệ thống có sẵn bộ nhớ miễn phí). Vì SQL Server phải lấy tất cả dữ liệu từ đĩa và tốc độ đĩa chậm hơn thời gian tìm nạp bộ nhớ, MongoDB vượt trội SQL.

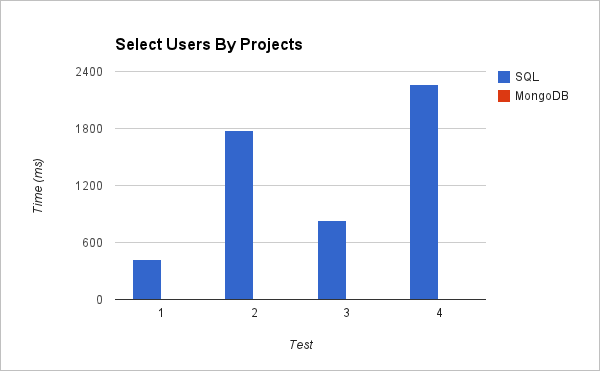


Hình 10. Chọn - Tên người dùng

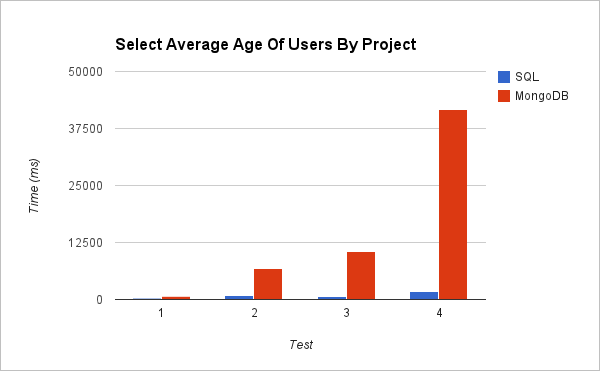
Đối với các truy vấn phức tạp hơn trong Hình 11 và 12, MongoDB có thời gian chạy nhanh hơn đáng kể so với SQL. Thời gian chạy cho MongoDB thấp đến mức không phải lúc nào cũng có thể nhìn thấy trong các số liệu. Thời gian chạy cho MongoDB trong Hình 10 dao động từ 0,13 đối với ca kiểm tra 1 đến 0,19 đối với ca kiểm tra 2. Thời gian chạy đối với MongoDB trong Hình 11 nằm trong khoảng từ 0,24 đối với Trường hợp Thử nghiệm 1 đến 0,42 đối với Trường hợp Thử nghiệm 4. Như thể hiện trong Hình 13, MongoDB hoạt động tốt trên các truy vấn phức tạp, ngoại trừ khi sử dụng các chức năng tổng hợp. Trong Hình 13, đối với Test Case 4, SQL có thời gian chạy 1.836 ms trong khi MongoDB có thời gian chạy 41.754ms. MongoDB chậm hơn SQL gần 23 lần. Chúng tôi tin rằng điều này là do MongoDB không có các hàm tổng hợp thực sự được xác định cho hầu hết các phương pháp, chẳng hạn như hàm SQL Average. Vì MongoDB phải sử dụng phương pháp MapReduce để cho phép nhà phát triển tạo các chức năng tổng hợp của riêng họ, điều này làm chậm hiệu suất một lượng đáng kể. Không giống như sử dụng SQL, các quyết định triển khai bổ sung được yêu cầu có ảnh hưởng đến hiệu suất của MongoDB. Sao chép dữ liệu trong mỗi bộ sưu tập thay vì một tham chiếu có thể dẫn đến hiệu suất nhanh hơn, mặc dù nó sẽ đòi hỏi nhiều bộ nhớ hơn. Ngoài việc xác định cách thực hiện các mối quan hệ, việc thực hiện các chức năng tổng hợp cũng cần thiết.



Hình 11. Chọn phòng ban theo người dùng



Hình 12. Chọn người dùng theo dự án



Hình 13. Chọn độ tuổi trung bình của người dùng theo dự án

1. **KẾT LUẬN VÀ CÔNG VIỆC TƯƠNG LAI**

Nhìn chung, khi so sánh SQL với MongoDB, MongoDB có hiệu suất chạy tốt hơn cho chèn, cập nhật và truy vấn đơn giản. SQL hoạt động tốt hơn khi cập nhật và truy vấn các thuộc tính không phải khóa, cũng như cho các truy vấn tổng hợp. MongoDB có thể là một giải pháp tốt cho các tập dữ liệu lớn hơn trong đó sơ đồ liên tục thay đổi hoặc trong trường hợp các truy vấn được thực hiện sẽ ít phức tạp hơn. Vì MongoDB không có lược đồ thực sự được xác định và SQL yêu cầu một định nghĩa lược đồ cứng, MongoDB sẽ dễ dàng xử lý một lược đồ động như hệ thống quản lý tài liệu với một số trường động và chỉ một số trường tìm kiếm được biết đến. Kết luận lại, MongoDB chắc chắn là sự lựa chọn cho những người dùng cần một cấu trúc cơ sở dữ liệu ít cứng nhắc hơn. MongoDB có thể là một giải pháp tốt cho các tập dữ liệu lớn hơn trong đó sơ đồ liên tục thay đổi hoặc trong trường hợp các truy vấn được thực hiện sẽ ít phức tạp hơn. Đối với những người dùng có lược đồ chặt chẽ được xác định và số lượng dữ liệu có cấu trúc khiêm tốn, chúng tôi cũng nhận thấy MongoDB hoạt động tốt hơn SQL nói chung. Tuy nhiên, có những nhược điểm đối với MongoDB, chẳng hạn như hiệu suất kém của nó đối với các chức năng tổng hợp và truy vấn dựa trên các giá trị không phải là khóa. Tuy nhiên, có những nhược điểm đối với MongoDB, chẳng hạn như hiệu suất kém của nó đối với các chức năng tổng hợp và truy vấn dựa trên các giá trị không phải là khóa. Cuối cùng, SQL là tiêu chuẩn công nghiệp và được hỗ trợ rộng rãi hơn nhiều so với MongoDB. Một điều chúng tôi muốn thấy là công việc trong tương lai từ bài báo này sẽ chạy MongoDB và SQL như một cơ sở dữ liệu phân tán. MongoDB được biết đến là hoạt động tốt nhất như một cơ sở dữ liệu phân tán vì vậy hiệu suất cho các truy vấn phức tạp của nó nên tăng lên khi nó được sử dụng theo cách này. Các công việc khác trong tương lai sẽ liên quan đến việc chạy các thí nghiệm tương tự trên cả MongoDB và SQL với một lược đồ lớn hơn và phức tạp hơn nhiều. Về lý thuyết MongoDB nên vượt trội hơn SQL do nó không yêu cầu một giản đồ thực sự để được định nghĩa. Vì cơ sở dữ liệu SQL quan hệ có chi phí đáng kể và yêu cầu bổ sung tham gia trong một giản đồ phức tạp hơn, chúng tôi giả thuyết rằng hiệu suất của SQL sẽ tiếp tục giảm so với MongoDB.

1. **TÀI LIỆU THAM KHẢO**
2. Elmasri, R. and Navathe, S. 2011. *Fundamentals of Database Systems*, Addison-Wesley.
3. Strauch, C. NoSQL Databases. *Selected Topics on Software-Technology*, Stuttgart Media University.
4. NoSQL. <http://nosql-database.org/>
5. MongoDB. <http://www.mongodb.org/>
6. BSON. <http://bsonspec.org/>
7. Hecht, R. and Jablinski, S. 2011. NoSQL Evaluation A Use Case Oriented Survey. *Proceedings International Conference on Cloud and Service Computing*, pp. 12-14.
8. Han, J. 2011. Survey on NOSQL Databases, *Proceedings 6th International Conference on Pervasive Computing and Applications*, pp. 363-366.
9. Leavitt, N. 2010. Will NoSQL Databases Live up to Their Promise?. Computer Magazine, Vol. 43 No. 2, pp. 12-14.
10. Stonebreaker, M. 2010. SQL Databases v. NoSQL Databases. *Communications of the ACM*, Vol. 25, No. 4, pp. 10-11.

Dean, J. and Ghemawat, S. 2004. MapReduce: Simplified Data Processing on Large Clusters. *Proceedings Symposium on Operating Systems*