Mục lục

[**1.** **NoSql** 2](#_Toc71791710)

[**Vì sao nên sử dụng cơ sở dữ liệu NoSQL?** 3](#_Toc71791711)

[**2.** **MongoDB là gì** 4](#_Toc71791712)

[*2.1. Định nghĩa về MongoDB* 4](#_Toc71791713)

[*2.1. Các thuật ngữ hay sử dụng trong MongoDB* 5](#_Toc71791714)

[**3.** **Một số câu lệnh cơ bản và các kiểu dữ liệu trong mongoDB** 7](#_Toc71791715)

[*3.1. Các thuật ngữ hay sử dụng trong MongoDB* 7](#_Toc71791716)

[*3.2. Các thuật ngữ hay sử dụng trong MongoDB* 8](#_Toc71791717)

[**4 . Ưu điểm của mongoDB** 11](#_Toc71791718)

[**5 . Nhược điểm của mongoDB** 12](#_Toc71791719)

[**6. Khi nào NÊN và KHÔNG NÊN dung mongoDB** 13](#_Toc71791720)

[*6.1. Sử dụng MongoDB trong trương hợp:* 13](#_Toc71791721)

[*6.2. Không Nên sử dụng MongoDB trong trương hợp:* 13](#_Toc71791722)

[**7. Cài đặt MongoDB** 13](#_Toc71791723)

MongoDB là gì? Cơ sở dữ liệu phi quan hệ



Bài viết này giúp anh có cái nhìn tổng quan về MongoDB. Chúng ta không lạ gì với cơ sở dữ liệu quan hệ, còn với cơ sở dữ liệu phi quan hệ thì sao? **MEAN stack** (MongoDB, Express, AngularJS, NodeJS) đang dần thay thế cho **LAMP stack** (Linux, Apache, MySQL, PHP/Python). Để tìm hiểu về MongoDB thì đầu tiên chúng ta sẽ cùng tìm hiểu xem NoSql (Phi quan hệ) là gì

1. **NoSql**

Đầu tiên về cơ sở dữ liệu phi quan hệ chúng ta cùng tìm hiểu về NoSql nhé, vậy NoSql là gì?.

* NoSQL là 1 dạng CSDL mã nguồn mở và được viết tắt bởi: None-Relational SQL hay có nơi thường gọi là Not-Only SQL.
* NoSQL được phát triển trên Javascript Framework với kiểu dữ liệu là JSON và dạng dữ liệu theo kiểu key và value.
* NoSQL ra đời như là 1 mảnh vá cho những khuyết điểm và thiếu xót cũng như hạn chế của mô hình dữ liệu quan hệ RDBMS (Relational Database Management System - Hệ quản trị cơ sở dữ liệu quan hệ) về tốc độ, tính năng, khả năng mở rộng,...
* Với NoSQL bạn có thể mở rộng dữ liệu mà không lo tới những việc như tạo khóa ngoại, khóa chính, kiểm tra ràng buộc .v.v ...
* NoSQL bỏ qua tính toàn vẹn của dữ liệu và transaction để đổi lấy hiệu suất nhanh và khả năng mở rộng.
* NoSQL được sử dụng ở rất nhiều công ty, tập đoàn lớn, ví dụ như FaceBook sử dụng Cassandra do FaceBook phát triển, Google phát triển và sử dụng BigTable,...

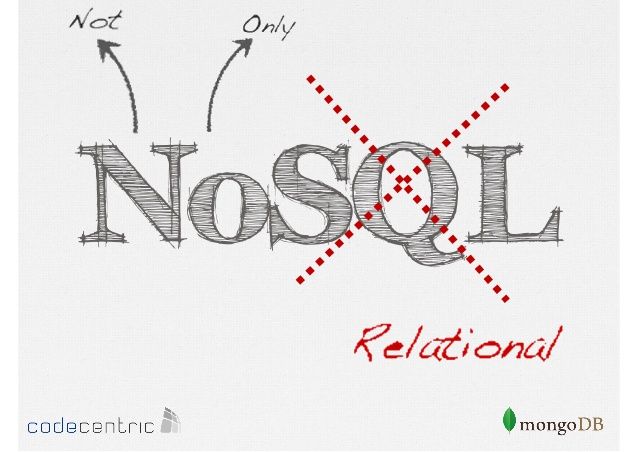
# **Vì sao nên sử dụng cơ sở dữ liệu NoSQL?**

Cơ sở dữ liệu NoSQL là lựa chọn cực kỳ thích hợp cho nhiều ứng dụng hiện đại, ví dụ như di động, web và trò chơi đòi hỏi phải sử dụng cơ sở dữ liệu cực kỳ thiết thực, linh hoạt, có khả năng thay đổi quy mô và hiệu năng cao để đem đến cho người dùng trải nghiệm tuyệt vời.

* Linh hoạt: Cơ sở dữ liệu NoSQL thường cung cấp các sơ đồ linh hoạt giúp công đoạn phát triển nhanh hơn và có khả năng lặp lại cao hơn. Mô hình dữ liệu linh hoạt biến cơ sở dữ liệu NoSQL thành lựa chọn lý tưởng cho dữ liệu không được tổ chức thành cấu trúc hoặc có cấu trúc chưa hoàn chỉnh.
* Khả năng thay đổi quy mô: Cơ sở dữ liệu NoSQL thường được thiết kế để tăng quy mô bằng cách sử dụng các cụm phần cứng được phân phối thay vì tăng quy mô bằng cách bổ sung máy chủ mạnh và tốn kém. Một số nhà cung cấp dịch vụ đám mây xử lý các hoạt động này một cách không công khai dưới dạng dịch vụ được quản lý đầy đủ.
* Hiệu năng cao: Cơ sở dữ liệu NoSQL được tối ưu hóa theo các mô hình dữ liệu cụ thể và các mẫu truy cập giúp tăng hiệu năng cao hơn so với việc cố gắng đạt được mức độ chức năng tương tự bằng cơ sở dữ liệu quan hệ.
* Cực kỳ thiết thực: Cơ sở dữ liệu NoSQL cung cấp các API và kiểu dữ liệu cực kỳ thiết thực được xây dựng riêng cho từng mô hình dữ liệu tương ứng.

Đến đây cũng phần nào hiểu về NoSql rồi, giờ chúng ta đi đến khái niệm MongoDB.

1. **MongoDB là gì**



### *2.1. Định nghĩa về MongoDB*

* MongoDB là một hệ quản trị cơ sở dữ liệu mã nguồn mở, là CSDL thuộc NoSql và được hàng triệu người sử dụng.
* MongoDB là một database hướng tài liệu (document), các dữ liệu được lưu trữ trong document kiểu JSON thay vì dạng bảng như CSDL quan hệ nên truy vấn sẽ rất nhanh.
* Với CSDL quan hệ chúng ta có khái niệm bảng, các cơ sở dữ liệu quan hệ (như MySQL hay SQL Server...) sử dụng các bảng để lưu dữ liệu thì với MongoDB chúng ta sẽ dùng khái niệm là **collection** thay vì bảng
* So với RDBMS thì trong MongoDB **collection** ứng với **table**, còn **document** sẽ ứng với **row** , MongoDB sẽ dùng các document thay cho row trong RDBMS.
* Các collection trong MongoDB được cấu trúc rất linh hoạt, cho phép các dữ liệu lưu trữ không cần tuân theo một cấu trúc nhất định.
* Thông tin liên quan được lưu trữ cùng nhau để truy cập truy vấn nhanh thông qua ngôn ngữ truy vấn MongoDB

Vậy là mình đã giới thiệu xong khái niệm và một số điểm khác cơ bản của MongoDB so với CSDL quan hệ rồi, tiếp theo sẽ là một vài thao tác với MongoDB để các bạn có thể hình dung rõ hơn nhé

### *2.1. Các thuật ngữ hay sử dụng trong MongoDB*

* **id** – Là trường bắt buộc có trong mỗi document. Trường \_id đại diện cho một giá trị duy nhất trong document MongoDB. Trường \_id cũng có thể được hiểu là khóa chính trong document. Nếu bạn thêm mới một document thì MongoDB sẽ tự động sinh ra một \_id đại diện cho document đó và là duy nhất trong cơ sở dữ liệu MongoDB.
* **Collection** – Là nhóm của nhiều document trong MongoDB. Collection có thể được hiểu là một bảng tương ứng trong cơ sở dữ liệu RDBMS (Relational Database Management System). Collection nằm trong một cơ sở dữ liệu duy nhất. Các collection không phải định nghĩa các cột, các hàng hay kiểu dữ liệu trước.
* **Cursor** – Đây là một con trỏ đến tập kết quả của một truy vấn. Máy khách có thể lặp qua một con trỏ để lấy kết quả.
* **Database** – Nơi chứa các Collection, giống với cơ sở dữ liệu RDMS chúng chứa các bảng. Mỗi Database có một tập tin riêng lưu trữ trên bộ nhớ vật lý. Một mấy chủ MongoDB có thể chứa nhiều Database.
* **Document** – Một bản ghi thuộc một Collection thì được gọi là một Document. Các Document lần lượt bao gồm các trường tên và giá trị.
* **Field** – Là một cặp name – value trong một document. Một document có thể có không hoặc nhiều trường. Các trường giống các cột ở cơ sở dữ liệu quan hệ.
* **JSON** – Viết tắt của JavaScript Object Notation. Con người có thể đọc được ở định dạng văn bản đơn giản thể hiện cho các dữ liệu có cấu trúc. Hiện tại JSON đang hỗ trợ rất nhiều ngôn ngữ lập trình.
* **Index** – Là những cấu trúc dữ liệu đặc biệt, dùng để chứa một phần nhỏ của các tập dữ liệu một cách dễ dàng để quét. Chỉ số lưu trữ giá trị của một fields cụ thể hoặc thiết lập các fields, sắp xếp theo giá trị của các fields này. Index hỗ trợ độ phân tích một cách hiệu quả các truy vấn. Nếu không có chỉ mục, MongoDB sẽ phải quét tất cả các documents của collection để chọn ra những document phù hợp với câu truy vấn. Quá trình quét này là không hiệu quả và yêu cầu MongoDB để xử lý một khối lượng lớn dữ liệu.

Hãy lưu ý sự khác biệt của các trường và \_id trong một document. Một \_id được dùng để đại diện cho một document và chúng được sinh ra khi thêm một Document vào Collection.

1. **Một số câu lệnh cơ bản và các kiểu dữ liệu trong mongoDB**

### *3.1. Các thuật ngữ hay sử dụng trong MongoDB*

| **CSDL** | **MySQL** | **MongoDB** |
| --- | --- | --- |
| Tạo csdl | CREATE DATABASE test; | use test; |
| Tạo bảng | CREATE TABLE students (ten\_cot - kieu\_du\_lieu); | db.createCollection('students'); |
| Tạo bản ghi | INSERT INTO studetns ('name', 'gender') VALUES('thanh', 'male'); | db.students.insert({ name:'thanh', gender: 'male'}); |
| Cập nhật | UPDATE students SET name = 'thanh update' WHERE id = 1; | db.students.update({ \_id: 1 },{$set:{ name: 'thanh update' }}); |
| Xóa bản ghi | DELETE FROM students Where id = 1; | db.students.remove({ \_id: 1}); |
| Tìm kiếm all | SELECT \* FROM students; | db.students.find({}); |
| Tìm kiếm | SELECT \* FROM students WHERE name = 'thanh'; | db.students.find({ name: 'thanh' }); |

### *3.2. Các kiểu dữ liệu sử dụng trong MongoDB*

MongoDB hỗ trợ các kiểu dữ liệu sau:

| **Type** | **Number** | **Alias** |
| --- | --- | --- |
| Double | 1 | “double” |
| String | 2 | “string” |
| Object | 3 | “object” |
| Array | 4 | “array” |
| Binary data | 5 | “binData” |
| Undefined | 6 | “undefined” |
| ObjectId | 7 | “objectId” |
| Boolean | 8 | “bool” |
| Date | 9 | “date” |
| Null | 10 | “null” |
| Regular Expression | 11 | “regex” |
| DBPointer | 12 | “dbPointer” |
| JavaScript | 13 | “javascript” |
| Symbol | 14 | “symbol” |
| JavaScript (with scope) | 15 | “javascriptWithScope” |
| 32-bit integer | 16 | “int” |
| Timestamp | 17 | “timestamp” |
| 64-bit integer | 18 | “long” |
| Decimal128 | 19 | “decimal” |
| Min key | -1 | “minKey” |
| Max key | 127 | “maxKey” |

* **Chuỗi:** Đây là kiểu dữ liệu được sử dụng phổ biến nhất để lưu giữ dữ liệu. Chuỗi trong MongoDB phải là UTF-8 hợp lệ.
* **Số nguyên:** Kiểu dữ liệu này được sử dụng để lưu một giá trị số. Số nguyên có thể là 32 bit hoặc 64 bit phụ thuộc vào Server của bạn.
* **Boolean**: Kiểu dữ liệu này được sử dụng để lưu giữ một giá trị Boolean (true/false).
* **Double**: Kiểu dữ liệu này được sử dụng để lưu các giá trị số thực dấu chấm động.
* **Min/ Max keys:** Kiểu dữ liệu này được sử dụng để so sánh một giá trị với các phần tử BSON thấp nhất và cao nhất.
* **Mảng:** Kiểu dữ liệu này được sử dụng để lưu giữ các mảng hoặc danh sách hoặc nhiều giá trị vào trong một key.
* **Timestamp:** Đánh dấu thời điểm một Document được sửa đổi hoặc được thêm vào.
* **Object:** Kiểu dữ liệu này được sử dụng cho các Document được nhúng vào.
* **Null:** Kiểu dữ liệu này được sử dụng để lưu một giá trị Null.
* **Symbol:** Kiểu dữ liệu này được sử dụng giống như một chuỗi
* **Date :** Kiểu dữ liệu này được sử dụng để lưu giữ date và time hiện tại trong định dạng UNIX time.
* **Object ID:** Kiểu dữ liệu này được sử dụng để lưu giữ ID của Document.
* **Binary data:** Kiểu dữ liệu này được sử dụng để lưu giữ dữ liệu nhị phân.
* **Code:** Kiểu dữ liệu này được sử dụng để lưu giữ JavaScrip code vào trong Document.
* **Regular expression:** Kiểu dữ liệu này được sử dụng để lưu giữ Regular Expresion.

*Ví dụ :* Dữ liệu của collection bao gồm 2 documents MinKey và MaxKey: { "\_id" : 1, x : { "$minKey" : 1 } } { "\_id" : 2, y : { "$maxKey" : 1 } }

Câu lệnh truy vấn sau sẽ cho kết quả là một documents có \_id: 1:

db.data.find( { x: { $type: "minKey" } } )

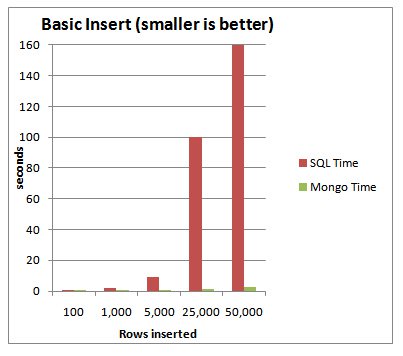
hoặc có thể viết

db.data.find( { x: { $type: "-1" } } )

(\*) <https://docs.mongodb.com/manual/reference/operator/query/type/>

**4 . Ưu điểm của mongoDB**

* Do MongoDB sử dụng lưu trữ dữ liệu dưới dạng Document JSON nên mỗi một collection sẽ có các kích cỡ và các document khác nhau, linh hoạt trong việc lưu trữ dữ liệu, nên bạn muốn gì thì cứ insert vào thoải mái.
* Dữ liệu trong MongoDB không có sự ràng buộc lẫn nhau, không có join như trong RDBMS nên khi insert, xóa hay update nó không cần phải mất thời gian kiểm tra xem có thỏa mãn các ràng buộc dữ liệu như trong RDBMS.
* MongoDB rất dễ mở rộng (Horizontal Scalability). Trong MongoDB có một khái niệm cluster là cụm các node chứa dữ liệu giao tiếp với nhau, khi muốn mở rộng hệ thống ta chỉ cần thêm một node với vào cluster:
* Trường dữ liệu “\_id” luôn được tự động đánh index (chỉ mục) để tốc độ truy vấn thông tin đạt hiệu suất cao nhất.
* Khi có một truy vấn dữ liệu, bản ghi được cached lên bộ nhớ Ram, để phục vụ lượt truy vấn sau diễn ra nhanh hơn mà không cần phải đọc từ ổ cứng.
* Hiệu năng cao: Tốc độ truy vấn (find, update, insert, delete) của MongoDB nhanh hơn hẳn so với các hệ quản trị cơ sở dữ liệu quan hệ (RDBMS). Với một lượng dữ liệu đủ lớn thì thử nghiệm cho thấy tốc độ insert của MongoDB có thể nhanh tới gấp 100 lần so với MySQL



**5 . Nhược điểm của mongoDB**

* Một ưu điểm của MongoDB cũng chính là nhược điểm của nó. MongoDB không có các tính chất ràng buộc như trong RDBMS nên khi thao tác với mongoDB thì phải hết sức cẩn thận.
* Tốn bộ nhớ do dữ liệu lưu dưới dạng key-value, các collection chỉ khác về value do đó key sẽ bị lặp lại. Không hỗ trợ join nên dễ bị dữ thừa dữ liệu.
* Khi insert/update/remove bản ghi, MongoDB sẽ chưa cập nhật ngay xuống ổ cứng, mà sau 60 giây MongoDB mới thực hiện ghi toàn bộ dữ liệu thay đổi từ RAM xuống ổ cứng điêù này sẽ là nhược điểm vì sẽ có nguy cơ bị mất dữ liệu khi xảy ra các tình huống như mất điện...

**6. Khi nào NÊN và KHÔNG NÊN dung mongoDB**

### *6.1. Sử dụng MongoDB trong trương hợp:*

* Nếu website của bạn có tính chất INSERT cao Bởi vì mặc định MongoDB có sẵn cơ chế ghi với tốc độ cao và an toàn.Website của bạn ở dạng thời gian thực nhiều, nghĩa là nhiều người thao tác với ứng dung. Nếu trong quá trình load bị lỗi tại một điểm nào đó thì nó sẽ bỏ qua phần đó nên sẽ an toàn.
* Website bạn có nhiều dữ liệu quá Giả sử web bạn có đến 10 triệu records thì đó là cơn ác mộng với MYSQL. Bởi vì MongoDB có khả năng tìm kiến thông tin liên quan cũng khá nhanh nên trường hợp này nên dùng nó.
* Máy chủ không có hệ quản trị CSDL Trường hợp này thường bạn sẽ sử dụng SQLITE hoặc là MongoDB.

### *6.2. Không Nên sử dụng MongoDB trong trương hợp:*

* Các ứng dụng cần sử dụng nhiều transaction (như ngân hàng) do Mongodb không có cơ chế transaction (giao dịch) để phục vụ cho các ứng dụng ngân hàng
* Các ứng dụng cần SQL (sử dụng joins).

**7. Cài đặt MongoDB**

**Trên Windows:**

1. Tải MongoDB từ link bên dưới: <https://www.mongodb.org/downloads>
2. Chạy file vừa tải về
3. Sau khi chạy xong file, vào thư mục C:\Program Files\MongoDB\Server\3.4\bin , chạy file mongod để khởi động mongoDB
4. Dùng địa chỉ 127.0.0.1:27017 để tạo connection đến mongoDB

**- Trên Linux:**

Tham khảo trên website : [install](https://www.youtube.com/watch?v=1uFY60CESlM)

hoặc tham khảo video cài đặt tại: [link](https://www.youtube.com/watch?v=1uFY60CESlM)

**Link tham khảo:**

<http://expressmagazine.net/development/2330/dinh-nghia-mongodb-nosql-la-gi> <https://viblo.asia/p/tim-hieu-ve-mongodb-4P856ajGlY3#_8-khi-nao-nen-su-dung-mongodb--7> <https://stackjava.com/mongodb/uu-nhuoc-diem-cua-mongodb-khi-nao-nen-dung-mongodb.html>