33333

TRƯỜNG ĐẠI HỌC SƯ PHẠM THÀNH PHỐ HỒ CHÍ MINH

KHOA CÔNG NGHỆ THÔNG TIN

**--🕮--**

**MÔN: CƠ SỞ DỮ LIỆU NÂNG CAO**

**BÁO CÁO CUỐI KÌ MONGODB**



**Giảng viên hướng dẫn:** Ths.Lương Trần Hy Hiến

**Nhóm sinh viên thực hiện:**

1. Nguyễn Công Khanh - 43.01.104.075
2. Ngô Xuân Hải - 43.01.104.041
3. Tạ Hoàng Trí - 43.01.104.189
4. Hoàng Văn Quý - 43.01.104.143
5. Nguyễn Văn Hiếu - 43.01.104.049

TP.Hồ Chí Minh, ngày 21 tháng 11 năm 2019

Mục Lục

[1. Giới thiệu về MongoDB 1](#_Toc25746665)

[2. Các thuật ngữ hay sử dụng trong MongoDB 1](#_Toc25746666)

[3. So sánh giữa RDBMS (Relational database management system) và MongoDB 3](#_Toc25746667)

[4. Các kiểu dữ liệu trong MongoDB 3](#_Toc25746668)

[6. Thiết kế Schema trong MongoDB 5](#_Toc25746669)

[7. Relationship trong MongoDB 7](#_Toc25746670)

* [Mô hình hóa Embeded Relationships 7](#_Toc25746671)
* [Mô hình hóa Referenced Relationship 8](#_Toc25746672)

[8. Tham chiếu Database trong MongoDB 8](#_Toc25746673)

[9. Chỉ mục (Index) trong MongoDB 9](#_Toc25746674)

[10. Phương thức ensureIndex() trong MongoDB 10](#_Toc25746675)

[11. Covered Query trong MongoDB 11](#_Toc25746676)

[12. MongoDB hoạt động như thế nào: 13](#_Toc25746677)

* [Ưu điểm 13](#_Toc25746678)
* [Nhược điểm 14](#_Toc25746679)

[13. Lợi thế của MongoDB 14](#_Toc25746680)

[14. Khi nào NÊN sử dụng MongoDB? 16](#_Toc25746681)

[15. Khi nào KHÔNG NÊN sử dụng MongoDB? 16](#_Toc25746682)

[16. Các công cụ quản trị MongoDB 16](#_Toc25746683)

[17. Cài đặt MongoDB 16](#_Toc25746684)

[18. Một số ví dụ về mongo 17](#_Toc25746685)

[19. Tổng kết và đánh giá: 21](#_Toc25746686)

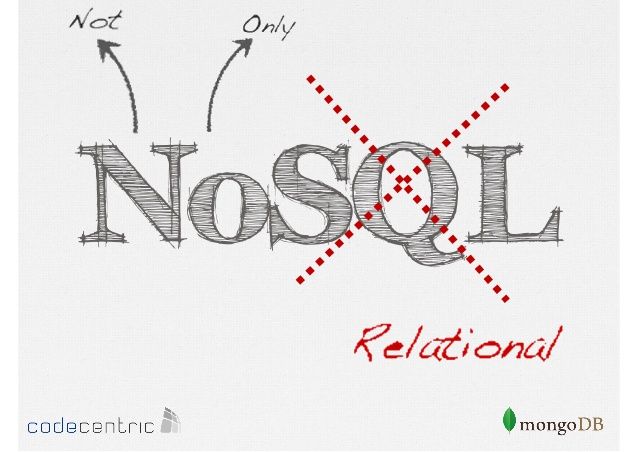
* [Tổng kết 21](#_Toc25746687)
* [Đánh giá 21](#_Toc25746688)

# 1. Giới thiệu về MongoDB

MongoDB là một cơ sở dữ liệu mã nguồn mở và là cơ sở dữ liệu NoSQL hàng đầu, được hàng triệu người sử dụng. MongoDB được viết bằng C++ và được thiết kế theo kiểu hướng đối tượng trong đó các bảng được cấu trúc một cách linh hoạt cho phép các dữ liệu lưu trên bảng không cần phải tuân theo một dạng cấu trúc nhất định nào.

NoSQL là 1 dạng CSDL mã nguồn mở không sử dụng Transact-SQL để truy vấn thông tin.NoSQL viết tắt bởi: None-Relational SQL, hay có nơi thường gọi là Not-Only SQL.CSDL này được phát triển trên Javascript Framework với kiểu dữ liệu JSON. (Cú pháp của JSON là “key:value”)

Ngoài ra, MongoDB là một cơ sở dữ liệu đa nền tảng, hoạt động trên các khái niệm Collection và Document, nó cung cấp hiệu suất cao, tính khả dụng cao và khả năng mở rộng dễ dàng



# 2. 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.

# 3. So sánh giữa RDBMS (Relational database management system) và MongoDB

|  |  |
| --- | --- |
| RDBMS | MongoDB |
| Database | Database |
| Table | Collection |
| Tuple/Row | Document |
| Column | Field |
| Table Join | Embedded Documents |
| Primary Key | Primary Key (mặc định là \_id) |

# 4. Các kiểu dữ liệu 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” |
| 32-bit integer | 16 | “int” |
| Timestamp | 17 | “Timestamp” |
| 64-bit integer | 18 | “long” |
| Min key | -1 | “minKey” |
| Max key | 123 | “maxkey” |

5. Một số câu lệnh dùng trong MongoDB

|  |  |
| --- | --- |
| Hiển thị database | show dbs |
| Tạo database | use database\_name |
| Xóa database | db.dropDatabase() |
| Tạo collection | db.createCollection(name, options) |
| Drop collection | db.COLLECTION\_NAME.drop() |
| Insert document | db.COLLECTION\_NAME.insert(document) |
|  | db.COLLECTION\_NAME.insertOne(document) |
|  | db.COLLECTION\_NAME.insertMany(document) |
| Truy vấn Document | db.COLLECTION\_NAME.find(query) |
|  | db.COLLECTION\_NAME.findOne(query) |
| Cập nhật Document | db.COLLECTION\_NAME.update(query, update, options) |
|  | db.COLLECTION\_NAME.updateOne(query, update, options) |
|  | db.COLLECTION\_NAME.updateMany(query, update, options) |
| Save | db.COLLECTION\_NAME.save({\_id:ObjectId(),NEW\_DATA}) |
| Xóa documnet | db.COLLECTION\_NAME.remove(query) |
| Giới hạn bản ghi | db.COLLECTION\_NAME.find().limit(NUMBER) |
| Sắp xếp bản ghi | db.COLLECTION\_NAME.find().sort({KEY:1}) |
| Tạo chỉ mục (index) | db.COLLECTION\_NAME.ensureIndex({KEY:1}) |
| …. |  |

+ query: tiêu chí để update.

+ update: document muốn update.

+ options: các document khác không bắt buộc.

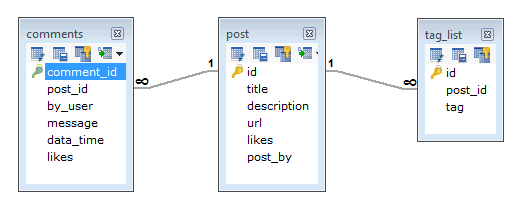
|  |  |
| --- | --- |
| **Phép toán** | **Cú pháp** |
| Bằng(Equality) | {key: value} |
| Nhỏ hơn (Less Than) | {key: {$lt: value}} |
| Nhỏ hơn bằng (Less Than Equals) | {key: {$lte: value}} |
| Lơn hơn (Greater Than) | {key: {$gt: value}} |
| Lớn hơn bằng (Greater Than Equals) | {key: {$gte: value}} |
| Khác (Not Equals) | {key: {$ne: value}} |
| Trong ( In) | {key: {$in: [value1, value2,..]}} |
| Không Thuộc (Not In) | {key: {$nin: [value1, value2,..]}} |

# 6. Thiết kế Schema trong MongoDB

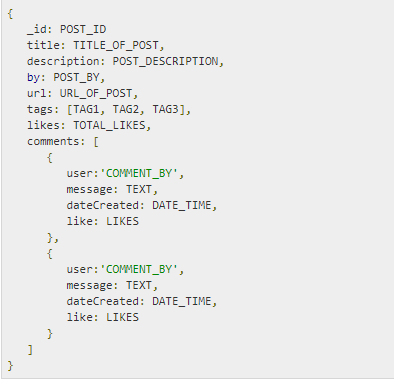
Giả sử, một khách hàng cần một thiết kế cơ sở dữ liệu cho trang blog của anh ta, và dưới đây, bạn xem các điểm khác nhau giữa thiết kế Schema của RDBMS và MongoDB. Website này có các yêu cầu sau:

* Mỗi post có tiêu đề, miêu tả và Url duy nhất.
* Mỗi post có thể có một hoặc nhiều tags.
* Mỗi post có tên người đăng và tổng số like.
* Mỗi post có các comment được cung cấp bởi người dùng cùng với tên, thông điệp, thời gian, và like của họ.
* Trên mỗi post, có thể có 0 hoặc nhiều comment.

Trong thiết kế Schema của RDBMS cho các yêu cầu trên sẽ có tối thiểu ba bảng dữ liệu:



Trong khi trong thiết kế Schema của MongoDB sẽ chỉ có một Collection Post có cấu trúc như sau:



# 7. Relationship trong MongoDB

Relationship trong MongoDB tượng trưng cho cách các Document có mối liên quan với nhau. Relationship có thể được mô hình hóa thông qua phương thức Embeded và Referenced. Những Relationship này có thể là 1:1, 1:N, N:1, hoặc N:N.

Chúng ta cùng xem xét trường hợp lưu giữ địa chỉ của người dùng. Một người dùng có thể có nhiều địa chỉ, điều này tạo ra một 1:N Relationship.

## Mô hình hóa Embeded Relationships



Phương pháp này duy trì tất cả dữ liệu có liên quan trong một Document đơn, điều này giúp cho việc lấy và duy trì dữ liệu dễ dàng. Toàn bộ Document có thể được lấy trong một truy vấn đơn, giống như:

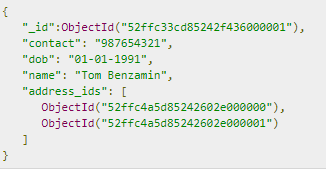
>db.users.findOne({"name":"Tom Benzamin"},{"address":1})

Ghi chú rằng, trong truy vấn trên, **db** và **users** tương ứng là Database và Collection.

Điểm hạn chế ở đây là, nếu Document được nhúng tiếp tục tăng kích cỡ quá nhiều, nó sẽ ảnh hưởng đến hiệu suất đọc/ghi.

## Mô hình hóa Referenced Relationship

Đây là phương pháp thiết kế Relationship tiêu chuẩn hóa. Trong phương pháp này, cả user và address document sẽ vẫn được duy trì một cách riêng rẽ, nhưng user document sẽ chứa một trường mà sẽ tham chiếu đến trường id của address document.



Như trên, user document chứa trường **address\_ids**, mà chứa ObjectIds của địa chỉ tương ứng. Sử dụng các ObjectIds này, chúng ta có thể truy vấn address document và lấy chi tiết địa chỉ từ đó. Với hướng tiếp cận này, chúng ta sẽ cần hai truy vấn: đầu tiên lấy các trường **address\_ids** từ **user** document và sau đó là lấy các địa chỉ này từ **address** collection.



# 8. Tham chiếu Database trong MongoDB

**Referenced Relationship**, còn được gọi là **Manual References**, trong đó chúng ta thao tác để lưu giữ id của các Document được tham chiếu bên trong Document khác. Tuy nhiên, trong các trường hợp mà một Document chứa nhiều tham chiếu từ các Collection khác nhau, thì chúng ta sử dụng **DBRefs** trong MongODB

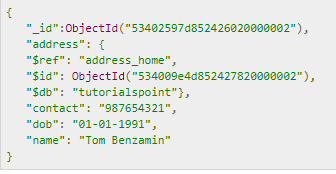
Có 3 trường trong DBRefs:

**$ref:** Trường này xác định Collection của Document được tham chiếu.

**$id:** Trường này xác định trường \_id của Document được tham chiếu.

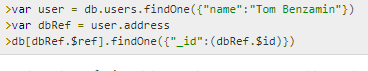
**$db:** Trường này là một trường tùy ý, chứa tên của Database mà Document được tham chiếu ở trong đó.

Giả sử một user document có trường **address** dạng DBRefs như sau

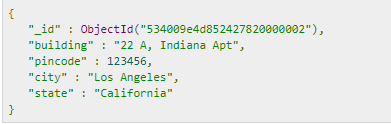


Trường address dạng DBRefs ở đây xác định rằng address document được tham chiếu ở trong address\_home collection dưới tutorialspoint database và có một id là: 534009e4d852427820000002.

Phần code sau sẽ nhìn vào Collection được xác định bởi tham số **$ref** (là address\_home trong trường hợp này) cho một Document với id được xác định bởi tham số **$id** trong DBRefs.



Code trên trả về address document sau đây, mà có mặt trong **address\_home** collection:



# 9. Chỉ mục (Index) trong MongoDB

Chỉ mục (Index) hỗ trợ việc phân giải các truy vấn hiệu quả hơn. Nếu không có chỉ mục, MongoDB phải quét qua mọi Document của một Collection để chọn các Document mà kết nối với lệnh truy vấn. Việc quét này có thể không hiệu quả và yêu cầu MongoDB xử lý một số lượng lớn dữ liệu.

Chỉ mục (Index) là các cấu trúc dữ liệu đặc biệt, lưu giữ một phần nhỏ của tập hợp dữ liệu, giúp việc "vọc" vào Collection một cách dễ dàng hơn. Chỉ mục lưu giữ giá trị của một trường cụ thể hoặc tập hợp các trường, được sắp xếp bởi giá trị của trường như đã được xác định trong chỉ mục.

# 10. Phương thức ensureIndex() trong MongoDB

Để tạo một chỉ mục, bạn cần sử dụng phương thức ensureIndex() của MongoDB.

Cú pháp

Cú pháp cơ bản của phương thức ensureIndex() là như sau:

>db.COLLECTION\_NAME.ensureIndex({KEY:1})

Ở đây, key là tên của trường mà bạn muốn tạo chỉ mục và 1 là cho thứ tự tăng dần. Để tạo chỉ mục theo thứ tự giảm dần, bạn cần sử dụng -1.

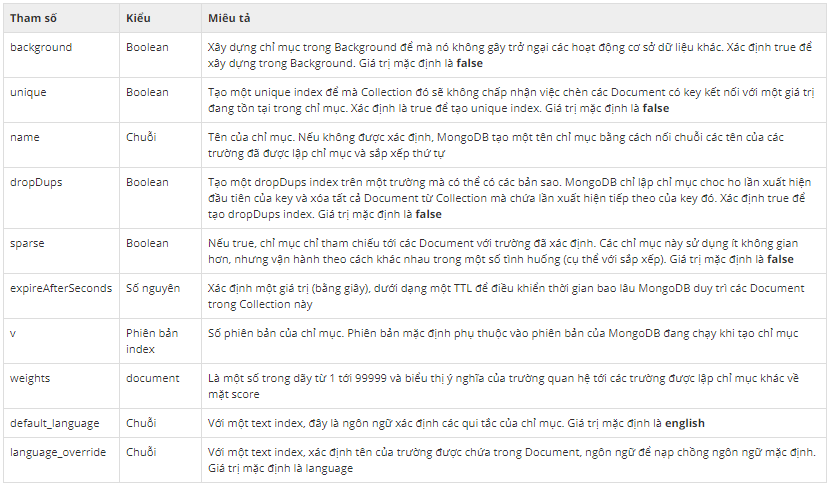
Ví dụ

>db.mycol.ensureIndex({"title":1})

Trong phương thức **ensureIndex()**, bạn có thể truyền nhiều trường, để tạo chỉ mục trên nhiều trường, bạn sử dụng:

>db.mycol.ensureIndex({"title":1,"description":-1})

Phương thức ensureIndex() cũng chấp nhận danh sách các tùy chọn tùy ý, được liệt kê dưới đây:



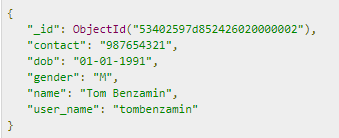
# 11. Covered Query trong MongoDB

Trong MongoDB Documentation chính thức, một Covered Query là một truy vấn mà trong đó:

* Tất cả các trường trong truy vấn là một phần của một chỉ mục và
* Tất cả các trường được trả về trong truy vấn là trong cùng chỉ mục.

Khi tất cả các trường có mặc trong truy vấn là một phần của chỉ mục, MongoDB kết nối các điều kiện truy vấn và trả về kết quả bởi sử dụng cùng chỉ mục đó mà không nhìn vào bên trong Document. Khi các chỉ mục có mặt trong RAM, việc lấy dữ liệu từ các chỉ mục là nhanh hơn khi so sánh với khi lấy dữ liệu bằng cách quét toàn bộ các Document.

Để nghiên cứu về Covered Query, bạn theo dõi Document sau trong **user** collection



Đầu tiên, chúng ta tạo một chỉ mục phức hợp cho users collection trên các trường gender và user\_name bởi sử dụng truy vấn sau:



Bây giờ, chỉ mục này sẽ bao phủ truy vấn sau:



Có thể nói rằng, với truy vấn trên, MongoDB sẽ không đi vào tìm kiếm các Document trong cơ sở dữ liệu. Thay vào đó, nó sẽ lấy dữ liệu cần thiết từ dữ liệu đã lập chỉ mục, điều này sẽ giúp tiến trình xảy ra nhanh hơn rất nhiều.

# 12. MongoDB hoạt động như thế nào:

## Ưu điểm

MongoDB hoạt động dưới một tiến trình ngầm service, luôn mở một cổng (Cổng mặc định là 27017) để lắng nghe các yêu cầu truy vấn, thao tác từ các ứng dụng gửi vào sau đó mới tiến hành xử lý.

Mỗi một bản ghi của MongoDB được tự động gắn thêm một field có tên “\_id” thuộc kiểu dữ liệu ObjectId mà nó quy định để xác định được tính duy nhất của bản ghi này so với bản ghi khác, cũng như phục vụ các thao tác tìm kiếm và truy vấn thông tin về sau.  
Trường dữ liệu “\_id” luôn được tự động đánh index (chỉ mục) để tốc độ truy vấn thông tin đạt hiệu suất cao nhất.

Mỗi khi có một truy vấn dữ liệu, bản ghi được cache (ghi đệm) lên bộ nhớ Ram, để phục vụ lượt truy vấn sau diễn ra nhanh hơn mà không cần phải đọc từ ổ cứng.

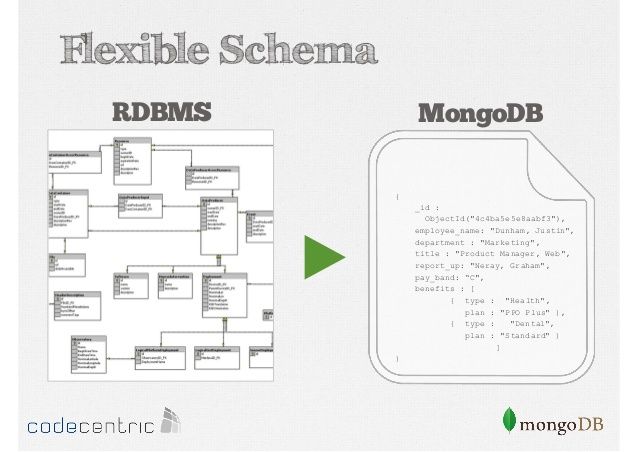
Khi có yêu cầu thêm/sửa/xóa bản ghi, để đảm bảo hiệu suất của ứng dụng mặc định MongoDB sẽ chưa cập nhật xuống ổ cứng ngay, mà sau 60 giây MongoDB mới thực hiện ghi toàn bộ dữ liệu thay đổi từ RAM xuống ổ cứng.

## Nhược điểm

Dữ liệu được caching, lấy RAM làm trọng tâm hoạt động vì vậy khi hoạt động yêu cầu một bộ nhớ RAM lớn

Như đã giới thiệu ở trên, mọi thay đổi về dữ liệu mặc định đều chưa được ghi xuống ổ cứng ngay lập tức vì vậy khả năng bị mất dữ liệu từ nguyên nhân mất điện đột xuất là rất cao

# 13. Lợi thế của MongoDB

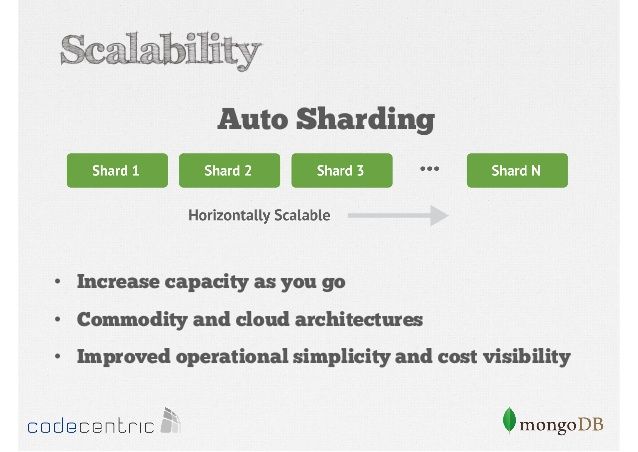


Ít schema hơn: Vì schema được sinh ra là để nhóm các đối tượng vào 1 cụm, dễ quản lý. Ví dụ như tạo 1 schema tên là Students chẳng hạn thì chỉ có những gì liên quan đến student thì mới được cho vào schema này. Trong khi đó trong mongodb thì chỉ 1 collection ta có thể chứa nhiều document khác nhau . Với mỗi document thì số trường, nội dung, kích thước lại có thể khác nhau.

Cấu trúc của một đối tượng rõ ràng.

Không có các Join phức tạp.

Khả năng mở rộng cực lớn: việc mở rộng dữ liệu mà không phải lo đến các vấn đề như khóa ngoại, khóa chính, kiểm tra ràng buộc, ... MongoDB cho phép thực hiện replication và sharding nên việc mở rộng cũng thuận lợi hơn.



Sử dụng bộ nhớ trong để lưu giữ cửa sổ làm việc cho phép truy cập dữ liệu nhanh hơn. Việc cập nhật được thực hiện nhanh gọn nhờ update tại chỗ (in-place).

# 14. Khi nào NÊN sử dụng MongoDB?

Sử dụng MongoDB trong trường 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.

# 15. Khi nào KHÔNG NÊN sử dụng MongoDB?

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).

# 16. Các công cụ quản trị MongoDB

RoboMongo

MongoExplorer

Mongodb Compass

# 17. Cài đặt MongoDB

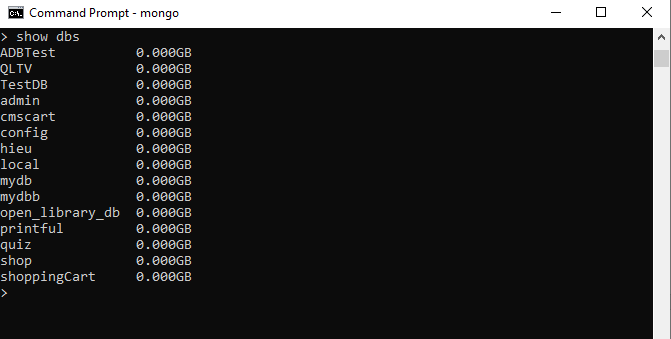
Trên Windows:

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

Chi tiết tại: <https://o7planning.org/vi/10265/huong-dan-cai-dat-va-cau-hinh-database-mongodb>

# 18. Một số ví dụ về mongo

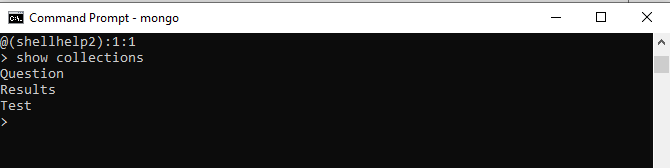
Show db



Tạo db ( lưu ý phải tạo ít nhất 1 collection thì mới hiển thị db)



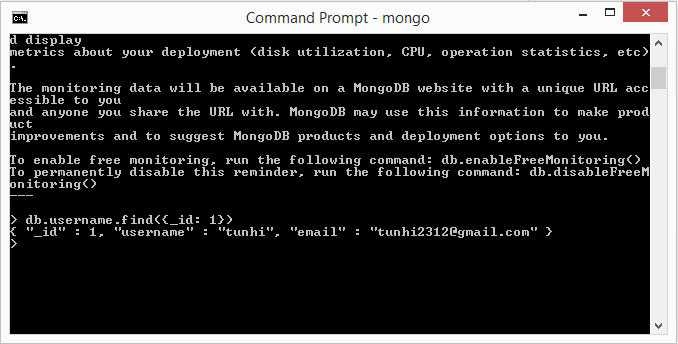
Show collection



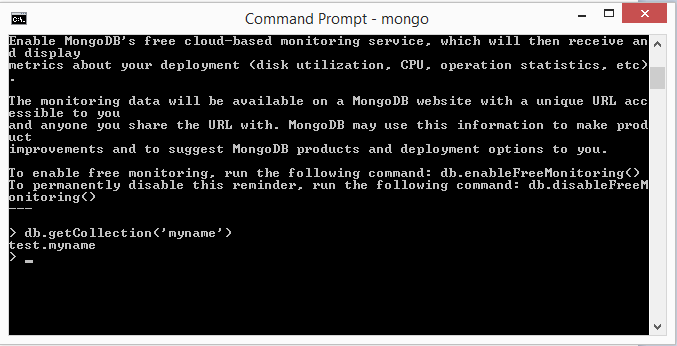
Find



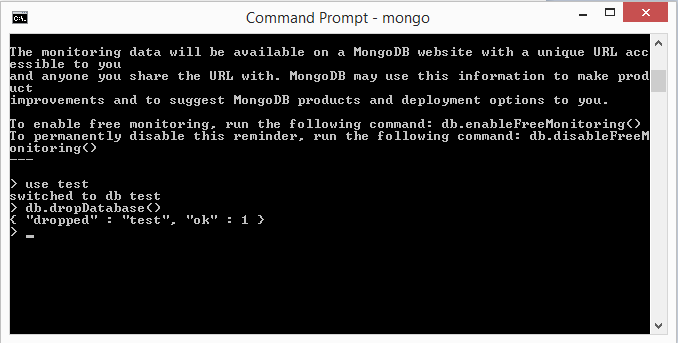
Find có điều kiện



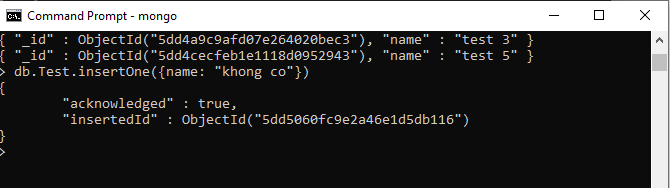
Get collection



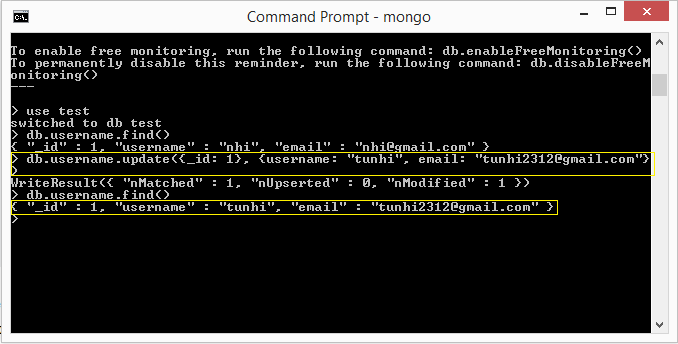
Drop database

**

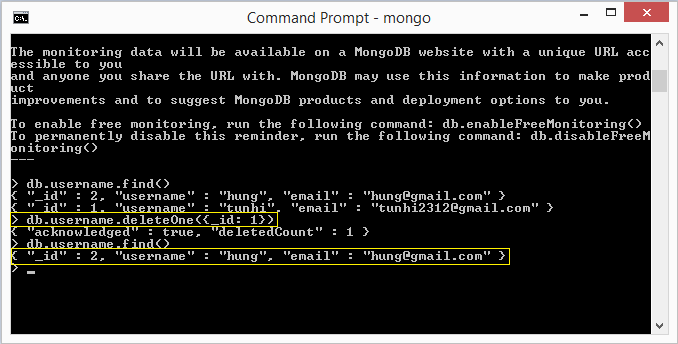
Insert



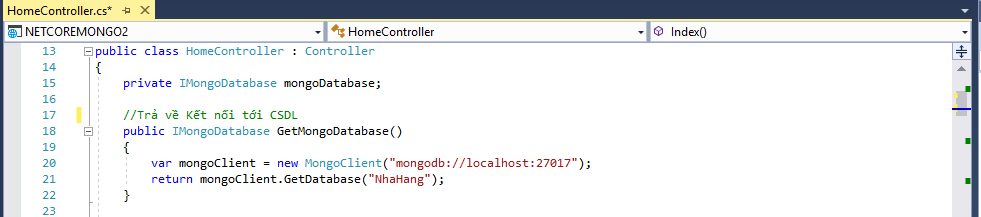
Update



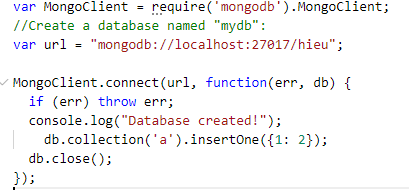
Xóa



Tạo kết nối c# với mongodb



Tạo kết nối với nodejs



19. Tổng kết và đánh giá:

* Tổng kết

MongoDB không sử dụng mô hình quan hệ ràng buộc(non-relational)

Mô hình Lưu trữ dạng {“key : value”}

MongoDB là 1 CSDL phân tán lưu chữ dữ liệu trên nhiều server khác nhau, đa dạng hóa dữ liệu, là dạng dữ liệu lớn( BigData).

Độ chịu tải cao ,chịu lỗi cao

Có khả năng nâng cấp theo chiều ngang, dễ dàng bảo trì, thay đổi,sửa chữa , phát triển

* Đánh giá

Chính vì những đặc điểm đó ta nên dùng nó cho các hệ thống, ứng dụng đòi hỏi lượng người dùng và dữ liệu lưu trữ đa dạng, lớn đến cực lớn.Ví dụ như mạng xã hội(facebook, Instagram, twitter…), các trang lưu trữ dữ liệu lớn như google, github, …..

TÀI LIỆU THAM KHẢO

1. <https://docs.mongodb.com/manual/tutorial/>
2. <https://en.wikipedia.org/wiki/MongoDB>
3. <https://vietjack.com/mongodb/>