**ĐẠI HỌC QUỐC GIA TP. HỒ CHÍ MINH**

**TRƯỜNG ĐẠI HỌC CÔNG NGHỆ THÔNG TIN**

**KHOA CÔNG NGHỆ PHẦN MỀM**

**Description: C:\Users\Long Van\Desktop\uit.gif**

**KHOÁ LUẬN TỐT NGHIỆP**

**TÌM HIỂU CƠ SỞ DỮ LIỆU NOSQL VÀ**

**XÂY DỰNG ỨNG DỤNG**

Giảng viên hướng dẫn : **Ths**. **Phạm Thi Vương**

Sinh viên thực hiện : **DƯƠNG THÂN DÂN - 08520057**

**BÙI NGỌC HUY - 08520544**

Lớp : **CNPM03**

Khoá : **2008 - 2012**

***TP. Hồ Chí Minh, tháng 11 năm 2012***

**MỞ ĐẦU**

Ngày nay với kỹ nguyên công nghệ bùng nổ, thành công của Internet đã khiến cho số lượng người dùng truy cập vào cùng một hệ thống ngày càng tăng. Điển hình như Facebook một tháng phục vụ hơn 1000 tỉ truy cập và hơn 800 triệu lượt khách ghé thăm thì ta mới hình dung được sự bùng nổ của thông tin như thế nào. Để giải quyết vấn đề bùng nổ như trên thì chúng ta đã mở rộng các hệ thống máy chủ siêu lớn, phân thành nhiều cụm đặt khắp nơi trên thế giới. Nhưng với tốc độ phát triển theo cấp số như hiện nay thì việc tăng số lượng máy chủ thôi vẫn chưa đủ. Ta cần xem xét và nâng cấp các giải pháp lưu trữ cho tương lai.

Hệ thống máy chủ cơ sở dữ liệu đòi hỏi phải rất mạnh mẽ nếu không máy chủ sẽ bị quá tải. Với các hệ thống với số lượng lên đến hàng triệu cho đến hàng tỉ thì việc hiệu năng tốt là việc bắt buộc.Các hệ RBDMs hiện nay thì vấn đề hiệu năng thường không tốt cho trường hợp này. Ngôn ngữ SQL là ngôn ngữ thông dịch với các ràng buộc trong các bảng khiến cho hiệu năng thực sự của hệ thống cơ sở dữ liệu khi thực thi là khá ì ạch với hệ thống lớn như kể trên.Chưa kể là với hệ thống lớn thì vấn đề phân tán dữ liệu, tính toàn vẹn dữ liệu là việc rất quan trọng. NoSQL đáp ứng được tất cả các yêu cầu này.Với tốc độ nhanh do không phải qua các câu truy vấn SQL, có tính sẵn sàng, phân tán cao và độ ổn định tuyệt vời. Rất thích hợp cho các hệ thống có lượt truy vấn lớn. Ở trong khoá luận này, chúng tôi sẽ nghiên cứu về một loại NoSQL khá phổ biến – RavenDB.

RavenDB là một cơ sở dữ liệu mã nguồn mở có hỗ trợ transactional (giao dịch) được viết cho nền tảng .NET. RavenDB đưa ra mô hình dữ liệu linh hoạt (flexible data model) nhằm đáp ứng yêu cầu của các hệ thống thế giới thực (real-world systems). RavenDB cho phép xây dựng những ứng dụng có hiệu suất cao(high-performance), độ trễ thấp(low-latency) một cách nhanh chóng và hiệu quả. RavenDB xứng đáng là một cơ sở dữ liệu

**LỜI CẢM ƠN**

Chúngemxin gửi lời cảm ơn sâu sắc đến thầy Phạm Thi Vương, đã giúp đỡ, tạo điều kiện cho nhóm hoàn thành tốt khóa luận tốt nghiệp này. Thầy đã tận tình hướng dẫn và đưa ra những nhận xét vô cùng quý giá để đề tài ngày càng hoàn thiện hơn. Những góp ý của thầy giúp cho chúng em tiếp cận, hiểu rõ và giải quyết vấn đề dễ dàng hơn.

Đồng thời, chúng em cũng xin bày tỏ lòng biết ơn đến quý thầy, cô Trường Đại Học Công Nghệ Thông Tin – Đại Học Quốc Gia Thành Phố Hồ Chí Minh, đặc biệt là các thầy, cô khoa Kỹ Thuật Phần Mềm đã tận tình truyền đạt kiến thức, kinh nghiệm cho chúng em từ những ngày đầu học tập tại trường. Sự nhiệt tình của các thầy, cô đã giúp cho chúng em có kiến thức nền tảng vững chắc cũng như kinh nghiệm thực tiễn quý báu để chúng em có thể hoàn thành tốt các nhiệm vụ học tập, làm việc và nghiên cứu.

Bên cạnh đó, chúng em cũng gửi lời cảm ơn đến gia đình, các anh, chị, bạn bè đã động viên, giúp đỡ chúng em rất nhiều trong quá trình học tập cũng như trong cuộc sống.

Thành phố Hồ Chí Minh, ngày 11, tháng 11, năm 2012

Nhóm sinh viên thực hiện

Dương Thân Dân – Bùi Ngọc Huy

**NHẬN XÉT CỦA GIẢNG VIÊN HƯỚNG DẪN**

**NHẬN XÉT CỦA GIẢNG VIÊN PHẢN BIỆN**

**MỤC LỤC**

[CHƯƠNG 1 – Tổng quan về cơ sở dữ liệu NoSQL 1](#_Toc342550209)

[1.1. Giới thiệu về NoSQL: 1](#_Toc342550210)

[1.2. Một số khái niệm mới NoSQL: 1](#_Toc342550211)

[1.3. Ưu nhược điểm của cơ sở dữ liệu NoSQL: 3](#_Toc342550212)

[1.31. Ưu điểm: 3](#_Toc342550213)

[1.32. Nhược điểm: 4](#_Toc342550214)

[1.4. Cách triển khai một ứng dụng NoSQL: 6](#_Toc342550215)

[CHƯƠNG 2 – Tìm hiểu về các giải pháp cơ sở dữ liệu NoSQL 6](#_Toc342550216)

[2.1. Wide Column Store / Column Families 7](#_Toc342550217)

[2.2. Key-Value Store / Tuple Store 7](#_Toc342550218)

[2.3. Document Store 7](#_Toc342550219)

[2.4. Graph Database 7](#_Toc342550220)

[CHƯƠNG 3- Tìm hiểu về RavenDB 7](#_Toc342550221)

[3.1. Lý do sử dụng RavenDB 7](#_Toc342550222)

[3.2. Đặc điểm của RavenDB 7](#_Toc342550223)

[3.3. Vài kĩ thuật đặc biệt khi sử dụng RavenDB 7](#_Toc342550224)

[CHƯƠNG 4 – Xây dựng ứng dụng sử dụng RavenDB 7](#_Toc342550225)

[4.1. Giới thiệu về ứng dụng 7](#_Toc342550226)

[4.2. Phân tích, thiết kế hệ thống 7](#_Toc342550227)

[4.3. Triển khai 7](#_Toc342550228)

[4.4. Kết quả test performance để thể hiện sức mạnh của NoSQL 7](#_Toc342550229)

[CHƯƠNG 5 – Tổng kết 7](#_Toc342550230)

[5.1. Kết quả đạt được 7](#_Toc342550231)

[5.1.1. Về mặt lý thuyết 7](#_Toc342550232)

[5.1.2. Về mặt thực nghiệm 7](#_Toc342550233)

[5.2. Hướng phát triển 7](#_Toc342550234)

[Tài liệu tham khảo 8](#_Toc342550235)

**DANH MỤC CÁC BẢNG, SƠ ĐỒ, HÌNH**

# CHƯƠNG 1 – Giới thiệu đề tài

## Vấn đề tìm hiểu

* Trong khoảng hơn 2 thập niên trở lại đây, hệ quản trị cơ sở dữ liệu quan hệ - RDBMs là sự lựa chọn duy nhất cho việc quản trị cơ sở dữ liệu. Tuy nhiên, với các yêu cầu mới hiện nay thì RDBMs đã bộc lộ yếu điểm. Chính sự quá chặc chẽ, yêu cầu nhất quán dữ liệu đã gây ra sự rườm rà, phức tạp làm giảm hiệu xuất hoạt động, nhất là trong trường hợp phải chứa một lượng lớn dữ liệu. Nhưng với sự bùng nổ công nghệ như hiện nay, nhất là với mạng Internet thì lượng dữ liệu cần lưu trữ ngày càng tăng. Yêu cầu cho việc lưu trữ ngày càng cao như: lưu trữ nhiều dữ liệu, tốc độ truy xuất nhanh, phân tán dữ liệu trên nhiều máy chủ… thì với mô hình cơ sở dữ liệu quan hệ như hiện nay thì rỏ ràng không thể đáp ứng đủ các yêu cầu trên.
* Mọi vấn đề đều có giải pháp. Thật vậy, những năm gần đây đã nổi lên một xu hướng lưu trữ mới, một cách thức trái ngược với cơ sở dữ liệu quan hệ - đó là cơ sở dữ liệu phi quan hệ - NoSQL. NoSQL sinh ra để khắc phục các vấn đề mà một cơ sở dữ liệu dạng RDBMs gặp phải. NoSQL sinh ra không phải để cạnh tranh với RDBMs mà là để đảm nhiệm những việc mà RDBMs chưa làm tốt.
* Mục tiêu mà NoSQL nhắm đến đó là hiệu xuất hoạt động cao với số lượng dữ liệu cực lớn. Tuy nhiên để đạt được điều đó thì NoSQL đã bỏ qua thông dịch trong SQL cùng với những truy vấn rườm ra. Việc sử dụng các ràng buộc quan hệ cùng truy vấn SQL có vẻ thân thiện và thích hợp với phần đông dữ liệu. Tuy nhiên, nếu dữ liệu quá đơn giản, các thủ tục SQL sẽ không cần thiết (theo Curt Monash - một nhà phân tích cơ sở dữ liệu, một blogger). Đồng thời NoSQL cũng có cách thiết kế dữ liệu khác với cơ sở dữ liệu truyền thống như: tương tưởng thiết kế dữ liệu phi quan hệ, lưu trữ dữ liệu dạng document, sử dụng tối đa indexes… Trong các đặt tính đó, dữ liệu phi quan hệ là một yếu tố quan trọng góp phần làm thanh công cho NoSQL. Dữ liệu phi quan hệ tức là không tuân theo các dạng chuẩn hóa mà cơ sở dữ liệu RDBMs đặt ra. Thay vào đó, khi thiết kế một cơ sở dữ kiệu NoSQL ta phải tuân theo một số quy tắt mới mà NoSQL đặt ra để đạt được hiệu xuất hoạt động cao. Chính sự khác biệt giữa 2 loại cơ sở dữ liệu này dẫn đến cách thiết kế cũng khác nhau. Đa số các lập trình viên đều quen với mô hình quan hệ truyền thống, do đó cần phải tìm hiểu kĩ cách thiết kế dữ liệu của NoSQL để đạt được hiệu xuất mong muốn.
* Đồng ý rằng RDBMs cung cấp một mô hình tuyệt vời để đảm bảo tính toàn vẹn dữ liệu. Tuy nhiên, rất nhiều người lựa chọn NoSQL đã nói rằng chúng không quá cần thiết cho nhu cầu của họ.
* Như vậy, trong đề tài này chúng tôi sẽ tìm hiểu xem NoSQL đã giải quyết các vấn đề trên như thế nào và áp dụng kiến thức tìm hiểu đó vào việc xây dựng một ứng dụng sử dụng cơ sở dữ liệu dạng NoSQL.

## Mục tiêu đề tài

* Tìm hiểu cách làm việc của NoSQL, phân loại và đặt điểm của từng loại.
* Tìm hiểu trường hợp áp dụng cơ sở dữ liệu dạng NoSQL. Phân tích làm rỏ ưu khuyết điểm của việc áp dụng cơ sở dữ liệu NoSQL. So sánh giữa việc sử dụng cơ sở dữ liệu RDBMs và cơ sở dữ liệu NoSQL.
* So sánh hiệu xuất giữa một cơ sở dữ liệu dạng NoSQL và cơ sở dữ liệu dạng RDBMs để làm rỏ hiệu suất hoạt động của NoSQL.
* Tìm hiểu cách thiết kế một cơ sở dữ liệu cho ứng dụng sử dụng NoSQL
* Sử dụng các kiến thức về NoSQL để xây dựng một ứng dụng sử dụng cơ sở dữ liệu dạng NoSQL đồng thời để tổng hợp lại kiến thức đã học trước đây.

## Đặc điểm

* **Đặc điểm công nghệ**: Đừng gọi chúng là database. CTO của Amazon, Werner Vogels đề cập đến hệ thống Dynamo của họ đã gọi nó là một "highly available key-value store". Google gọi BigTable để nhấn mạnh  đây là "distributed storage system for managing structured data" (hệ thống lưu trữ và quản lý dữ liệu cấu trúc có phân tán).  
  Có thể thổi bay một lượng dữ liệu cực lớn. Hypertable, một open source column-based database trên mô hình BigTable được sử dụng cho local search engine của Zvents Inc có thể ghi tới 1 tỷ cell dữ liệu mỗi ngày (theo  Doug Judd một kỹ sư của Zvents). Trong khi đó BigTable kết hợp với MapReduce có thể xử lý tới 20 petabytes dữ liệu mỗi ngày.
* **Đặc điểm dề tài**: đề tài mang tính chất tìm hiểu, tổng hợp kiến thức từ nhiều nguồn liên quan và sau đó là phân tích đưa ra kết luận. Song song việc tìm hiểu là áp dụng vào thực tế. Áp dụng có thành công hay không và thành công đến mức độ nào đều do kết quả của quá trình tìm hiểu mang lại. Do đó cần tìm hiểu, nghiên cứu một cách kĩ lưỡng, mỗi vấn đề đều phải có phân tích và đưa ra kết luận. Cần đưa ra các kết luận hữu ích như: khi nào nên sử dụng cơ sở dữ liệu NoSQL, nên chọn loại cơ sở dữ liệu NoSQL nào, hướng dẫn cách xây dựng một cấu trúc dữ liệu cho cơ sở dữ liệu NoSQL…

## Hướng giải quyết

* Do loại cơ sở dữ liệu này khá mới mẻ nên rất ít sách viết đầy đủ, mà cần tìm tài liệu để học tập chủ yếu từ Internet. Các tài liệu viết dưới dạng nhỏ lẽ, viết riêng cho từng vấn đề, rất ít tài liệu bao quát nên khi tìm hiểu cần tổng hợp lại cho đầy đủ và dễ hiểu.
* Công việc tìm hiểu thôi vẫn chưa đủ, ta cần có những phân tích, kết luận dựa trên kết quả tìm hiểu đó.
* Tìm hiểu đến đâu viết demo đến đấy. Có viết demo mới hiểu rỏ được vấn đề.
* Khi thực hiện test hiệu suất hoạt động, cần thực hiện với một lượng lớn dữ liệu và nhiều người dùng để chứng tỏ sức mạng của cơ sở dữ liệu NoSQL.

# CHƯƠNG 2 – Tổng quan về cơ sở dữ liệu NoSQL

## NoSQL là gì ?

* NoSQL còn có nghĩa là Non-Relational (NoRel) - không ràng buộc. Tuy nhiên, thuật ngữ đó ít phổ dụng hơn và ngày nay người ta thường dịch NoSQL thành Not Only SQL - Không chỉ SQL. NoSQL ám chỉ đến những cơ sở dữ liệu không dùng mô hình dữ liệu quan hệ để quản lý dữ liệu trong lĩnh vực phần mềm, bởi thế nếu dữ liệu của bạn thực sự có tính quan hệ chặt chẽ, hãy sử dụng RDBMs.
* NoSQL có dạng mô hình tự do (schema-free). Bạn không phải thiết kế các table và các cấu trúc như với SQL, việc duy nhất bạn phải làm là cứ thế insert các giá trị mới.

## Lịch sử

Thuật ngữ NoSQL được giới thiệu lần đầu vào năm 1998 sử dụng làm tên gọi chung cho các lightweight open source relational database (cơ sở dữ liệu quan hệ nguồn mở nhỏ) nhưng không sử dụng [SQL](http://thegioilaptrinh.net/sql-va-lich-su-ra-doi-5-1-1.html) cho truy vấn.   
  
Vào năm 2009, Eric Evans, nhân viên của Rackspace giới thiệu lại thuật ngữ NoSQL trong một hội thảo về cơ sở dữ liệu nguồn mở phân tán. Thuật ngữ NoSQL đánh dấu bước phát triển của  thế hệ database mới: distributed (phân tán) + non-relational (không ràng buộc).

## Định nghĩa

* Thế hệ database kế tiếp là một thế hệ cơ sở dữ liệu non-relational (không ràng buộc), distributed (phân tán), open source, horizontal scalable (khả năng mở rộng theo chiều ngang) có thể lưu trữ, xử lý từ một lượng rất nhỏ cho tới hàng petabytes dữ liệu trong hệ thống có độ chịu tải, lỗi cao với những đòi hỏi về tài nguyên phần cứng thấp.
* Một số đặc điểm nhận dạng cho thế hệ database mới này bao gồm:
* Schema-free.
* Hỗ trợ mở rộng dễ dàng.
* API đơn giản.
* Eventual consistency (nhất quán cuối) và/hoặc transactions hạn chế trên các thành phần dữ liệu đơn lẻ.
* Không giới hạn không gian dữ liệu,...
* NoSQL storage đặc biệt phổ dụng trong thời kỳ Web 2.0 bùng nổ, nơi các mạng dịch vụ dữ liệu cộng đồng cho phép người dùng tạo hàng tỷ nội dung trên web. Do đó, dữ liệu lớn rất nhanh vượt qua giới hạn phần cứng và cần phải giải quyết bằng bài toán phân tán. Nửa đầu năm 2009, người ta đã manh nha thuật ngữ [NoSQL](http://thegioilaptrinh.net/co-so-du-lieu-nosql-khong-the-thieu-cho-ung-dung-web-1350-1-1.html) đánh dấu sự trưởng thành của thế hệ database mới trong khi những sản phẩm phần mềm có thể đã được phát triển từ trước đó rất lâu.

## Kiến trúc

* Các RDBMs hiện tại đã bộc lộ những yếu kém như việc đánh chỉ mục một lượng lớn dữ liệu, phân trang, hoặc phân phối luồng dữ liệu media (phim, ảnh, nhạc, ...). Cơ sở dữ liệu quan hệ được thiết kế cho những mô hình dữ liệu nhỏ thường xuyên đọc viết trong khi các Social Network Services lại có một lượng dữ liệu cực lớn và cập nhật liên tục do số lượng người dùng quá nhiều ở một thời điểm. Thiết kế trên Distributed NoSQL giảm thiểu tối đa các phép tính toán, I/O liên quan kết hợp với batch processing đủ đảm bảo được yêu cầu xử lý dữ liệu của các mạng dịch vụ dữ liệu cộng đồng này. Facebook, Amazon là những ví dụ điểm hình.
* Về cơ bản, các thiết kế của NoSQL lựa chọn mô hình lưu trữ tập dữ liệu theo cặp giá trị key-value. Khái niệm node được sử dụng trong quản lý dữ liệu phân tán. Với các hệ thống phân tán, việc lưu trữ có chấp nhận trùng lặp dữ liệu. Một request truy vấn tới data có thể gửi tới nhiều máy cùng lúc, khi một máy nào nó bị chết cũng không ảnh hưởng nhiều tới toàn bộ hệ thống. Để đảm bảo tính real time trong các hệ thống xử lý lượng lớn, thông thường người ta sẽ tách biệt database ra làm 2 hoặc nhiều database. Một database nhỏ đảm bảo vào ra liên tục, khi đạt tới ngưỡng thời gian hoặc dung lượng, database nhỏ sẽ được gộp (merge) vào database lớn có thiết kế tối ưu cho phép đọc (read operation). Mô hình đó cho phép tăng cường hiệu suất I/O - một trong những nguyên nhân chính khiến performance trở nên kém.

## Một ****số thuật ngữ liên quan****

* **Non-relational**: relational - ràng buộc - thuật ngữ sử dụng đến các mối quan hệ giữa các bảng trong cơ sở dữ liệu quan hệ (RDBMs) sử dụng mô hình khóa gồm 2 loại khóa: khóa chính và khóa phụ (primary key + foreign key) để ràng buộc dữ liệu nhằm thể hiện tính nhất quán dữ liệu từ các bảng khác nhau.   
  Non-relational là khái niệm không sử dụng các ràng buộc dữ liệu cho nhất quán dữ liệu ở NoSQL database.
* **Distributed storage**: mô hình lưu trữ phân tán các file hoặc dữ liệu ra nhiều máy tính khác nhau trong mạng LAN hoặc Internet  dưới sự kiểm soát của phần mềm.
* **Eventual consistency (nhất quán cuối)**: tính nhất quán của dữ liệu không cần phải đảm bảo ngay tức khắc sau mỗi phép write. Một hệ thống phân tán chấp nhận những ảnh hưởng theo phương thức lan truyền và sau một khoảng thời gian (không phải ngay tức khắc), thay đổi sẽ đi đến mọi điểm trong hệ thống, tức là cuối cùng (eventually) dữ liệu trên hệ thống sẽ trở lại trạng thái nhất quán.
* **Vertical scalable (khả năng mở rộng chiều dọc)**:  Khi dữ liệu lớn về lượng,  phương pháp tăng cường khả năng lưu trữ và xử lý bằng việc cải tiến phần mềm và cải thiện phần cứng trên một máy tính đơn lẻ được gọi là khả năng mở rộng chiều dọc. Ví dụ việc tăng cường CPUs, cải thiện đĩa cứng, bộ nhớ trong một máy tính,... cho DBMs nằm trong phạm trù này. Khả năng mở rộng chiều dọc còn có một thuật ngữ khác scale up.
* **Horizontal scalable (khả năng mở rộng chiều ngang)**: Khi dữ liệu lớn về lượng, phương pháp tăng cường khả năng lưu trữ và xử lý  là dùng nhiều máy tính phân tán. Phân tán dữ liệu được hỗ trợ bởi phần mềm tức cơ sở dữ liệu.   
    
  Trong khi giá thành phần cứng ngày càng giảm, tốc độ xử lý, bộ nhớ ngày càng tăng thì  horizontal scalable là một lựa chọn đúng đắn. Hàng trăm máy tính nhỏ được chập lại tạo thành một hệ thống tính toán mạnh hơn nhiều so với vi xử lý RISC truyền thống đơn lẻ. Mô hình này tiếp tục được hỗ trợ bởi các công nghệ kết nối Myrinet và InfiniBand. Từ đó chúng ta có thể quản lý, bảo trì từ xa, xây dựng batch procession (xử lý đồng loạt tập lệnh) tốt hơn. Do những đòi hỏi về tốc độ xử lý I/O cao, lượng cực lớn dữ liệu,...  scale horizontally sẽ thúc đẩy các công nghệ lưu trữ mới phát triển giống như object storage devices (OSD).

## Một số khái niệm mới NoSQL:

* **Fields** - tương đương với khái niệm Columns trong SQL
* **Document** - thay thế khái niệm row trong SQL. Đây cũng chính là khái niệm làm nên sự khác biệt giữa NoSQL và SQL, 1 document chứa số cột (fields) không cố định trong khi 1 row thì số cột(columns) là định sẵn trước.
* **Collection** - tương đương với khái niệm table trong SQL. Một collection là tập hợp các document. Điều đặc biệt là một collection có thể chứa các document hoàn toàn khác nhau.
* **Key-value** - cặp từ khóa - giá trị được dùng để lưu trữ dữ liệu trong NoSQL
* **Cursor** - tạm dịch là con trỏ. Chúng ta sẽ sử dụng cursor để lấy dữ liệu từ database.
* **Indexes** ~ counterparts  
  Trong các hệ cơ sở dữ liệu quan hệ, các cột được định nghĩa theo bảng còn với hệ cơ sở  dữ liệu không ràng buộc, các cột được định nghĩa ở mỗi document. Bởi thế, các document quản lí gần như tất cả, các collection không cần quản lí chặt chẽ những gì đang xảy ra trong nó nữa.

## Ưu nhược điểm của cơ sở dữ liệu NoSQL:

### Ưu điểm:

* **Nó là nguồn mở:** Các sản phẩm nguồn mở đưa ra cho những người phát triển với một vài lợi  ích lớn lao, trong đó có tình trạng không có chi phí của chúng. Những  lợi ích khác: phần mềm nguồn mở có xu hướng sẽ là tin cậy hơn, an ninh  hơn và nhanh hơn để triển khai so với các lựa chọn thay thế sở hữu độc  quyền.  
    
  Các hệ quản trị cơ sở dữ liệu (CSDL) NoSQL là Cassandra, CouchDB, Hbase, [MongoDB](http://www.thegioilaptrinh.net/bai-viet-332-1-1.html) và Redis.
* **Việc mở rộng phạm vi là mềm dẻo**: NoSQL thay thế câu thần chú cũ của các nhà quản trị CSDL về 'mở rộng  phạm vi' với một thứ mới: 'mở rộng ra ngoài'. Thay vì bổ sung thêm các  máy chủ lớn hơn để điều khiển nhiều tải dữ liệu hơn, thì CSDL NoSQL cho  phép một công ty phân tán tải qua nhiều máy chủ khi mà tải gia tăng.
* **Các CSDL NoSQL khác nhau cho những dự án khác nhau**: MongoDB và Redis là những lựa chọn tốt cho việc lưu trữ các dữ liệu  thống kê ít được đọc mà lại được viết thường xuyên, như một số đếm truy  cập web chẳng hạn.  
    
  Hadoop, một [CSDL](http://www.thegioilaptrinh.net/diendan-9-1.html) dạng tự do, phân tán làm tốt công việc lưu trữ các dữ  liệu lớn như các con số thống kê thời tiết hoặc công việc phân tích  nghiệp vụ.  
    
  Memcache, một CSDL nhất thời chóng tàn, tuyệt vời trong lưu trữ các  phiên làm việc web, các khóa, và các con số thống kê ngắn hạn.  
    
  Cassandra và Riak (các lưu trữ dư thừa, tự động tạo bó cluster) làm tốt  trong các môi trường với các ứng dụng có tính sẵn sàng cao, khi thời  gian sống tối đa là sống còn.
* **NoSQL được các hãng lớn sử dụng**: Các công ty như Amazon, BBC, Facebook và Google dựa vào các CSDL NoSQL.
* **NoSQL phù hợp với công nghệ đám mây**: NoSQL và đám mây là một sự trùng khớp tự nhiên. Các máy chủ ngày nay là  không đắt và có thể dễ dàng mở rộng phạm vi được theo yêu cầu có sử dụng  một dịch vụ như là Amazon EC2. Giống như tất cả công nghệ đám mây, EC2  dựa vào ảo hóa. Liên kết yếu của ảo hóa là sự thực thi của I/O, với bộ  nhớ và CPU các các kết nối mạnh.  
    
  Các CSDL NoSQL hầu hết sử dụng bộ nhớ qua đĩa như là vị trí ghi đầu tiên  - vì thế ngăn ngừa được sự thực thi không ổn định của I/O. Và vì NoSQL  lưu trữ dữ liệu thường thúc đẩy được tính mở rộng phạm vi theo chiều  ngang thông qua việc ngăn chia, chúng có khả năng tận dụng được việc  cung cấp mềm dẻo của đám mây.

### Nhược điểm:

* **Nguồn mở có thể có nghĩa là sự hỗ trợ không đồng đều cho các doanh nghiệp:** Trong khi các nhà cung cấp chủ chốt của RMBMS như Oracle, IBM hay Sybase  đưa ra sự hỗ trợ tốt nổi tiếng cho các khách hàng doanh nghiệp cỡ vừa,  thì các doanh nghiệp nhỏ hơn, thường là các nhà cung cấp nguồn mở mới  thành lập không thể được mong đợi cung cấp được sự hỗ trợ có thể so sánh  được - ngoại trừ một nhúm các khách hàng blue chip.  
    
  Nhà cung cấp nguồn mở trung bình thiếu sự tiếp cận toàn cầu, các dịch vụ hỗ trợ và sự tin cậy của một Oracle hay IBM.
* **Chưa đủ chín cho các doanh nghiệp**: Dù những triển khai của chúng tại một số công ty lớn, thì các CSDL NoSQL  vẫn đối mặt với một vấn đề về sự tin cậy chính với nhiều doanh nghiệp.  Điểm sống còn đối với sự thiếu của NoSQL về độ chín muồi và các vấn đề  về tính không ổn định có thể, trong khi trích ra tính chín muồi, giàu  chức năng và ổn định của các RDBMS được thiết lập đã từ lâu.
* **Những hạn chế về tri thức nghiệp vụ**: Có một vài câu hỏi xung quanh những khả năng về tri thức nghiệp vụ (BI)  của các CSDL NoSQL. Liệu các [CSDL](http://www.thegioilaptrinh.net/bai-viet-1216-1-1.html) này có thể cung cấp dạng phân tích dữ  liệu lớn và mạnh mà các doanh nghiệp đã quen với các RDBMS? Cần bao  nhiêu sự tinh thông về lập trình cần có để tiến hành những truy vấn và  phân tích hiện đại?  
    
  Các câu trả lời là không tích cực. Các CSDL NoSQL không có nhiều sự đeo  bám tới các công cụ BI thường được sử dụng, trong khi những yêu cầu và  phân tích hiện đại đơn giản nhất có liên quan khác nhiều tới sự tinh  thông về lập trình. Tuy vậy, các giải pháp là sẵn sàng. Quest Software,  ví dụ, đã tạo ra Toad cho các CSDL đám mây, mà nó phân phối các khả năng  truy vấn hiện đại tới một số CSDL NoSQL.
* **Thiếu sự tinh thông**: Tính rất mới mẻ của NoSQL có nghĩa là không có nhiều lập trình viên và  người quản trị mà biết công nghệ này - là cho khó khăn cho các công ty  tìm người với sự tinh thông phù hợp. Đối lại, thế giới của RDBMS có hàng  ngàn những người đủ tư cách.
* **Những vấn đề về tính tương thích**: Không giống như các CSDL quan hệ, các CSDL NoSQL chia sẻ ít theo cách  thức của các tiêu chuẩn. Mỗi CSDL NoSQL có các giao diện lập trình ứng  dụng API riêng của mình, các giao diện truy vấn độc nhất vô nhị, và  những sự riêng biệt. Sự thiếu hụt các tiêu chuẩn có nghĩa là nó không có  khả năng để chuyển một cách đơn giản từ một nhà cung cấp này sang một  nhà cung cấp khác, nếu bạn trở nên không hạnh phúc với dịch vụ.

## Cách triển khai một ứng dụng NoSQL:

# CHƯƠNG 3 – Tìm hiểu về các giải pháp cơ sở dữ liệu NoSQL

Có 4 loại cơ sở dữ liệu NoSQL chính:

## Wide Column Store / Column Families

## Key-Value Store / Tuple Store

## Document Store

## Graph Database

# CHƯƠNG 3- Tìm hiểu về RavenDB

## Lý do sử dụng RavenDB

## Đặc điểm của RavenDB

## Vài kĩ thuật đặc biệt khi sử dụng RavenDB

# CHƯƠNG 4 – Xây dựng ứng dụng sử dụng RavenDB

## Giới thiệu về ứng dụng

## Phân tích, thiết kế hệ thống

## Triển khai

## Kết quả test performance để thể hiện sức mạnh của NoSQL

# CHƯƠNG 5 – Tổng kết

## Kết quả đạt được

### Về mặt lý thuyết

### Về mặt thực nghiệm

## Hướng phát triển

# Tài liệu tham khảo