TRƯỜNG ĐẠI HỌC SƯ PHẠM TP HỒ CHÍ MINH

KHOA CÔNG NGHỆ THÔNG TIN

TP.HCM, ngày 30 tháng 10 năm 2018

Đề tài : Nghiêm cứu hệ cơ sở dữ liệu NoSQL – MongoDB

Nhóm thực hiện :

1.Nguyễn Tấn Duẩn – 42.01.104.214
2.Nguyễn Thái Thông – 42.01.104.214

GVHD: Thầy Lương Trần Huy Hiến

Hệ quản trị Cơ sở dữ liệu NoSQL MongoDB

I. Tổng quan về NoSQL

1. NoSQL là gì ?

1.1 Thuật ngữ

NoSQL 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 là SQL**

Đây là thuật ngữ chung cho các hệ CSDL không sử dụng mô hình dữ liệu quan hệ. NoSQL đặc biệt nhấn mạnh đến mô hình lưu trữ cặp giá trị - khóa và hệ thống lưu trữ phân tán.

1.2 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 cho truy vấn.

Vào năm 2009, Eric Evans, nhân viên của Rackspace giới thiệu lại thuật ngữ NoSQL khi Johan Oskarsson của Last.fm muốn tổ chức một hội thảo về cơ sở dữ liệu nguồn mở phân tán. Thuật ngữ NoSQL đánh dấu bước phát triển của thế hệ CSDL mới: phân tán (distributed) + không ràng buộc (non-relational).

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 cần phải giải quyết bằng bài toán phân tán.

Ghi chú: Một mệnh đề khá thú vị về non-relational data store: "*select fun, profit from real_world where relational=false;*".

1.3 Tốt hơn SQL

Các hệ CSDL quan hệ (RDBM) hiện tại bộc lộ những yếu kém trong những tác vụ như đánh chỉ mục một lượng lớn dữ liệu, phân trang, hoặc phân phối luồng dữ liệu media (phim, ảnh, nhạc, ...). CSDL quan hệ được thiết kế cho những mô hình dữ liệu không quá lớn trong khi các dịch vụ mạng xã hội lại có một lượng dữ liệu cực lớn và cập nhật liên tục do số lượng người dùng quá nhiều.

Thế hệ CSDL mới - NoSQL - giảm thiểu tối đa các phép tính toán, tác vụ đọc-ghi liên quan kết hợp với xử lý theo lô (batch processing) đảm bảo được yêu cầu xử lý dữ liệu của các dịch vụ mạng xã hội. Hệ CSDL này có thể lưu trữ, xử lý từ lượng rất nhỏ đến hàng petabytes dữ liệu với khả năng chịu tải, chịu lỗi cao nhưng chỉ đòi hỏi về tài nguyên phần cứng thấp.

NoSQL thiết kế đơn giản, nhẹ, gọn hơn so với RDBMs. Ngoài memory cached, dữ liệu nhỏ,… các NoSQL dạng này đặc biệt thích hợp cho thiết bị cầm nơi mà bộ nhớ và tốc độ xử lý hạn chế hơn so với máy tính thông thường. Khi khối lượng dữ liệu cần lưu trữ và lượng vào/ra cực lớn, RDBM đòi hỏi khắt khe và cao về phần cứng, chi phí thiết lập, vận hành đắt thì các mô hình lưu trữ phân tán trong NoSQL trở nên vượt trội. Thiết kế đặc biệt tối ưu về hiệu suất, tác vụ

đọc - ghi, ít đòi hỏi về phần cứng mạnh và đồng nhất, dễ dàng thêm bớt các node không ảnh hưởng tới toàn hệ thống …

Các mô hình dữ liệu đặc thù của NoSQL cũng cấp API tự nhiên hơn so với việc dùng RDBM.

Những ràng buộc về giấy phép sử dụng cùng với một khoản phí không nhỏ cũng là ưu thế. Chấp nhận NoSQL đồng nghĩa với việc bạn tham gia vào thế giới nguồn mở nơi mà bạn có khả năng tùy biến mạnh mẽ các sản phẩm, thư viện theo đúng mục đích của mình.

Bảng dưới đây đưa ra một số so sánh giữa RDBM và NoSQL.

Tính năng	CSDL quan hệ	NoSQL
Hiệu suất	Kém hơn SQL Relational giữa các table	Cực tốt Bỏ qua SQL Bỏ qua các ràng buộc dữ liệu
Khả năng mở rộng	Hạn chế về lượng.	Hỗ trợ một lượng rất lớn các node.
Hiệu suất đọc-ghi	Kém do thiết kế để đảm bảo sự vào/ra liên tục của dữ liệu	Tốt với mô hình xử lý lô và những tối ưu về đọc-ghi dữ liệu.
Thay đổi số node trong hệ thống	Phải shutdown cả hệ thống. Việc thay đổi số node phức tạp.	Không cần phải shutdown cả hệ thống. Việc thay đổi số node đơn giản, không ảnh hưởng đến hệ thống.
Phần cứng	Đòi hỏi cao về phần cứng.	Đòi hỏi thấp hơn về giá trị và tính đồng nhất của phần cứng

1.4 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).

2 Kiến trúc

2.1. Sơ lược.

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 keyvalue. 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 biệtdatabase 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.

2.2. Một số đặc điểm.

High Scalability: Gần như không có một giới hạn cho dữ liệu và người dùng trên hệ thống.

High Availability: Do chấp nhận sự trùng lặp trong lưu trữ nên nếu một node (commodity machine) nào đó bị chết cũng không ảnh hưởng tới toàn bộ hệ thống.

Atomicity: Độc lập data state trong các operation.

Consistency: chấp nhận tính nhất quán yếu, cập nhật mới không đảm bảo rằng các truy xuất sau đó thấy ngay được sự thay đổi. Sau một khoảng thời gian lan truyền thì tính nhất quán cuối cùng của dữ liệu mới được đảm bảo.

Durability: dữ liệu có thể tồn tại trong bộ nhớ máy tính nhưng đồng thời cũng được lưu trữ lại đĩa cứng.

Deployment Flexibility: việc bổ sung thêm/loại bỏ các node, hệ thống sẽ tự động nhận biết để lưu trữ mà không cần phải can thiệp bằng tay. Hệ thống cũng không đòi hỏi cấu hình phần cứng mạnh, đồng nhất.

Modeling flexibility: Key-Value pairs, Hierarchical data (dữ liệu cấu trúc), Graphs.

Query Flexibility: Multi-Gets, Range queries (load một tập giá trị dựa vào một dãy các khóa).

Phi quan hệ (hay không ràng buộc): relational - ràng buộc - thuật ngữ sử dụng đến các mối quan hệ giữa các bảng trong cơ sở dữ liệu quan hệ (RDBM) sử dụng mô hình 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.

Lưu trữ phân tán: mô hình lưu trữ phân tán các tập tin hoặc dữ liệu ra nhiều máy khác nhau trong mạng LAN hoặc Internet dưới sự kiểm soát của phần mềm.

Nhất quán cuối: tính nhất quán của dữ liệu không cần phải đảm bảo ngay tức khắc sau mỗi phép ghi. 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 để cuối cùng dữ liệu trên hệ thống sẽ trở lại trạng thái nhất quán.

Triển khai đơn giản, dễ nâng cấp và mở rộng.

Mô hình dữ liệu và truy vấn linh hoạt. …

2.3. What is NoSQL (technically speaking)?

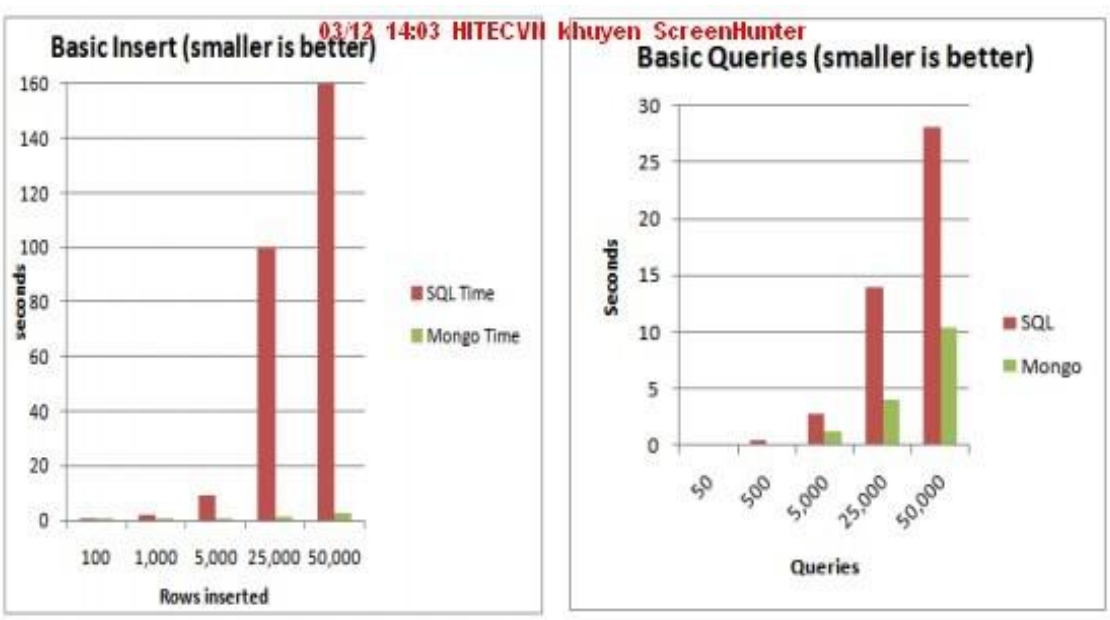
Có nhiều cách định nghĩa khác nhau và ở đây CTO của Amazon, Werner Vogels đề cập đến hệ thống Dynamo của họ đã gọi nó là một "highly available key-value store". Google gọi là BigTable để nhấn mạnh đây là "distributed storage system for managing structured data" (hệ thống lưu trữ và quản lý dữ liệu cấu trúc có phân tán).

Nó có thể xử lý một lượng dữ liệu cực lớn trong thời gian có hạn. 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.

Bằng việc bỏ qua thông dịch trong SQL cùng với những truy vấn rườm rà, NoSQL cho ta một kiến trúc tối ưu về tốc độ thực thi (ghi và truy vấn dữ liệu). 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).

Raffaele Sena, một senior computer scientist ở Adobe Systems Inc. đã nói rằng ConnectNow Web collaboration service của họ sử dụng Java clustering software từ Terracotta thay cho cơ sở dữ liệu quan hệ đã khiến "hệ thống của họ trở nên mạnh hơn, phức tạp hơn so với việc sử dụng cơ sở dữ liệu quan hệ".

Các thiết kế database có tính đặc thù (như document-oriented database) sẽ lược bỏ được tầng chuyển đổi sang mô hình lưu trữ quan hệ từ interface của nó đồng thời khiến giao tiếp tương tác trở nên tự nhiên hơn.



MongoDB vs. SQL Server 2008 Performance Showdown

Không quá cần thiết. 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 trong dự án ConnectNow của Adobe, dữ liệu người dùng trong một session không cần thiết phải lưu lại, chúng sẽ bị xóa khi người dùng logoff. Vì vậy, một keyvalue memory storage là đủ dùng.

2.4 Ứng dụng NoSQL

Nhiều người chấp nhận NoSQL là do vấn đề chi phí hoặc ý thức hệ, nói không với nguồn đóng. Việc đó cũng đồng nghĩa với việc chấp nhận sự non nớt và những hỗ trợ kém hơn. Nếu bạn vẫn thích thiết kế mô hình dữ liệu dạng bảng, CSDL SQL sẽ là lựa chọn.

NoSQL đặc biệt thích hợp cho các ứng dụng cực lớn (dịch vụ tìm kiếm, mạng xã hội ,…) và nhỏ. Với những ứng dụng vừa và lớn thì RDBMs vẫn thích hợp hơn.

Thiết kế NoSQL chấp nhận tính nhất quán yếu và có thể không dùng đến ‘transaction’. Với những ứng dụng đòi hỏi sự chặt chẽ của dữ liệu thì cần ‘transaction’ đảm bảo tính toàn

vẹn, cơ sở dữ liệu truyền thống là lựa chọn thích hợp hơn. NoSQL thích hợp cho các mô hình lưu trữ dữ liệu có tính đặc thù như object oriented, document oriented, xml database,…

Thường chúng ta sử dụng rất hạn chế những khả năng mà các CSDL RDBM cung cấp nhưng vẫn phải trả phí cho nó. Nếu không cần đến các tính năng cao cấp, không cần các chức năng của SQL hoặc rất ghét viết các câu lệnh SQL thì hãy nghĩ đến NoSQL.

II. MongoDB

1. Tổng quan về MongoDB

1.1. Lịch sử ra đời MongoDB

Phát triển **MongoDB** bắt đầu tại **10gen** (a software company) trong năm 2007, khi công ty xây dựng một Nền tảng như một dịch vụ tương tự như **Google App Engine** . Trong năm 2009, **MongoDB** trở thành mã nguồn mở như là một sản phẩm độc lập. với giấy phép **AGPL** .

Trong tháng 3 năm 2011, từ phiên bản 1.4, MongoDB đã hoàn thiện và sẵn sàng cho các ứng dụng.

Phiên bản ổn định mới nhất (tháng 3 năm 2012) là 2.0.3, phát hành vào tháng 2 năm 2012.

1.2. Các thành phần của MongoDB

Các tập tin thực thi của MongoDB:

MySQL executable	Oracle executable	Mongo executable
`mysqld`	`oracle`	`mongod`
`mysql`	`sqlplus`	`mongo`

Các khái niệm về Data và cấu trúc tổ chức dữ liệu(so sánh với MySQL):

MySQL term	Mongo term/concept
database	database
table	collection
index	index
row	BSON document
column	BSON field
join	embedding and linking
primary key	_id field
group by	aggregation

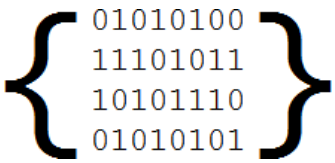
MongoDB truy vấn được thể hiện như các đối tượng **JSON**. Biểu đồ dưới đây cho thấy những ví dụ so sánh **MySQL** và Mongo về cú pháp ngôn ngữ truy vấn.

Các biểu thức truy vấn trong MongoDB có định dạng **JSON**:

SQL Statement	Mongo Statement
CREATE TABLE USERS (a Number, b Number)	db.createCollection ("mycoll")
ALTER USERS TABLE ADD ...	
INSERT INTO USERS VALUES (3,5)	db.users.insert ({a: 3, b: 5})
SELECT a,b FROM users	db.users.find ({}, {a: 1, b: 1})
SELECT * FROM users	db.users.find ()
SELECT * FROM users WHERE age=33	db.users.find ({age: 33})
SELECT a,b FROM users WHERE age=33	db.users.find({age:33}, {a:1,b:1})
SELECT * FROM users WHERE age=33 ORDER BY name	db.users.find({age:33}).sort({name:1})
SELECT * FROM users WHERE age>33	db.users.find ({name:{$gt:33}})
SELECT * FROM users WHERE age!=33	db.users.find ({name:{$ne:33}})
SELECT * FROM users WHERE name LIKE "%Joe%"	db.users.find ({name:Joe /})
SELECT * FROM users WHERE name LIKE "Joe%"	db.users.find({name:/^Joe/})
SELECT * FROM users WHERE age>33 AND age<=40	db.users.find({'age':{$gt:33,$lte:40}})
SELECT * FROM users ORDER BY name DESC	db.users.find().sort({name:-1})
SELECT * FROM users WHERE a=1 and b='q'	db.users.find ({a: 1, b: 'q'})
SELECT * FROM users LIMIT 10 SKIP 20	db.users.find().limit(10).skip(20)

2 Cấu trúc kiểu dữ liệu BSON

2.1. Giới thiệu về BSON



BSON viết tắt của Binary JSON là một cấu trúc nhị phân được mã hóa của các tài liệu giống như JSON. Giống như JSON, BSON hỗ trợ nhúng các tài liệu và mảng trong các tài liệu và các mảng khác. BSON cũng có phần mở rộng đó các loại dữ liệu mà không phải là một phần của JSON. Ví dụ: BSON có kiểu ngày và BinData.

2.2. Các kiểu dữ liệu cơ bản trong BSON

byte	1 byte (8-bits)
int32	4 bytes (32-bit signed integer)
int64	8 bytes (64-bit signed integer)
double	8 bytes (64-bit IEEE 754 floating point)

2.3. Quy định cụ thể các kiểu dữ liệu BSON

Trong BSON biểu diễn dữ liệu dạng ngữ nghĩa. **Ví dụ** \x01 là biểu diễn cho byte 0000 0001 dấu * trong biểu diễn kiểu của JSON có nghĩa là biểu thức đó được lặp lại n lần(n>=0) .

Ví dụ: "\x01"*2 -> \01\01 "\x01"* thì trong trường hợp này biểu thức này có thể ko tồn tại hoặc ko giới hạn

document	::=	int32 e_list "\x00"	**BSON Document**
e_list	::=	element e_list	**Sequence of elements**
	|	""	
element	::=	"\x01" e_name double	**Floating point**
	|	"\x02" e_name string	**UTF-8 string**

	| "\x03" e_name document	**Embedded document**
	| "\x04" e_name document	**Array**
	| "\x05" e_name binary	**Binary data**
	| "\x06" e_name	**Undefined — *Deprecated***
	| "\x07" e_name (byte*12)	**ObjectId**
	| "\x08" e_name "\x00"	**Boolean "false"**
	| "\x08" e_name "\x01"	**Boolean "true"**
	| "\x09" e_name int64	**UTC datetime**
	| "\x0A" e_name	**Null value**
	| "\x0B" e_name cstring cstring	**Regular expression**
	| "\x0C" e_name string (byte*12)	**DBPointer — *Deprecated***
	| "\x0D" e_name string	**JavaScript code**
	| "\x0E" e_name string	**Symbol**
	| "\x0F" e_name code_w_s	**JavaScript code w/ scope**
	| "\x10" e_name int32	**32-bit Integer**
	| "\x11" e_name int64	**Timestamp**
	| "\x12" e_name int64	**64-bit integer**
	| "\xFF" e_name	**Min key**
	| "\x7F" e_name	**Max key**
e_name	::= cstring	**Key name**
string	::= int32 (byte*) "\x00"	**String**
cstring	::= (byte*) "\x00"	**CString**

binary	::=	int32 subtype (byte*)	**Binary**
subtype	::=	"\x00"	**Binary / Generic**
	|	"\x01"	**Function**
	|	"\x02"	**Binary (Old)**
	|	"\x03"	**UUID**
	|	"\x05"	**MD5**
	|	"\x80"	**User defined**
code_w_s	::=	int32 string document	**Code w/ scope**

Ví dụ:

{"hello": "world"}	→	"\x16\x00\x00\x00\x02hello\x00 \x06\x00\x00\x00world\x00\x00"
{"BSON": ["awesome", 5.05,1986] **}**	→	"**1\x00\x00\x00**\x04BSON\x00&\x00 \x00\x00\x020\x00\x08\x00\x00 \x00awesome\x00\x011\x00333333 \x14@\x102\x00\xc2\x07\x00\x00 \x00**x00**"

II.4 Các đặc điểm của BSON

BSON được thiết kế để có ba đặc điểm sau đây:

- Dung lượng nhỏ: Giữ trên không không gian đến mức tối thiểu là quan trọng đối với bất kỳ định dạng biểu diễn dữ liệu, đặc biệt là khi được sử dụng qua mạng.
- Traversable: BSON thiết kế dễ dàng.

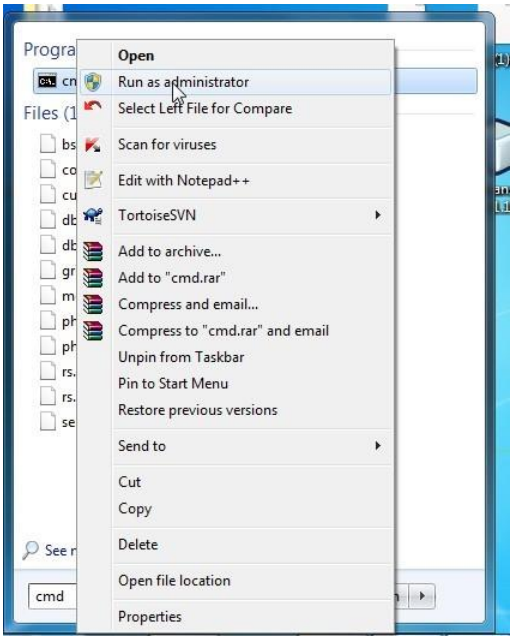
- Hiệu quả:Mã hóa dữ liệu đến BSON và giải mã từ BSON có thể được thực hiện rất nhanh chóng trong hầu hết các ngôn ngữ do việc sử dụng các loại dữ liệu C.

3. Hướng dẫn cài đặt và sử dụng MongoDB

- Download MongoDB
- 32bit MongoDB direct download | 64bit direct download
- Giải nén vào thư mục D:/MongoDb
- Thêm thư mục:

  thêm thư mục data (D:/MongoDB/data)

  thêm thư mục logs(D:/MongoDB/logs)
- Run **Window Command processor** bằng tài khoản administrator



- Cài đặt MongoDB

  mongod --logpath D:\MongoDB\logs\logfilename.log --logappend --dbpath D:\MongoDB\data –install

- Nếu trong tên file có ký tự đặc biệt hoặc dấu cách:

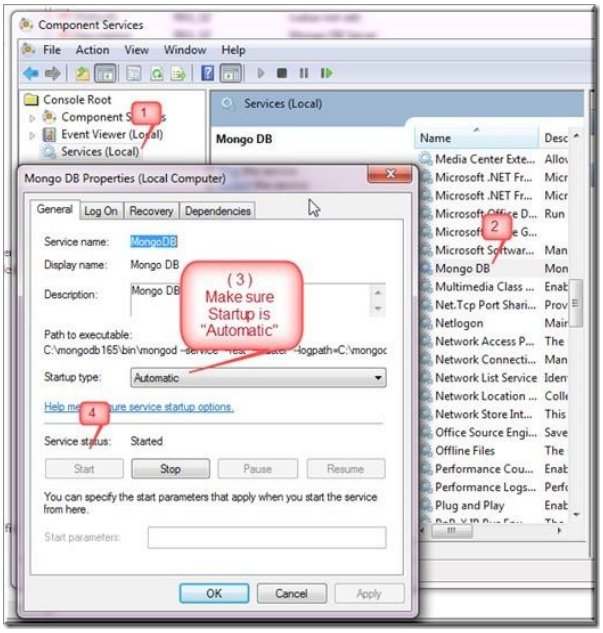
mongod --logpath "D:\MongoDB\logs\logfilename.log" --logappend --dbpath "D:\MongoDB\data" –install

- Đăng ký services

  mongod --logpath d:\mongo\logs\logfilename.log --logappend --dbpath d:\mongo\data –service

- Restart máy

- Quản lý service MongoDB



Hoặc có thể gõ lệnh ở **Window Command processor (**chạy dưới tài khoản Administrator**)**

Net start/stop/restart "Mongo DB"

- Chạy MongoDB: http://localhost:28017/

Khi thấy thông báo như trên là quá trình cài đặt đã thành công.

4. Sao lưu và phục hồi dữ liệu với MongoDB

4.1 Mongodump

các tham lựa chọn:

--help trợ giúp

-v [--verbose] liệt kê chi tiết

-h [--host] arg địa chỉ máy chủ mongo

-d [--db] arg cơ sở dữ liệu cần export

-c [--collection] arg collection cần export

-u [--username] arg username

-p [--password] arg password

--dbpath arg Chỉ thị này thay thế cho chỉ tường minh đến từng file. Tuy nhiên thử mục trỏ đến chỉ chứa các file của a databse.

--directoryperdb if dbpath được chỉ định, thì mỗi cơ sở dữ liệu sẽ chứa trong 1 thư mục nhất định

-o [--out] arg (=dump) thư mục chứa file dump

-q [--query] arg json query

Ví dụ:

$ ./mongodump --host prod.example.com

connected to: prod.example.com

all dbs

DATABASE: log to dump/log

log.errors to dump/log/errors.bson

713 objects

log.analytics to dump/log/analytics.bson

234810 objects

DATABASE: blog to dump/blog

blog.posts to dump/log/blog.posts.bson

59 objects

DATABASE: admin to dump/admin

với TH dùng cổng mặc định và local host chúng ta chỉ cần gõ

$ ./mongodump

Ví dụ: chỉ dump 1 collection

$ ./mongodump --db blog --collection posts

connected to: 127.0.0.1

DATABASE: blog to dump/blog

blog.posts to dump/blog/posts.bson

59 objects

4.2 mongorestore

Cú pháp: ./mongorestore [options] [directory or filename to restore from]

các chỉ thị:

--help	trợ giúp
-v [--verbose]	liệt kê chi tiết
-h [--host] arg	server mongo
-d [--db] arg	tên database muốn phục hồi
-c [--collection] arg	collection muốn phục hồi
-u [--username] arg	username
-p [--password] arg	password

Ví dụ:

Giả sử ta có database nằm trong thư mực D:/backup/db1 vào database DB

./mongorestore –d DB –dbpath D:/backup/db1

5. So sánh hiê êu năngMongoDB với Mysql

MongoDB test results:	**MySQL tets results:**
siege -f ./stress_urls.txt -c 300 -r 10 -d1 -i	*siege -f ./stress_urls_mysql.txt -c 300 -r 10 -d1 –i*
Transactions: 2994 hits Availability: 99.80 % Elapsed time: 11.95 secs Data transferred: 3.19 MB Response time: 0.26 secs Transaction rate: 250.54 trans/sec Throughput: 0.27 MB/sec Concurrency: 65.03 Successful transactions: 2994 Failed transactions: 6 Longest transaction: 1.47 Shortest transaction: 0.00	Transactions: 2832 hits Availability: 94.40 % Elapsed time: 23.53 secs Data transferred: 2.59 MB Response time: 0.74 secs Transaction rate: 120.36 trans/sec Throughput: 0.11 MB/sec Concurrency: 89.43 Successful transactions: 2832 Failed transactions: 168 Longest transaction: 16.36 Shortest transaction: 0.00