**Comparing NoSQL MongoDB to an SQL DB**

**(So sánh NoSQL MongoDB với SQL DB)**

**ABSTRACT ( Lớp trừu tượng )**

Các giải pháp của cơ sở dữ liệu NoSQL ngày càng trở nên phổ biến nhiều hơn trong một thế giới hiện đang bị chi phối bởi quan hệ cơ sở dữ liệu SQL . Cơ sở dữ liệu NoSQL được thiết kế để cung cấp các giải pháp cho khối lượng lớn dữ liệu không được cấu trúc. Tuy nhiên, những ưu điểm (hoặc nhược điểm) của việc sử dụng cơ sở dữ liệu NoSQL cho dữ liệu được cấu trúc và không nhất thiết phải là "Lớn", là không rõ ràng. Không có nhiều nghiên cứu so sánh hiệu suất xử lý một lượng dữ liệu có cấu trúc khiêm tốn 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 đối với cơ sở dữ liệu có cấu trúc có kích thước vừa phải. Kết quả cho thấy rằng MongoDB hoạt động tốt hoặc tốt hơn cơ sở dữ liệu quan hệ, trừ khi các chức năng tổng hợp được sử dụng.

Danh mục và mô tả chủ đề

H.2.4 [Hệ thống thông tintems]: Hệ thống – cơ sở dữ liệu quan hệ

Điều khoản chung

Perfor mance

từ khóa

MongoDB, NoSQL, hiệu suất , quan hệ, có cấu trúc dữ liệu,

SQ

Danh mục và mô tả chủ đề

H.2.4 [Hệ thống thông tintems]: Hệ thống – cơ sở dữ liệu quan hệ

Điều khoản chung

Perfor mance

từ khóa

MongoDB, NoSQL, hiệu suất , quan hệ, có cấu trúc dữ liệu,

SQ

Danh mục và mô tả chủ đề

H.2.4 [Hệ thống thông tintems]: Hệ thống – cơ sở dữ liệu quan hệ

Điều khoản chung

Perfor mance

từ khóa

MongoDB, NoSQL, hiệu suất , quan hệ, có cấu trúc dữ liệu,

SQL

Danh mục và mô tả chủ đề

H.2.4 [Hệ thống thông tintems]: Hệ thống – cơ sở dữ liệu quan hệ

Điều khoản chung

Perfor mance

từ khóa

MongoDB, NoSQL, hiệu suất , quan hệ, có cấu trúc dữ liệu,

SQL

**Danh mục và mô tả chủ đề**

H.2.4 [Hệ thống thông tintems]: Hệ thống – cơ sở dữ liệu quan hệ

**Điều khoản chung**

Perfor mance

**Từ khóa**

MongoDB, NoSQL, hiệu suất, quan hệ, cấu trúc dữ liệu

MongoDB, NoSQL, hiệu suất , quan hệ, có cấu trúc dữ liệu,

SQL

MongoDB, NoSQL, hiệu suất , quan hệ, có cấu trúc dữ liệu,

SQ

Điều khoản chung

Perfor mance

Điều khoản chung

Perfor mance

**1. INTRODUCTION ( Giới Thiệu )**

Việc triển khai cơ sở dữ liệu phổ biến nhất hiện nay là dựa trên mô hình quan hệ sử dụng SQL làm ngôn ngữ truy vấn của nó. Tuy nhiên, NoSQL (Not Only SQL- phi quan hệ) là giải pháp cơ sở dữ liệu bắt đầu trở nên nổi bật hơn với một số lượng lớn nhanh chóng phát triển dữ liệu đang được thu thập hôm nay. Dữ liệu này thường không có cấu trúc, phức tạp và không phù hợp với mô hình quan hệ. Ví dụ về loại dữ liệu này là bản ghi chép điện thoại thông minh trong vị trí được phát cứ sau vài giây, ghi lại từ máy video trong không gian công cộng và một số những thứ to lớn từ một số trang và một số tài liệu trên web. Tuy nhiên, có những thứ tồn tại một số lượng lớn của cơ sở dữ liệu có kích thước-khiêm tốn với dữ liệu có cấu trúc rằng vẫn cần được lưu trữ và cần được xử lý trong cơ sở dữ liệu.

Hiện nay có một chút thảo luận về trong thế giới cơ sở dữ liệu hoặc không phải NoSQL(phi quan hệ) cơ sở dữ liệu sẽ thay thế cơ sở dữ liệu quan hệ. Với việc di chuyển hướng tới NoSQL(phi quan hệ) cơ sở dữ liệu và tính khả dụng mã nguồn mở của NoSQL(phi quan hệ) cơ sở dữ liệu, câu hỏi một cách tự nhiên cho một nhà thiết kế cơ sở dữ liệu là có nên chọn cơ sở dữ liệu NoSQL(phi quan hệ) hay không. Nếu cơ sở dữ liệu không có cấu trúc và vô cùng lớn, thì sau đó NoSQL(phi quan hệ) cơ sở dữ liệu là một lựa chọn tốt. Tuy nhiên, nếu nó ko sử dụng một cách rõ ràng NoSQL(phi quan hệ) cơ sở dữ liệu 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 cấu trúc dữ liệu.

Dữ liệu trong mô hình quan hệ được thường xuyên đại diện bởi một lược đồ cơ sở dữ liệu[1] , để nắm giữ ngữ nghĩa của cơ sở dữ liệu. Những đối tượng trong cơ sở dữ liệu với một số đặc điểm như nhau, loại và định dạng được tập hợp lại cùng nhau, tạo thành cấu trúc dữ liệu. Mô hình quan hệ được xây dựng trên giả thuyết của cấu trúc dữ liệu, với dữ liệu của nó được lưu trữ trong các hàng và các cột của một bảng, theo đó mỗi hàng có cùng số lượng và loại của dữ liệu cột.

Các bảng trong cơ sở dữ liệu quan hệ thường được chuẩn hóa cái mà dẫn đến sáng tạo ra nhiều bảng. Truy vấn những bảng đó yêu cầu tìm nạp và kết hợp thông tin từ nhiều các bảng khác nhau. Kết hợp thông tin dựa trên sự phù hợp giá trị cho khóa chính và khóa ngoại trên nhiều bảng trong mô hình quan hệ yêu cầu sử dụng tham gia một hoạt động. Lược đồ càng lớn và càng nhiều bảng cần được nối, cơ sở dữ liệu quan hệ càng mất nhiều thời gian để tìm nạp dữ liệu.

NoSQL(phi quan hệ) có thể giúp xử lý dữ liệu không có cấu trúc. Dữ liệu có thể ở dạng bán cấu trúc, sao cho các đối tượng dữ liệu tương tự có thể được tập hợp lại cùng nhau, nhưng những đối tượng có thể có các đặc điểm khác nhau, thông tin lược đồ cũng có thể được trộn lẫn với các giá trị dữ liệu trong

dữ liệu bán cấu trúc, như là tìm thấy trong dữ liệu XML. Phi cấu trúc dữ liệu có thể thuộc bất kỳ loại nào và có thể không có định dạng. Dữ liệu này không thể được đại diện bởi bất kỳ loại lược đồ nào, như trang web trong HTML.

Các đặc điểm điển hình của cơ sở dữ liệu SQL, chẳng hạn như các đặc tính ACID, yêu cầu một lượng chi phí nhất định, bảo trì hoặc loại bỏ trong cơ sở dữ liệu NoSQL để đạt hiệu suất tối đa. Nhiều cơ sở dữ liệu NoSQL sắp xếp 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, số hoặc cấu trúc phức tạp với ngữ nghĩa độc nhất. Sự phát triển của các truy vấn nhiều tổ hợp hơn, không có ngôn ngữ truy vấn tiêu chuẩn, và có giới hạn cho các hoạt động. Đặc biệt, không có hoạt động tham gia. Đặc biệt, nhìn chung quá trình xử lý đơn giản hơn giá cả phải chăng hơn và linh hoạt hơn.

Trong bài viết này, chúng tôi so sánh một trong những giải pháp NoSQL, MongoDB [4,5], tới tiêu chuẩn SQL cơ sở dữ liệu quan hệ, SQL Server. Chúng tôi chạy thử nghiệm bằng cách sử dụng cấu trúc có kích thước khiêm tốn cơ sở dữ liệu để xác định hiệu suất của các cơ sở dữ liệu quan hệ sang cơ sở dữ liệu NoSQL(phi quan hệ). Chúng tôi lưu ý rằng chúng tôi không xem xét bất kỳ thuộc tính ACID nào trong nghiên cứu của mình. 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 lựa chọn tốc độ hoạt động. Kết quả từ các thử nghiệm của chúng tôi cung cấp thông tin chi tiết về khả năng sử dụng cơ sở dữ liệu NoSQL(phi quan hệ) cho dữ liệu có cấu trúc có kích thước khiêm tốn.

Bài viết này được tổ chức như sau. Trong phần 2 chúng ta thảo luận RELATED WORK(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 thí nghiệm của chúng tôi. Kết luận và công việc trong tương lai xuất hiện trong Phần 5.

Quyền tạo kỹ thuật số hoặc bản cứng của tất cả hoặc một phần của công việc này cho bản thân hoặc các lớp học được cấp để sử dụng miễn phí với điều kiện là các bản sao không đc tạo ra hoặc phân phối vì lợi nhuận hoặc lợi thế thương mại và các bản sao đó phải chịu thông báo này và trích dẫn đầy đủ trên trang đầu tiên. Để sao chép, tái xuất bản, đăng trên máy chủ hoặc phấn phối lại vào danh sách, phải yêu cầu trước mới có sự cho phép cụ thể và/hoặc một khoản phí.

**2. RELATED WORK( Công việc liên quan )**

Có rất nhiều blog, bài viết và bình luận trực tuyến về lợi ích của cơ sở dữ liệu NoSQL(phi quan hệ) với cơ sở dữ liệu SQL, nhưng ít tài liệu lý thuyết so sánh NoSQL với SQL. Thay vào đó, gần đây đã có một số bài báo đánh giá các cơ sở dữ liệu NoSQL khác nhau có sẵn [6-8]. Trong [6] các tác giả thực hiện một trường hợp nghiên cứu của 14 cơ sở dữ liệu NoSQL(phi quan hệ) khác nhau qua các tính năng như mô hình dữ liệu của họ.

Cơ sở dữ liệu NoSQL(phi quan hệ) được so sánh với cơ sở dữ liệu SQL trong [9]. Chi phí phát sinh từ quá trình xử lý giao dịch trực tuyến nhằm mục đích không liên quan đến SQL, nhưng thay vào đó là các thành phần khác. Tác giả xem xét bốn thành phần chi phí đăng nhập, khoá, quản lý bộ đệm. Loại bỏ các chi phí liên quan đến một hoặc nhiều hơn trong số này có thể cung cấp tốc độ tăng gấp đôi. 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 NoSQL triển khai MonggoDB, không có lược đồ cơ sở dữ liệu hoặc bảng. Thay vào đó, MongoDB [4,5] sử dụng một "bộ sưu tập" tương tự như bảng và "tài liệu" tương tự như tương tự như các hàng, để lưu trữ dữ liệu và thông tin lược đồ. MongoDB tự động tạo khóa chính (ID) để nhận dạng duy nhất từng tài liệu. ID và tài liệu là mặt khái niệm tương tự như một cặp khóa-giá trị. MongoDB cố gắng giữ hầu hết dữ liệu trong bộ nhớ để các truy vấn đơn giản mất ít thời gian hơn bằng cách loại bỏ nhu cầu truy xuất dữ liệu từ đĩa cứng. Một sự phản đối cho điều này là khi tập dữ liệu trở nên lớn hơn bộ nhớ khả dụng, thì MongoDB sẽ phải bắt đầu truy vấn đĩa cứng cho kết quả. Khi MongoDB là một triển khai NoSQL, nó có khả năng mở rộng cao và không yêu cầu định nghĩa lược đồ cứng. Điều này thỉnh thoảng 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 thao tác tham gia hoạt động trong MongoDB. Lưu trữ dữ liệu trong MongoDB có thể được thực hiện theo một trong hai cách [4]. Cách một là lồng các tài liệu vào trong nhau. Tùy chọn này có thể hoạt động cho các mối quan hệ một-một hoặc 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 các vòng lặp lồng nhau vô hạn. Cách hai là ưu trữ một tham chiếu đến tài liệu khác thay vì lồng toàn bộ tài liệu. Với cách hai này MongoDB sẽ chỉ cần truy xuất tài liệu được tham chiếu khi người dùng yêu cầu dữ liệu bên trong tài liệu được tham chiếu. Nhược điểm của tính năng này là MongoDB không có cách tích hợp để truy xuất một đối tượng dựa trên tham chiếu. Điều này có nghĩa là người dùng phải xác định phương thức riêng của họ để truy xuất dữ liệu khi tham chiếu. Nó được triển khai dưới dạng truy vấn bộ sưu tập các đối tượng trong MongoDB đối với tham chiếu dựa trên khóa chính của nó làm cho MongoDB có hành vi và mục đích tương tự như một phép nối SQL.

Một số vấn đề chính với MongoDB và cơ sở dữ liệu NoSQL khác là thiếu nhiều tính năng quan hệ tiêu chuẩn cơ sở dữ liệu hiện có. MongoDB chỉ cung cấp các hoạt động nguyên tử trong một tài liệu. Một số tính năng chính khác của MongoDB thiếu là chức năng tổng hợp nhất; MongoDB không có nhiều chức năng tổng hợp đơn giản được xây dựng giống như cơ sở dữ liệu quan hệ. MongoDB có một cách để giải quyết xung quanh vấn đề này và điều đó cho phép người sử dụng xác định chức năng thông qua phương pháp MapReduce [4].

MapReduce là một chương trình trừu tượng được đề xuất trong [10] được sử dụng ngày nay để xử lý dữ liệu lớn. MapReduce bao gồm hai giai đoạn:

Giai đoạn Map là trong đó người sử dụng chỉ định tính toán được áp dụng trên tất cả đầu vào và giai đoạn Reduce, trong đó đầu ra từ giai đoạn Map được tổng hợp bằng cách sử dụng tính toán do người sử dụng chỉ định khác. Ví dụ cổ điển dùng để minh họa MapReduce là đếm từ, trong đó sự xuất hiện của mỗi từ trong một tài liệu đã được tính toán. Trong giai đoạn Map, với mỗi từ đã gặp trong tài liệu, một cặp khóa-giá trị được phát ra bao gồm từ khóa và 1 giá trị, ví dụ <hello, 1>. Trong giai đoạn Reduce, tất cả các lần xuất hiện của cùng một từ được kết hợp và các giá trị '1' của chúng được cộng lại, cho kết quả là số lượng từ.

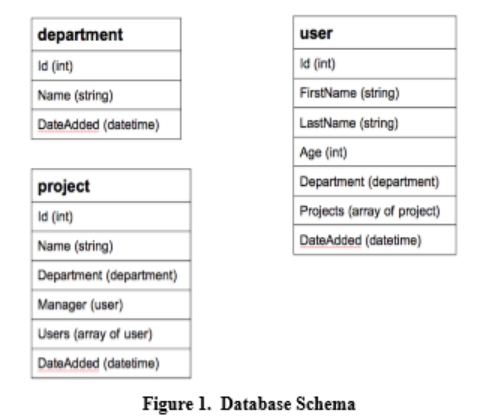
Trong phần tiếp theo, chúng tôi mô tả các thí nghiệm nghiên cứu của chúng tôi về hiệu suất của SQL và MongoDB khi xem xét các phần chèn, cập nhật và lựa chọn các hoạt động tham gia yêu cầu và MapReduce.

**3.** **EXPERIMENTS** **( Cuộc thí nghiệm )**

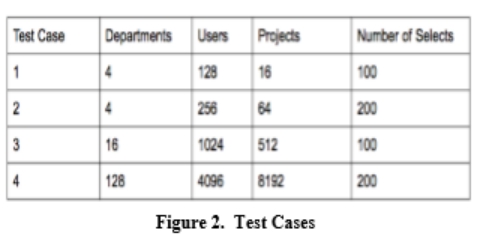
Trong phần thiết lập ban đầu, chúng tôi đã cài đặt cả MongoDB và Microsoft SQL Server Express trên một hệ điều hành được hỗ trợ với bộ xử lý Intel i7 , tất cả bốn lõi trong bộ xử lý Intel i7 này đều chạy với 3.4GHz và 8GB RAM DDR3. Cả hai cơ sở dữ liệu đều được cài đặt và lưu trữ dữ liệu của chúng trên ổ **SSD (Ổ đĩa bán dẫn)** để đọc và ghi nhanh nhất có thể. Chúng tôi đã viết cả hai ứng dụng thử nghiệm trong **C#** bằng cách sử dụng **Visual Studio 2012** dưới **.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 bản truy vấn, chúng tôi đã sử dụng SQL để kết quả sẽ không bị sai lệch. Về phía MongoDB, chúng tôi đã sử dụng phiên bản mới nhất 1.7 của trình điều khiển **MongoDB C#** và tất cả các truy vấn được viết trực tiếp bằng trình ghi truy vấn của MongoDB.

Thử nghiệm của chúng tôi bao gồm bốn thử nghiệm riêng biệt với 100 lần chạy cho mỗi thử nghiệm. Hình 1 cho thấy lược đồ đối tượng được sử dụng cho các bài kiểm tra. Như thể hiện trong Hình 1, dữ liệu bao gồm 3 bảng / collections đại diện cho bộ phận ( Department) , dự án (Project) và người dùng (User). Có những mối liên hệ giữa các đối tượng dữ liệu này, như người dùng ( User) có một bộ phận (Department) cụ thể và các mảng dự án (Projects). Tương tự, một dự án được liên kết với một bộ phận (Department) cụ thể, nhà quản lý là người dùng và cũng là người dùng của các mảng dự án.

Trong quá trình 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 nối. Ví dụ: một truy vấn liên quan đến Người dùng (User) và Bộ phận (Department) yêu cầu kết nối giữa bảng người dùng (user) và bảng bộ phận (department) dựa trên giá trị chung cho bộ phận (Department) và id. Trong MongoDB, một truy vấn như vậy yêu cầu sao chép dữ liệu của bộ phận (Department) trong bộ sưu tập Người dùng (User) hoặc truy xuất dữ liệu dựa trên khóa (key) . 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 đại diện cho mối quan hệ M: N giữa các đối tượng người dùng (Users) và dự án (Project). Bảng bổ sung này thay thế mảng dự án trong bảng Người dùng (User) và mảng người dùng trong bảng Dự án (Project) .



Hình 2 minh họa số lượng Phòng ban (Departments), Người dùng (Users) và Dự án (Projects) đối với mỗi trường hợp trong số bốn trường hợp thử nghiệm. Tổng số bộ được sử dụng trong các trường hợp kiểm tra dao động từ 148 cho trường hợp kiểm tra 1 đến 12,416 cho trường hợp kiểm tra 4. Số lượng lựa chọn sẽ được thảo luận sau.



Đối với các thử nghiệm, chúng tôi đã tạo bảng ứng dụng điều khiển riêng biệt trong C# để cố gắng sử dụng cùng một mã cho cả SQL và MongoDB khi khả thi. Mã cho các phần chèn chỉ khác nhau trong bộ tài liệu tham khảo của chúng đến các cơ sở dữ liệu khác nhau. Các phương pháp để thực hiện việc cập nhật và truy vấn rất khác nhau giữa hai lần thực hiện. Cơ sở dữ liệu liên quan sử dụng cú pháp SQL trong khi cơ sở dữ liệu MongoDB sử dụng JavaScript Object Notation (JSON) .Các chức năng tổng hợp khác nhau thậm chí nhiều hơn giữa hai lần triển khai do thêm tính phức tạp của việc sử dụng các hàm 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. Mối quan hệ giữa các đối tượng dữ liệu, chẳng hạn như Bộ phận (Department) trong đối tượng Người dùng đã được triển khai trong MongoDB bằng cách lưu trữ một tham chiếu đến tài liệu khác thay vì lồng toàn bộ tài liệu. Ví dụ: một tham chiếu đến một Bộ đã được lưu trữ trong một đối tượng Người dùng.

Đối với phần Cập nhật trong thử nghiệm 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 (Primary key) của họ. Loại thứ hai cập nhật tên họ (last name) của bất kỳ người dùng nào có tên khớp với tên đã chọn (tên ( First name) được chọn cho thử nghiệm của chúng tôi là Jacob). Loại thứ ba cập nhật tên của một phần tư các dự án dựa trên khóa (key) của chúng. Chúng tôi đã thử nghiệm từng loại trong số ba loại cập nhật này trên cả bốn trường hợp kiểm thử.

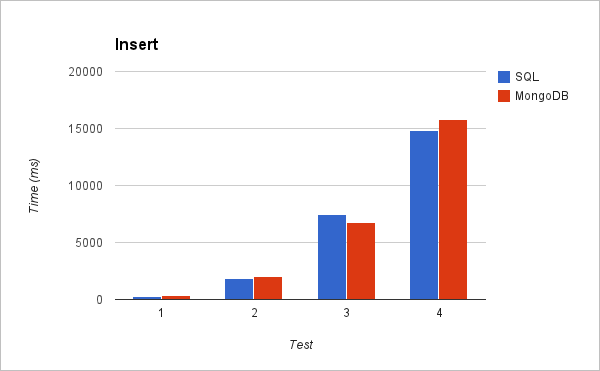
Chúng tôi cũng đã thử nghiệm bảy truy vấn chọn ( Select) khác nhau. Các lựa chọn được chia thành hai loại: đơn giản và phức tạp. Các truy vấn đơn giản liên quan đến việc chọn dữ liệu chỉ có mảng đối tượng. Các truy vấn phức tạp liên quan đến nhiều loại đối tượng, truy vấn lồng nhau và hàm tổng hợp. Số lượt chọn chạy cho trường hợp kiểm tra 1 và 3 là 100 và số lượt chọn chạy cho trường hợp 2 và 4 là 200.

Bốn lựa chọn đơn giản khác nhau mà chúng tôi đã tạo cho các thử nghiệm như sau. Lựa chọn đầu tiên đã truy xuất một bộ phận bằng khóa chính của nó. Lựa chọn thứ hai đã truy xuất một bộ phận bằng một cái tên được chọn ngẫu nhiên. Tất cả các phòng ban đều có tên duy nhất và tên được đảm bảo phù hợp với một kết quả trong department’s collection. Lựa chọn thứ ba đã truy xuất người dùng bằng khóa chính của nó. Lựa chọn thứ tư đã truy xuất tất cả người dùng có tên khớp với tên được chọn ngẫu nhiên; Tên được chọn ngẫu nhiên này có thể không khớp với bất kỳ người dùng nào trong **collection**.

Ba lựa chọn phức tạp khác nhau được sử dụng trong các thử nghiệm liên quan đến **Joint operation (Hoạt động chung)** và theo sau. Lựa chọn phức tạp đầu tiên đã truy xuất tất cả các bộ phận chứa một hoặc nhiều người dùng mà có tên ( First name) được chọn ngẫu nhiên. Vì tên đầu tiên (First name )được gán ngẫu nhiên nên có thể trả lại hoặc không 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 số ba dự án được chọn ngẫu nhiên. Vì người dùng được gán ngẫu nhiên cho các dự án, truy vấn này cũng có thể không trả về bất kỳ kết quả phù hợp hợp lệ nào. Lựa chọn phức tạp cuối cùng trả về độ tuổi trung bình của những người dùng làm việc trên bất kỳ dự án nào trong ba dự án được chọn ngẫu nhiên và rõ ràng, chứa hàm tập hợp trung bình (aggregate function average). Truy vấn cuối cùng này có thể trả về không nếu không có người dùng nào được gán cho các dự án.

**4. RESULTS ( Kết quả)**

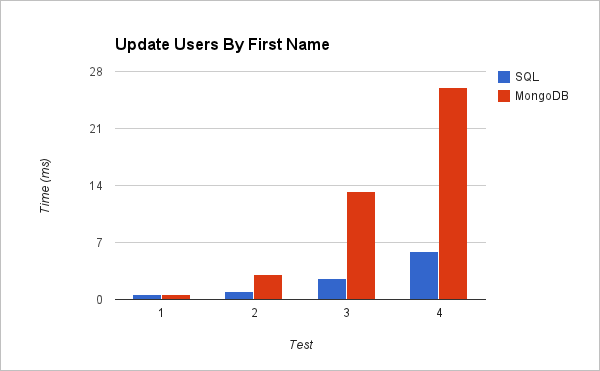
Theo như kết quả của những cuộc thí nghiệm được trình bày ở biểu đồ 3 thông qua bài trích dẫn 13. Mỗi con số so sánh thời gian trung bình, tính bằng milli giây, của việc thực hiện SQL so với việc thực hiện MongoDB theo công việc đưa ra cho mỗi trường hợp trong 4 trường hợp thí nghiệm(Test case). Từ những bước đầu, chúng tôi đã thực hiện 100 và 200 lần cho mỗi thí nghiệm để giảm chênh lệch trong bất kì trường hợp nào trong thời gian trung bình.



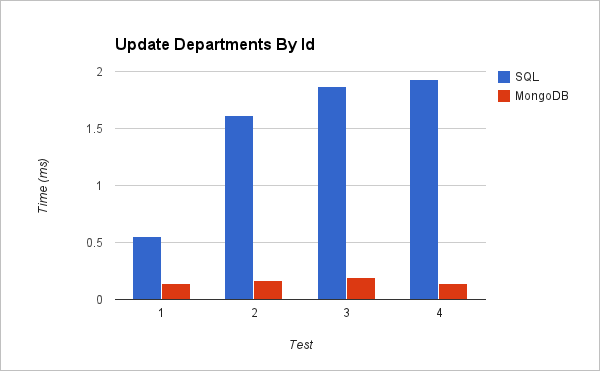
**Figure 3. Insert Speed**

Biểu đồ 3 thể hiện sự khác biệt trong viêc chèn thêm tốc độ giữa 2 hệ thống cơ sở dữ liệu. Cả 2 cơ sở dữ liệu đều thể hiện 1 cách nhất quán hơn so với những cơ sỡ dữ liệu khác. SQL thực hiện nhanh hơn MongoDB trong trường hợp thí nghiệm(Test case) 1, 2 và 4, trong khi MongoDB chạy nhanh hơn SQL ở mức độ không đáng kể trong trường hợp thí nghiệm(Test case) 3. Như đã lưu ý ở trên, mối quan hệ giữa việc chèn thêm MongoDB được thực hiện và các dối tượng dữ liệu sử dụng như 1 sự tham khảo. Điều này không yêu cầu chèn thêm thông tin trong MongoDB. MongoDB chiềm thấp hơn 34% trong trường hợp thí nghiệm (Test case)1 so với SQL và thấp hơn 7% trong trường hợp thí nghiệm (Test case) 2 và 4, trong khi SQL thấp hơn 10% trong trường hợp thí nghiệm(Test case) 3 so với MongoDB.

Biểu đồ 4,5 và 6 thể hiện thời gian qua đơn vị đo ms cho mỗi lần cập nhật khác nhau mà chúng tôi thực hiện. Ở lần cập nhật trong biểu đồ 4, chúng tôi đã thể hiện cập nhật của người dùng dựa trên tên người dùng với tên không được lập chỉ mục trong bất kì cách nào. Ở biểu đồ 4, MongoDB thể hiện sự nhất quán tệ hơn so với máy chủ SQL(SQL Server) chỗ mỗi lần thực hiện 4 trường hợp thí nghiệm. Bởi vì biểu đồ số 4 là bản cập nhật dựa trên cột không được lập chỉ mục, và bản chất phi cấu trúc của MongoDB cho phép bất kì truy vấn không được lập chỉ mục để thể hiện những tìm kiếm phức tạp trên mỗi dự liệu, MongoDB mất nhiều thời gian hơn để thực hiện cập nhật trên dữ liệu lớn hơn SQL. Trong trường hợp thí nghiệm(Test case) 1, MySQL và MongoDB hầu như thể hiện giống nhau. Khi kích thước của cơ sở dữ liệu tăng lên, thời gian để MongoDB cập nhật người dùng bởi tên vùng của họ từ lần thứ 3 chậm hơn SQL cho trường hợp thí nghiệm thứ (Test case) 2, từ lần thứ 5 trở đi chậm hơn MongoDB trong trường hợp thí nghiệm thứ (Test case ) 3.

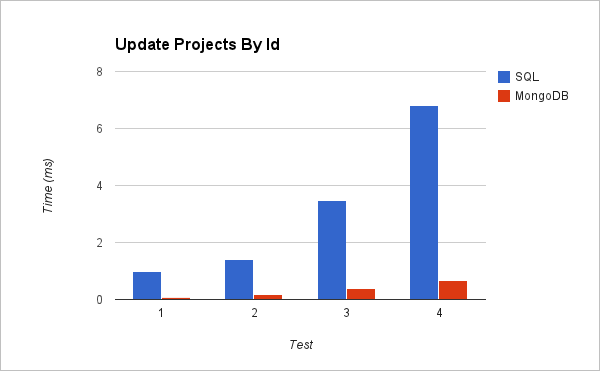


**Figure 4. Update - First Name**



**Figure 5. Update – Department ID**

MongoDB vượt trội hơn SQL trong mọi trường hợp cập nhật liên quan đến việc sử dụng khóa chính, được thể hiện ở bản đồ 5 và 6. Bản cập nhật trong bản đồ 5 liên quan đến bộ phận cập nhật sử dụng tên của bộ phận đó, trong khi bản cập nhật trong bản đồ 6 liên quan đến việc cập nhật sử dụng tên dự án (project ID). Sự đa dạng về thời gian trong mỗi lần cập nhật thì nhỏ, đặc biệt là MongoDB, ít hơn 0.2ms trong 4 thử nghiệm (Test case). Máy chủ SQL (SQL Server) dao động từ 1 phần nhỏ từ 0.55ms trong trường hợp thử nghiệm 1 đến 1.93 cho lần thứ nghiệm 2. Ở biểu đồ 6, máy chủ SQL that hơn 13 lần so với MongoDB trong thử nghiệm (Test case). 4 khi cập nhật bới định danh của phòng ban(Department ID). Chúng tôi tin rằng sự chênh lệch này là bởi vì MongoDB có 1 chỉ mục được xây dựng trước trên khóa chính của tài liệu, cái mà thể hiện nhanh hơn khóa chính của máy chủ được chỉ định( SQL Server’s)

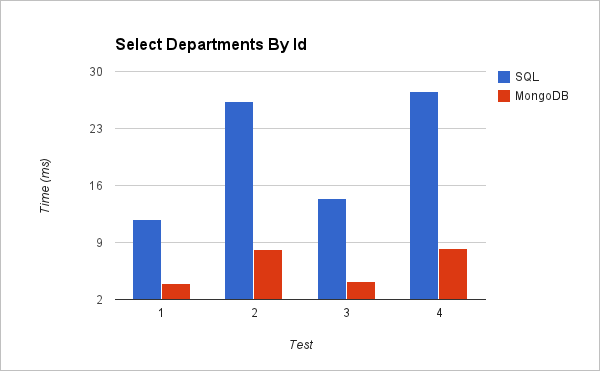


**Figure 6. Update – Project ID**

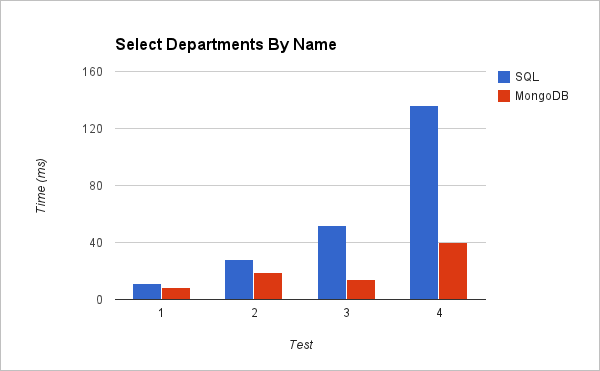
Biểu đồ 7 thông qua trích dẫn 13 minh họa thời gian thực hiện các truy vấn được chọn, với biểu đồ 7-10 thể hiện kết quả cho truy vấn đơn giản và biều đồ 11-13 thể hiện kết qủa cho các truy vấn phức tạp. Các lựa chọn đơn giản không nằm trong khóa của dữ liệu, cũng không thể hiện trong giá trị khác. Các lựa chọn phức tạp liên quan đến nhóm, mối quan hệ M:N và sự tổng hợp. Chúng tôi lưu ý rằng mặc dù kích thước của bảng tăng lên ở trường hợp thử nghiệm 1-4, mỗi thử nghiệm có 1 số lượng lựa chọn đa dạng. Trường hợp thí nghiệm thứ 2 có số lượng lựa chọn tăng hơn so với thử nghiệm 1, trong khi thử nghiệm 3 có cùng số lượng lựa chọn với thử nghiệm 1, nhưng nhiều dữ liệu hơn. Tương tự, lựa chọn thử nghiệm số 2 và 4 là như nhau, nhưng thử nghiệm 4 có nhiều dữ liều hơn. Vì vậy, không phải lúc nào cũng hữu ích khi so sánh thời gian thực hiện các thử nghiệm với nhau, nhưng tốt hơn là so sánh thời gian thực hiện của MongoDB với SQL.

Ở biểu đồ 7, thời gian để lựa chọn bộ phận bởi tên định danh của SQL nhanh hơn 3.5 lần so với MySQL trong tất cả các trường hợp thử nghiệm. Mặc dù kích thước dữ liệu tăng lên từ thử nghiệm 1 đên 4, thời gian thực hiện cho thí nghiệm này cũng bị ảnh hưởng bởi số lượng lựa chọn, như đã minh họa bởi việc giảm thời gian thực hiện cho cả thí nghiệm 3 với thí nghiệm 2.

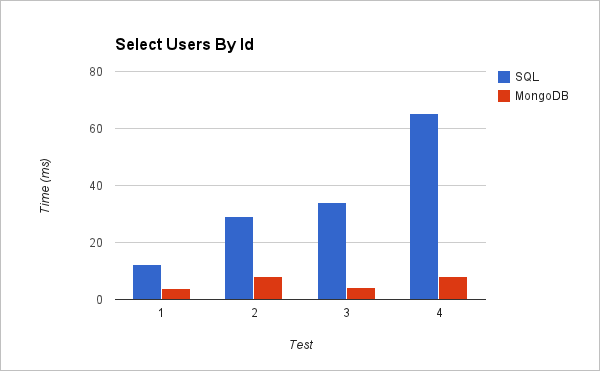
Biểu đồ 8 minh chứng thời gian lựa chọn bộ phận bởi tên. SQL thấp hơn 1.5 lần so với MongoDB cho lần thử nghiệm 1 và 2, và chậm hơn 3 lần so với MongoDB cho lần thử nghiệm 2 và 4. Tương tự ở biểu đồ số 9, các bộ phận được chọn với tên, nhưng không đáng kể. MongoDB nhanh hơn SQL 3 lần cho lần thử nghiệm 1, 2 và 8 lần cho lần thử nghiệm 3, 4. Đáng chú ý là ở biểu đồ 8 và 9, thời gian thực hiện cho SQL tăng như số lượng dữ liệu tăng, trong khi thời gian thực hiện của MongoDB có liên quan đến số lượng dữ liệu cũng như số lần chọn được thể hiện. Trường hợp thí nghiệm 1 và 3 liên quan 100 lựa chọn, trong khi trường hợp 2 và 4 liên quan đến 200 lựa chọn, và có thời gian thực hiện cao hơn.



**Figure 7. Select – Department ID**



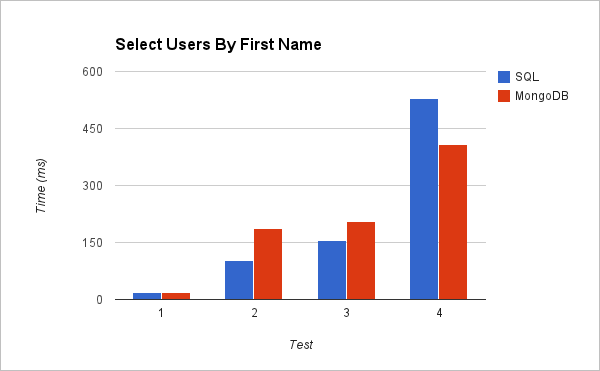
**Figure 8. Select – Department Name**



**Figure 9. Select – User ID**

Truy vấn ở biểu đồ 10 thể hiện người dùng lựa chọn dựa trên tên của họ. Kết quả ở biểu đồ 10 đồng nhất với kết quả ở biểu đồ 4 trong lần thử nghiệm 1-3, cái mà người dùng được cập nhật dựa trên tên của họ. SQL có thời gian thực hiện nhanh hơn MongoDB ở lần thử nghiệm 1-3 bởi vì nó có thể chọn các thuộc tính riêng biệt trong bảng liên quan 1 cách dễ dàng hơn là định vị giá trị của bộ sưu tập MongoDB(MongoDB’s collection). Tuy nhiên, khi dữ liệu tăng lên đủ trong trường hợp thí nghiệm 4, thời gian thực hiện của SQL cao hơn 23% so với MongoDB. Thời gian xử lí truy vấn của SQL trở nên thấp hơn thời gian tìm kiếm của MongoDB khi dữ liệu tăng lên.

Biểu đồ 7-9 minh họa rằng MongoDB vượt trội hơn hẳn SQL trong tất cả các thử nghiệm. Chúng tôi tin rằng điều này là bởi vì sự kết hợp của chỉ số được sử dụng bởi MongoDB và bộ nhớ sử dụng của nó. MongoDB sử dụng các tiệp kết nối để lưu trữ tất cả các dữ liệu trong bộ nhớ hơn là lưu trên ổ đĩa ( Miễn là hệ thống có bộ nhớ sẵn). Bởi vì máy chủ SQL phải truy xuất tât cá dữ liệu của nó từ ổ đĩa và tốc độ ổ đĩa chậm hơn thời gian tìm nạp bộ nhớ, MongoDB vượt trội hơn SQL.



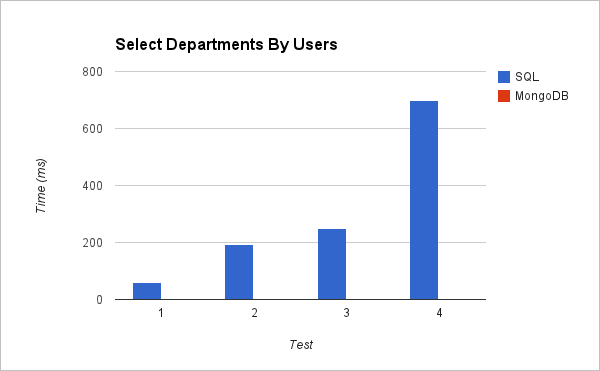
**Figure 10. Select – User First Name**

Trong nhiều truy vấn phức tạp ở biểu đồ 11 và 12, MongoDB có thời gian thực hiện nhanh hơn đáng kể so với SQL. Thời gian thực hiện của MongoDB quá thấp, không dễ nhìn thấy được. Thời gian thực hiện cho MongoDB ở biểu đồ 10 dao động từ 0.13 trong trường hợp thí nghiệm 1 đến 0.19 ở thí nghiệm 2. Thời gian thực hiện cho MongoDB ở biểu đồ 11 dao động từ 0.24 trong trường hợp thí nghiệm 1 (Test case 1) đến 0.42 ở thí nghiệm 4 (Test case 4).

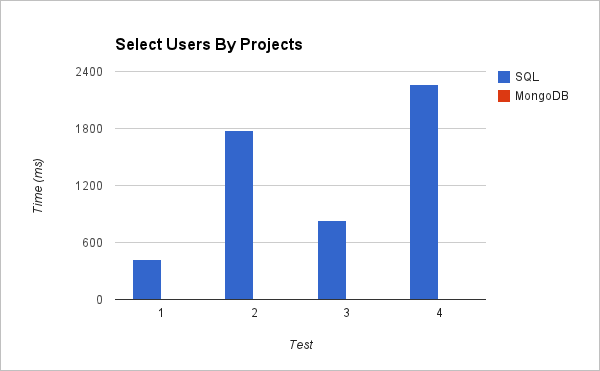
Như được thể hiện ở biểu đồ 13, MongoDB thể hiện tốt hơn trong những truy vấn phức tạp, ngoại trừ khi bàn đến việc sử dụng tổng hợp chức năng. Ở biểu đồ 13, thí nghiệm 4 (Test case 4), SQL có thời gian thực hiện 1.836ms trong khi MongoDB có thời gian thực hiện là 41.754ms.

MongoDB hầu như chậm hơn SQL 23 lần. Chúng tôi tin rằng điều này là bởi vì MongoDB không có chức năng tổng hợp được xác định với nhiều phương pháp, chẳng hạn như hàm trung bình SQL. Bởi vì MongoDB phải sử dụng phương pháp MapReduce để cho phép người phát triển tạo chức năng tổng hợp của nó, điều này làm chậm sự thực hiện 1 cách đáng kể.

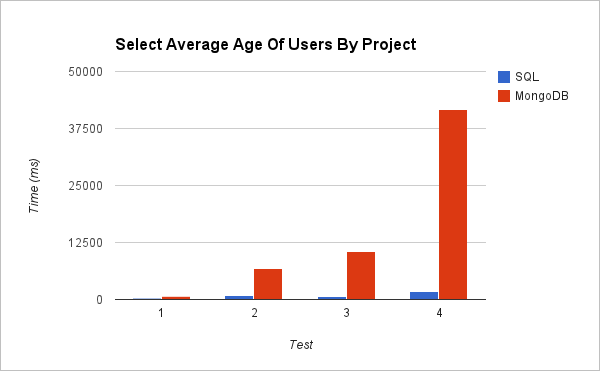
Không giống như sử dụng SQL, các quyết định thực hiện bổ sung được yêu cầu có ảnh hưởng đến hiệu suất của MongoDB. Sao chép dữ liệu trong mỗi bộ sưu tập thay vì bản trích dẫn có thể dẫn đến kết quả của hiện thực hiện nhanh hơn, mặc dù nó sẽ yêu cầu lưu trữ nhiều hơn. Thêm vào cách triển khai các mối quan hệ, chúng ta cũng cần triển khai các chức năng tổng hợp.



**Figure 11. Select Departments by Users**



**Figure 12. Select Users by Project**



**Figure 13. Select Average Age of Users by Project**

**5. CONCLUSIONS AND FUTURE WORK (Kết luận và công việc trong tương lai)**

Nhìn chung, khi so sánh SQL với MongoDB, MongoDB có hiệu suất thời gian chạy tốt hơn cho các phần 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 (non-key attribute) , cũng như hàm tổng hợp các truy xuất.

MongoDB có thể là một giải pháp tốt cho các tập dữ liệu lớn hơn, trong đó lược đồ (**schema**) 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 định nghĩa lược đồ cứng **(rigid schema),** MongoDB sẽ dễ dàng xử lý một lược đồ động (**dynamic schema)** như một hệ thống quản lý tài liệu với một số trường động (**dynamic fields)** và chỉ một số lĩnh vực **(Fields)** tìm kiếm được nhiều người biết đến. Tóm lại, MongoDB chắc chắn là sự lựa chọn cho những người dùng cần cấu trúc cơ sở dữ liệu linh động hơn. MongoDB có thể là một giải pháp tốt cho các tập dữ liệu lớn hơn, trong đó lược 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 một lược đồ nghiêm ngặt được xác định và một lượng cấu trúc dữ liệu vừa phải, 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 hàm tổng hợp và truy vấn dựa trên các giá trị không khóa (**non-key**) . Ngoài ra, MongoDB yêu cầu thêm nỗ lực trong việc triển khai so với SQL và các quyết định bắt buộc ảnh hưởng đến hiệu suất của nó. Cuối cùng, SQL là tiêu chuẩn ngành **(industry standard)** và được khuyến khích rộng rãi hơn nhiều so với MongoDB.

Một điều chúng tôi muốn thấy như công việc trong tương lai từ bài báo này là 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 sẽ tăng lên khi nó được sử dụng theo cách này. Các công việc khác trong tương lai sẽ liên quan đến việc chạy các thử nghiệm tương tự trên cả MongoDB và SQL với một lược đồ lớn hơn và phức tạp hơn nhiều. Về lý thuyết, MongoDB vượt trội hơn SQL do nó không yêu cầu một lược đồ thực sự (true schema) được xác định. Vì hệ quản trị cơ sỡ dữ liệu quan hệ SQL có tổng chi phí đáng kể và yêu cầu các liên kết bổ sung trong một lược đồ phức tạp hơn, chúng tôi đưa ra giả thuyết rằng hiệu suất của SQL sẽ tiếp tục giảm so với MongoDB.

**NOTE**

* FAQ (Frequently Asked Questions): các câu hỏi thường gặp, nó đưa ra những câu hỏi phổ biến nhất mà người dùng thường gặp sau đó có phần giải đáp cho các câu hỏi
* HTML (HyperText Markup Language): Ngôn ngữ đánh dấu được thiết kế ra để tạo nên các trang web với các mẩu thông tin được trình bày trên World Wide Web
* LAN (Local Area Network): Mạng máy tính nội bộ
* Network Administrator: Người quản trị thiên về phần cứng
* OSI: Open System Interconnection (mô hình chuẩn OSI)
* PPP (Point-to-Point Protocol): giao thức kết nối Internet tin cậy thông qua Modem
* RAM (Read-Only Memory): Bộ nhớ khả biến cho phép truy xuất đọc-ghi ngẫu nhiên đến bất kỳ vị trí nào trong bộ nhớ.
* Array là Mảng
* Schema là Lược đồ
* Field là Trường
* Constrain là Ràng buộc
* Relational database có tên đầy đủ là relational database management system, được viết tắt là RDBMS **một cơ sở dữ liệu lưu trữ và trao quyền truy cập đến các điểm dữ liệu liên kết với nhau**, RDBMS là cơ sở cho SQL, và cho tất cả các hệ thống cơ sở dữ liệu hiện đại như MS SQL Server, IBM DB2, Oracle, MySQL và Microsoft Access.