
Bài 1: MongoDB là gì?

1. MongoDB là gì?

MongoDB là một hệ quản trị cơ sở dữ liệu mã nguồn mở thuộc họ NoSQL, được thiết kế theo kiểu hướng đối tượng, các bảng trong MongoDB được cấu trúc rất linh hoạt, cho phép các dữ liệu lưu trữ trên bảng không cần tuân theo một cấu trúc nhất định nào cả (điều này rất thích hợp để làm big data).

MongoDB lưu trữ dữ liệu theo hướng tài liệu (document), các dữ liệu được lưu trữ trong document kiểu JSON nên truy vấn sẽ rất nhanh.

2. Ưu điểm của MongoDB

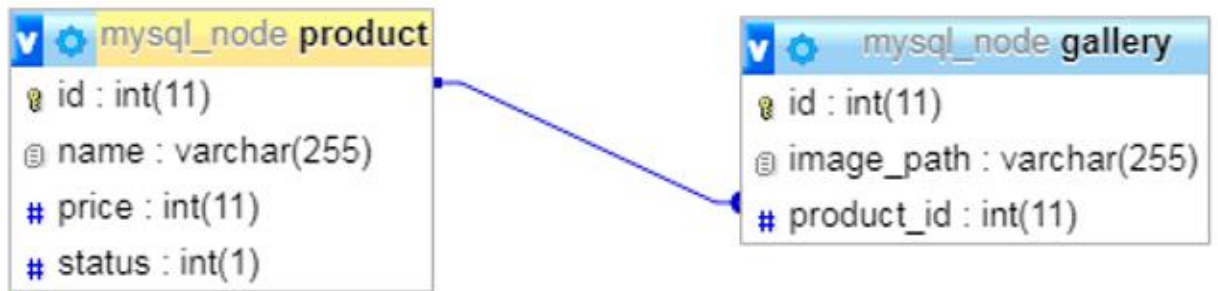
- Schema linh hoạt: Do MongoDB sử dụng lưu trữ dữ liệu dưới dạng Document JSON nên mỗi một collection sẽ các kích cỡ và các document khác nhau.
- Cấu trúc đối tượng rõ ràng: Tuy rằng cấu trúc của dữ liệu là linh hoạt nhưng đối tượng của nó được xác định rất rõ ràng.
- Sử dụng bộ nhớ nội tại, nên truy vấn sẽ rất nhanh.
- MongoDB rất dễ mở rộng.
- Không có các join: Điều này cũng góp phần tạo nên tốc độ truy vấn cực nhanh trên MongoDB.
- MongoDB phù hợp cho các ứng dụng realtime.

3. Nhược điểm của MongoDB

- Điều đầu tiên phải kể đến ở đây là MongoDB không có các tính chất ràng buộc như trong RDBMS nên khi thao tác với MongoDB thì phải hết sức cẩn thận.
- MongoDB sử dụng sẽ hao tốn tài nguyên của hệ thống nhiều hơn RDBMS. Nhưng đến thời điểm hiện tại thì vấn đề này không còn là điều lo ngại nữa do sự phát triển của phần cứng.

4. MongoDB với RDBMS

Đầu tiên, để cho dễ hiểu chúng ta sẽ xem xét dữ liệu trên MySQL như sau:



Lúc này nếu như trên mySQL thì chúng ta phải thực hiện tạo 2 bảng (table) để lưu trữ dữ liệu. Nhưng với MongoDB thì chúng ta sẽ chỉ cần 1 collection để lưu trữ như sau:

```
{
  "_id": "_id",
  "name": "name",
  "price": "price",
  "status": "status",
  "gallery": [
    {"image_path": "image_path"},
    {"image_path": "image_path2"},
  ]
}
```

Sau đây là một vài so sánh sự tương đồng giữa MongoDB và RDBMS (Cơ sở dữ liệu quan hệ).

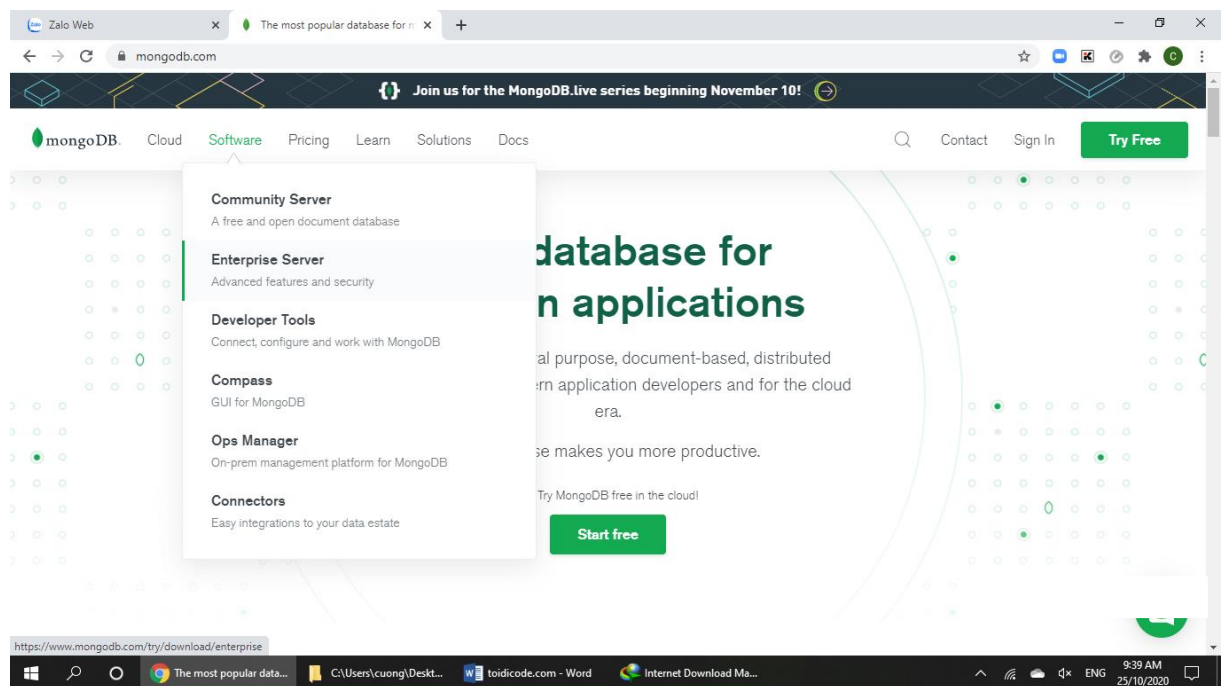
RDBMS	MongoDB
Table	Collection
Row	Document
Column	Field
Joins	Embedded documents, linking
Primary key	Key

Bài 2: Cài Đặt MongoDB trên Windows

1. Cài đặt MongoDB

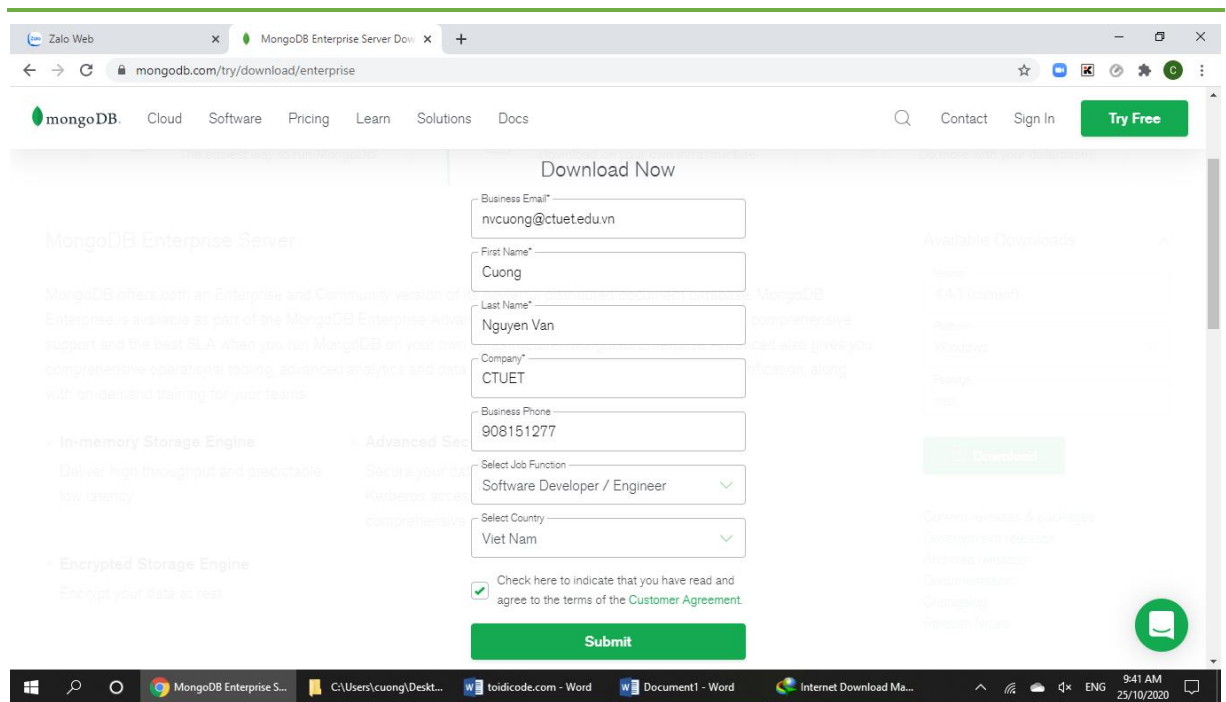
Đầu tiên các bạn truy cập vào địa chỉ website <https://www.mongodb.com/download-center?jmp=nav>

Và làm theo các hướng dẫn sau (lưu ý có thể lúc bạn đọc tài liệu này giao diện trang chủ đã thay đổi).



Chọn Enterprise Server -> chọn Windows x64 -> Download.

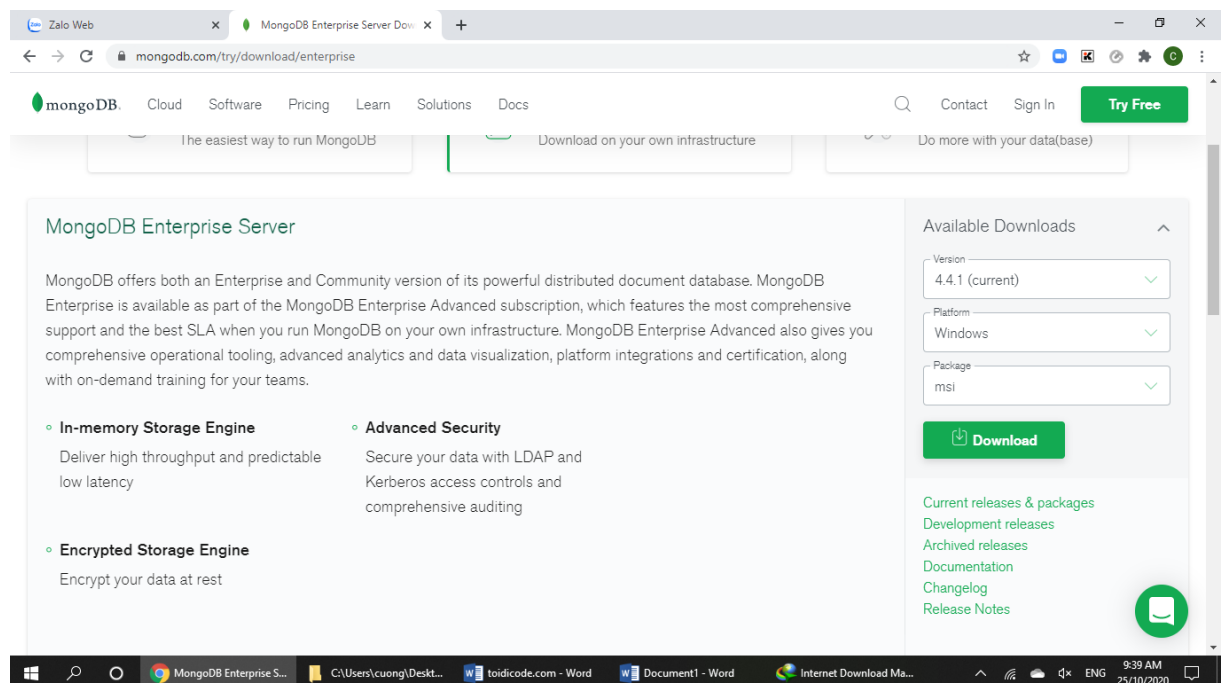
Tiếp đó bạn cần điền đầy đủ thông tin vào trong form (nhớ checked vào ô Check here...) và chọn Submit.



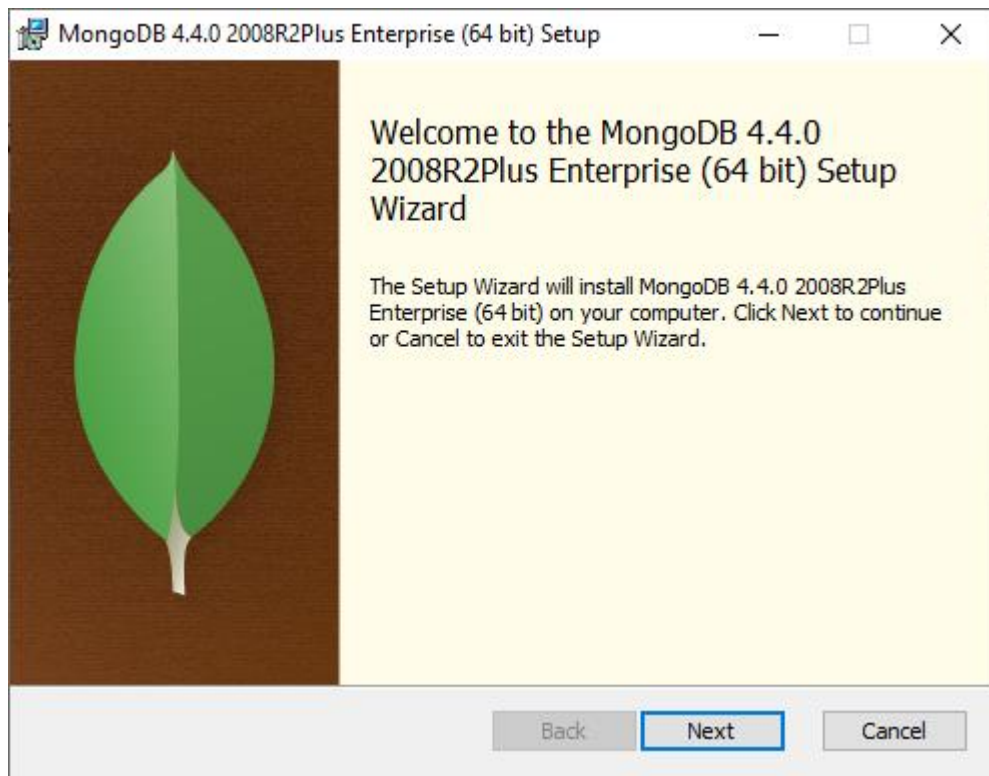
Lúc này MongoDB cung cấp cho chúng ta hai lựa chọn

- **archive** là download tập tin zip không cần cài đặt.
- **msi** là tập tin cài đặt.

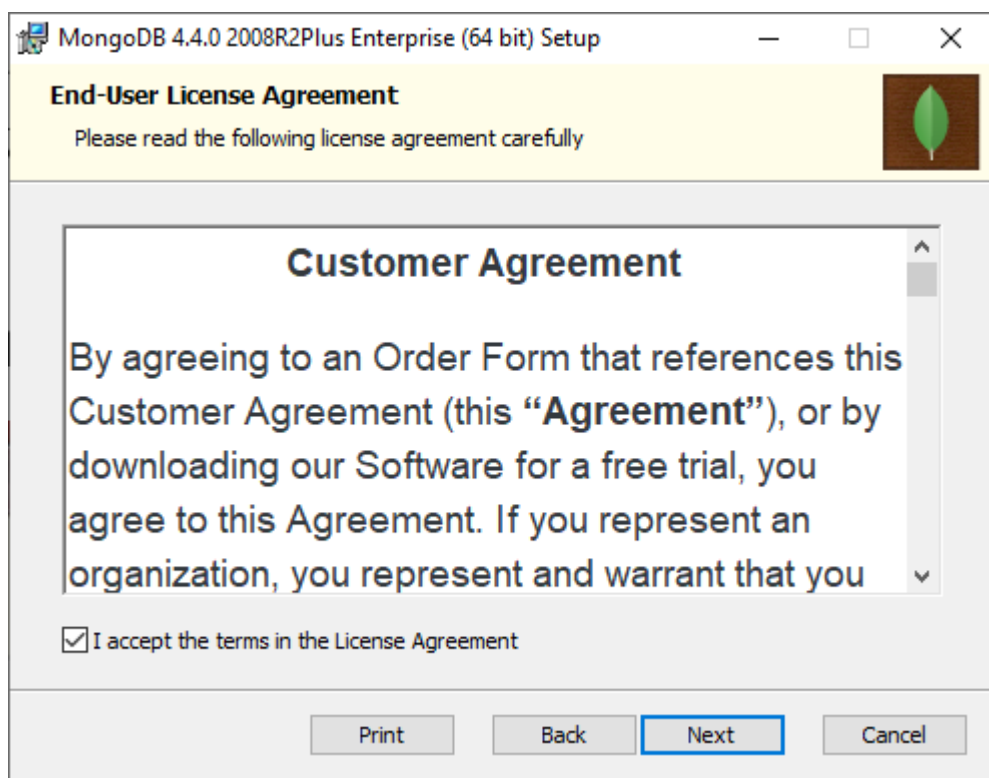
Ở đây chúng ta chọn **msi**.



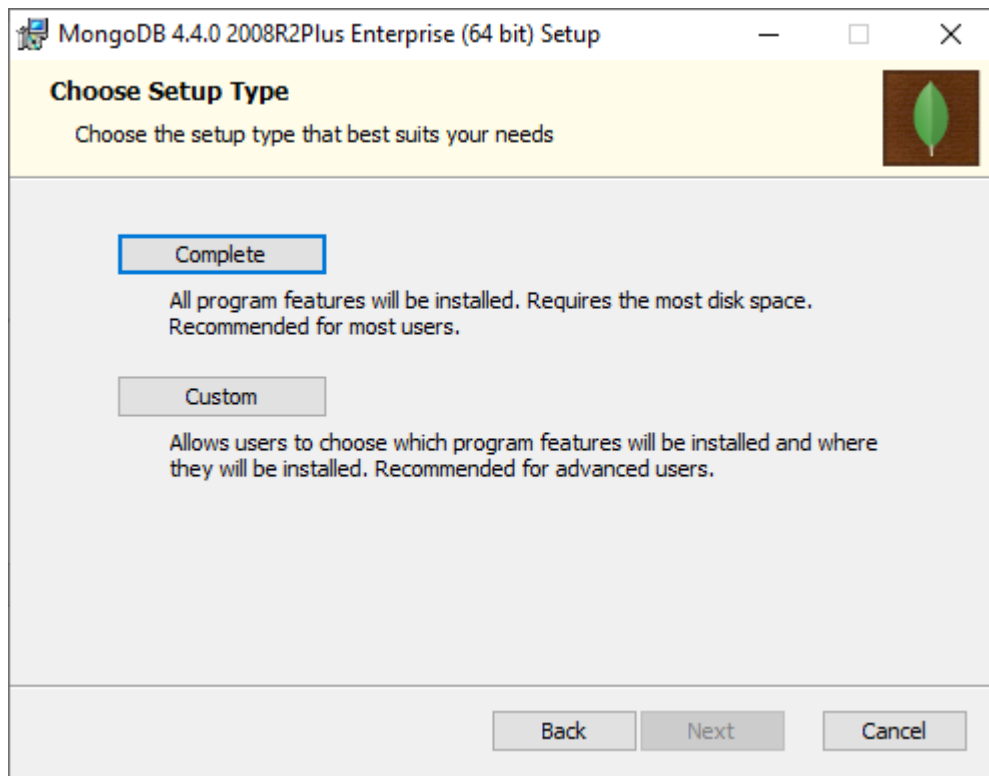
Sau khi đã tải về thành công, các bạn thực thi tập tin msi để cài đặt.



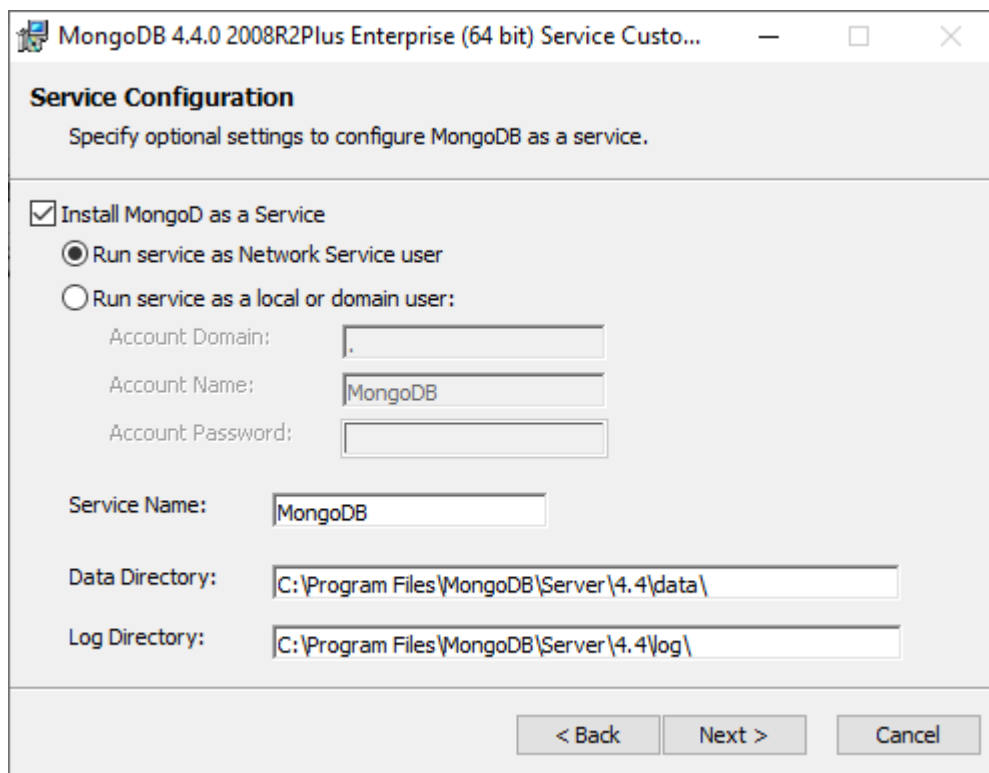
Các bạn chọn Next để tiếp tục



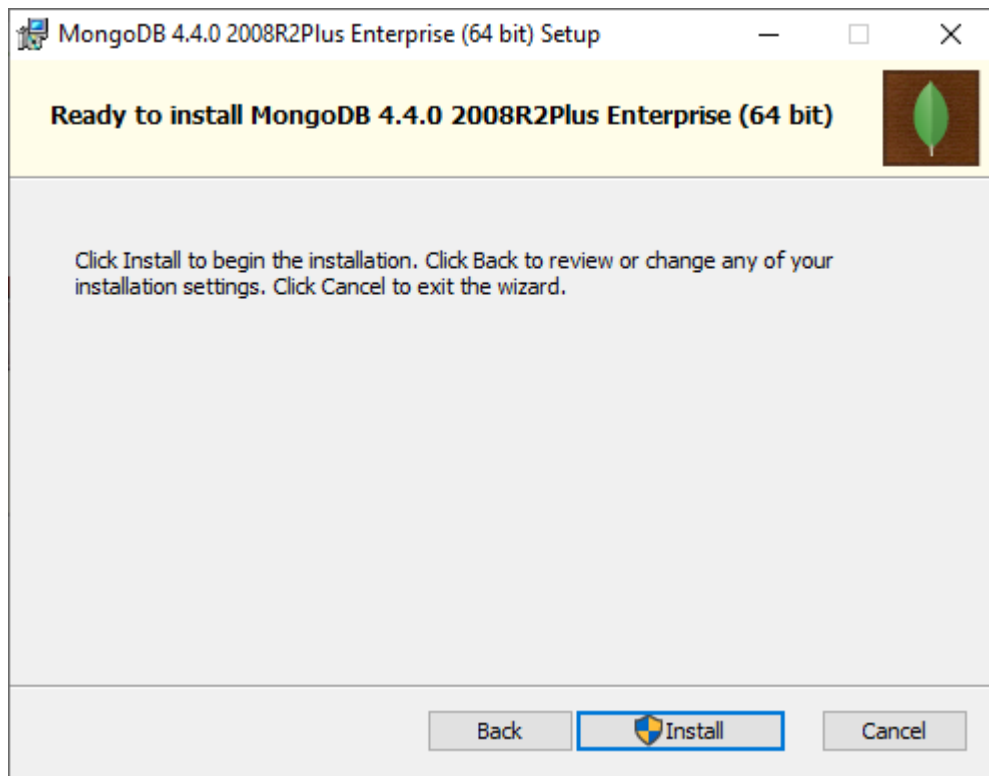
check vào check box I accept... và chọn Next.



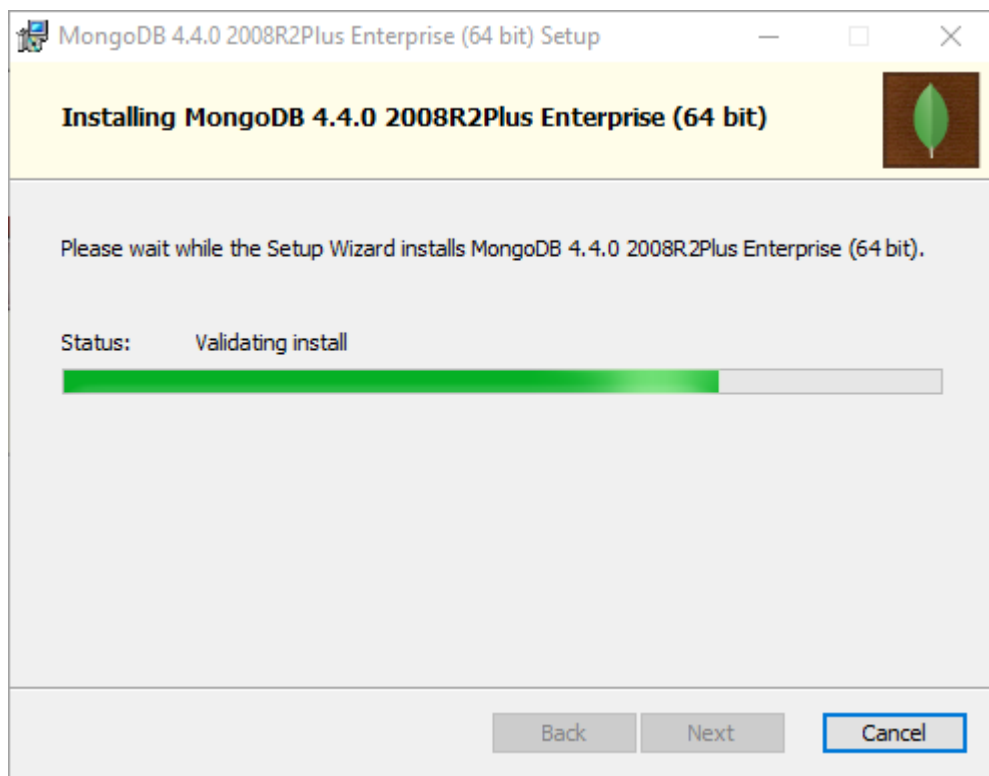
Lúc này có 2 lựa chọn là Complete và Custom. Nếu chọn Complete thì sẽ tiến hành cài đặt theo mặc định, còn Custom thì sẽ cho phép bạn chỉnh sửa thư mục cài đặt MongoDB.

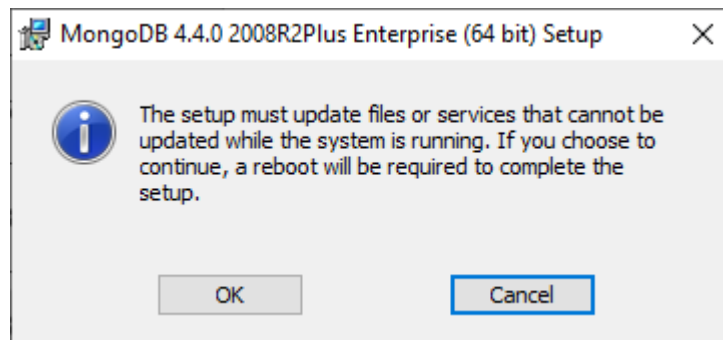


Ở đây, nếu muốn thay đổi đường dẫn cài đặt MongoDB thì bạn chọn Browser... và chọn thư mục chứa rồi chọn tiếp Next.

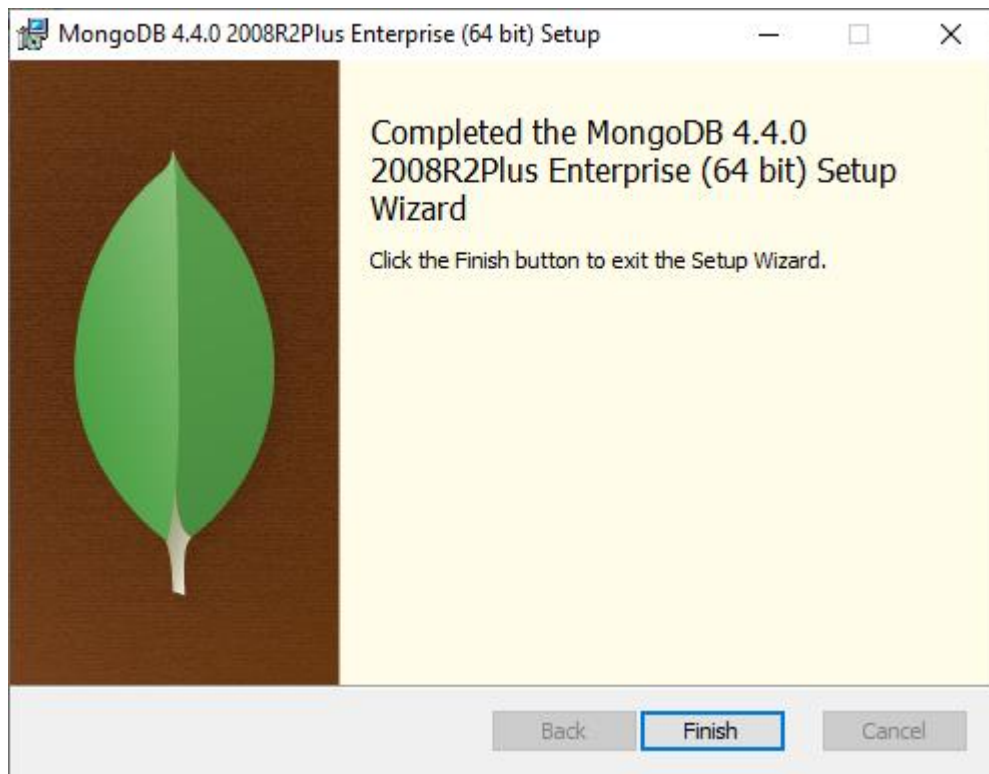


Chọn Install để cài đặt





Đợi tiến trình cài đặt

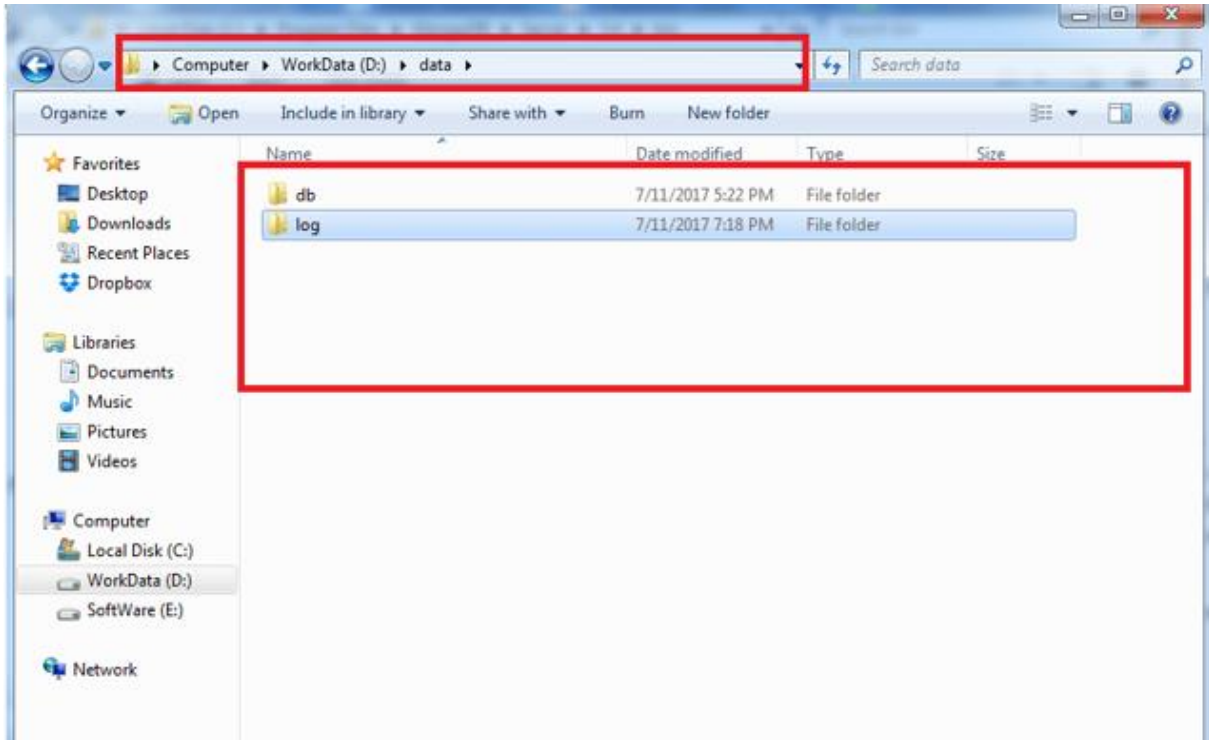


Chọn **Finish** để hoàn tất quá trình cài đặt.

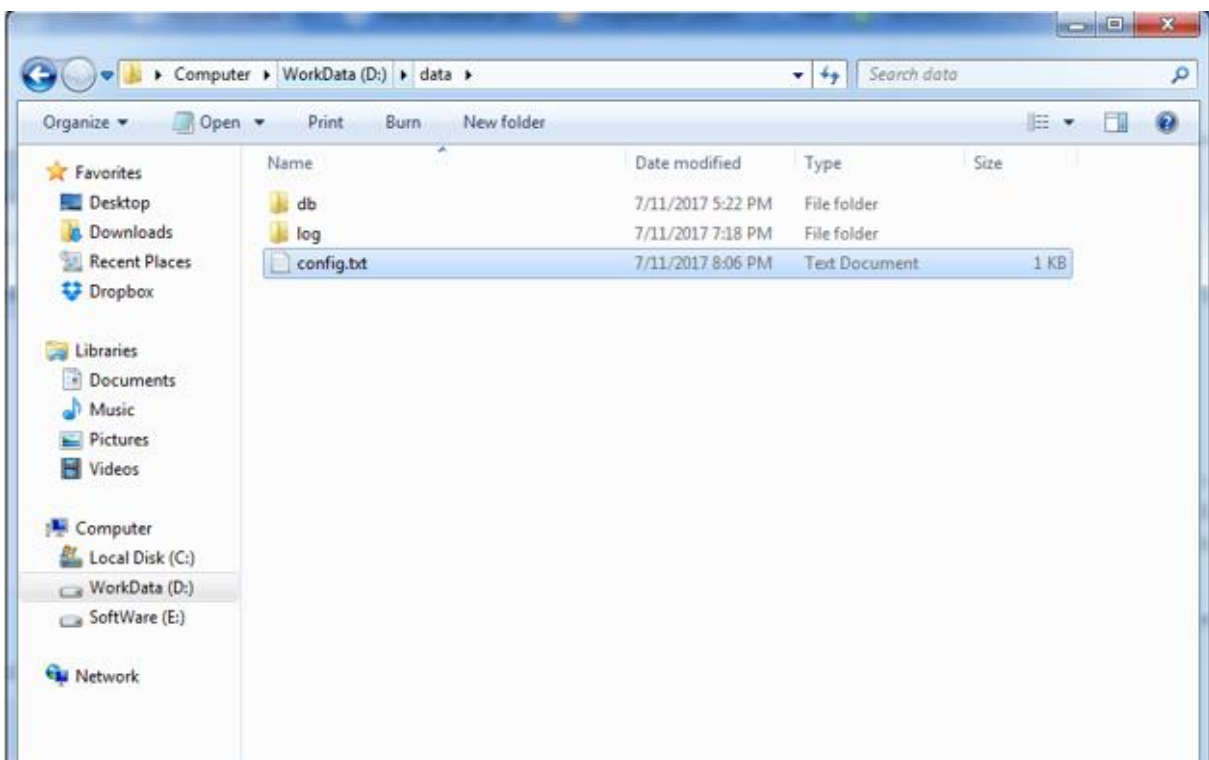
Bài 3: Cấu hình và chạy MongoDB trên Windows

1. Cấu hình MongoDB

- Sau khi đã cài đặt thành công MongoDB thì chúng ta tạo tiếp một thư mục (ở đâu tùy ý) để lưu trữ data và log.



Và đồng thời tạo thêm một tập tin text có tên **config.txt**



Trong tập tin này chúng ta sẽ cấu hình thư mục chứa db và log như sau:

```
dbpath=D:\data\db
```

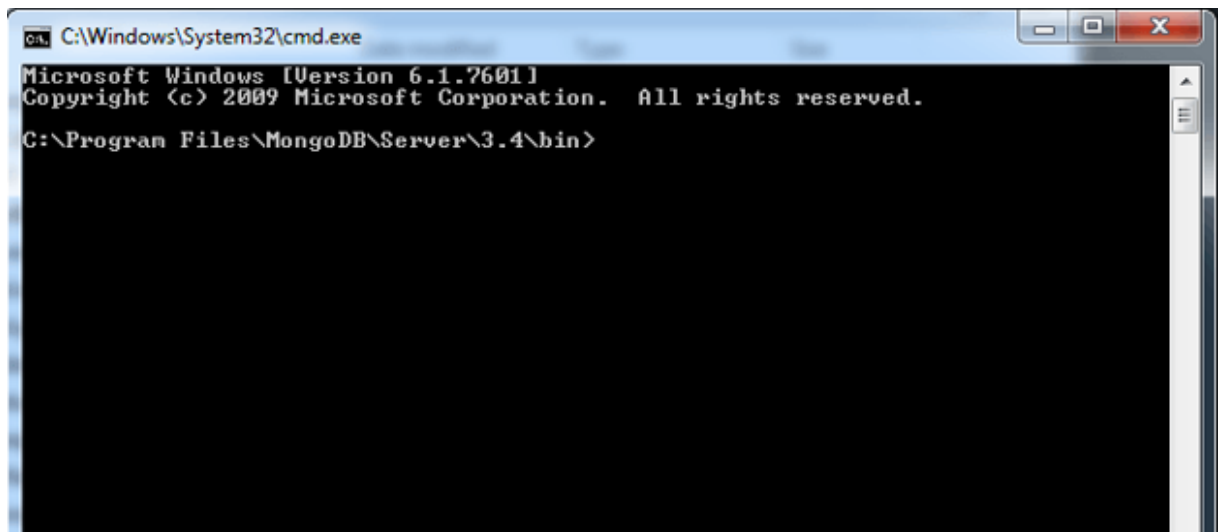
```
logpath=D:\data\log\mongolog.log
```

Trong đó:

- dbpath là tham số cấu hình thư mục chứa db.
- logpath là tham số cấu hình tập tin chứa log.

2. Khởi động MongoDB

Tiếp đó để chạy MongoDB, chúng ta mở cmd (Windows + R) và di chuyển (cd) đến thư mục cài đặt MongoDB vừa cài đặt và vào thư mục bin.



Tiếp đó chúng ta chạy lệnh sau để cấu hình MongoDB:

```
mongod --config "D:\data\config.txt"
```

Chú ý: đường dẫn trên là đường dẫn đến tập tin config.txt mà chúng ta vừa tạo ở trên.

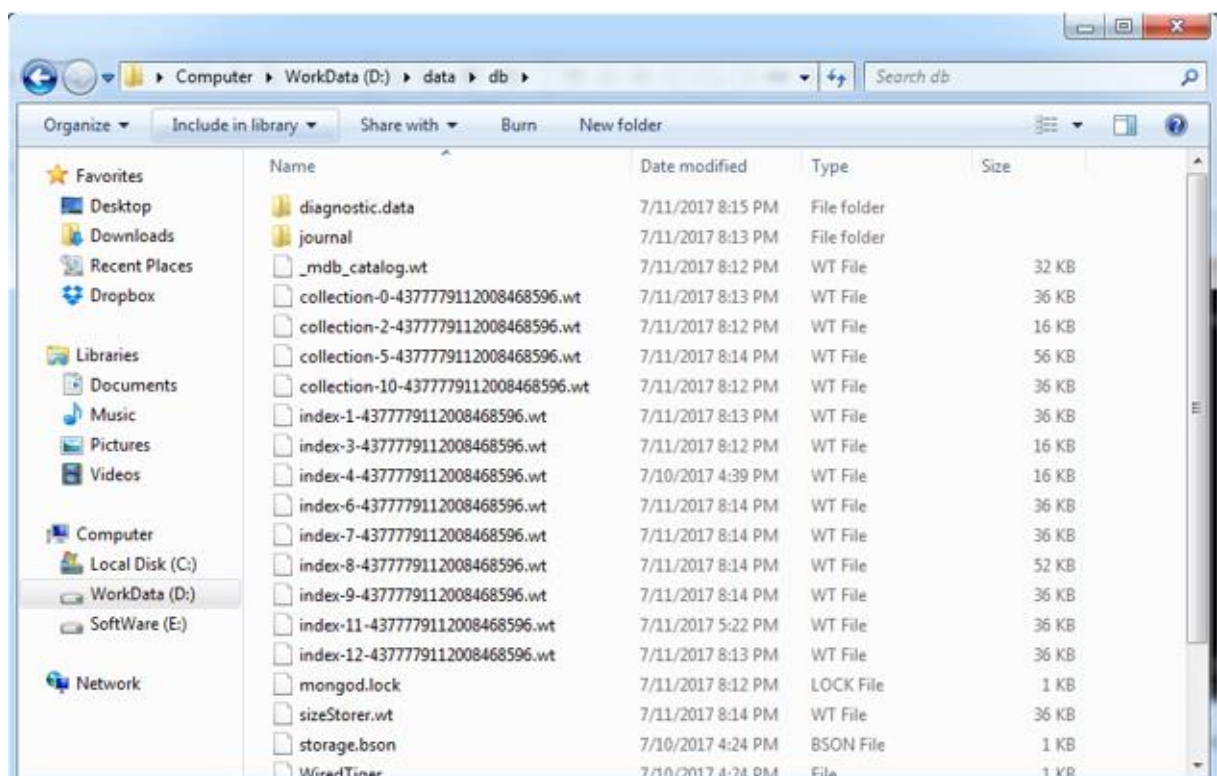
```
C:\Windows\System32\cmd.exe - mongod --config "D:\data\config.txt"

Microsoft Windows [Version 6.1.7601]
Copyright (c) 2009 Microsoft Corporation. All rights reserved.

C:\Program Files\MongoDB\Server\3.4\bin>mongod --config "D:\data\config.txt"
```

Nếu không có lỗi gì như hình và dấu _ cứ nhấp nháy thì chúng ta đã khởi chạy thành công MongoDB.

- Nếu như cần chắc chắn hơn, bạn truy cập vào đường dẫn đã cấu hình để lưu trữ database trong tập tin **config.txt** sẽ thấy có một loạt các tập tin lạ được sinh ra (đó là các tập tin của MongoDB).



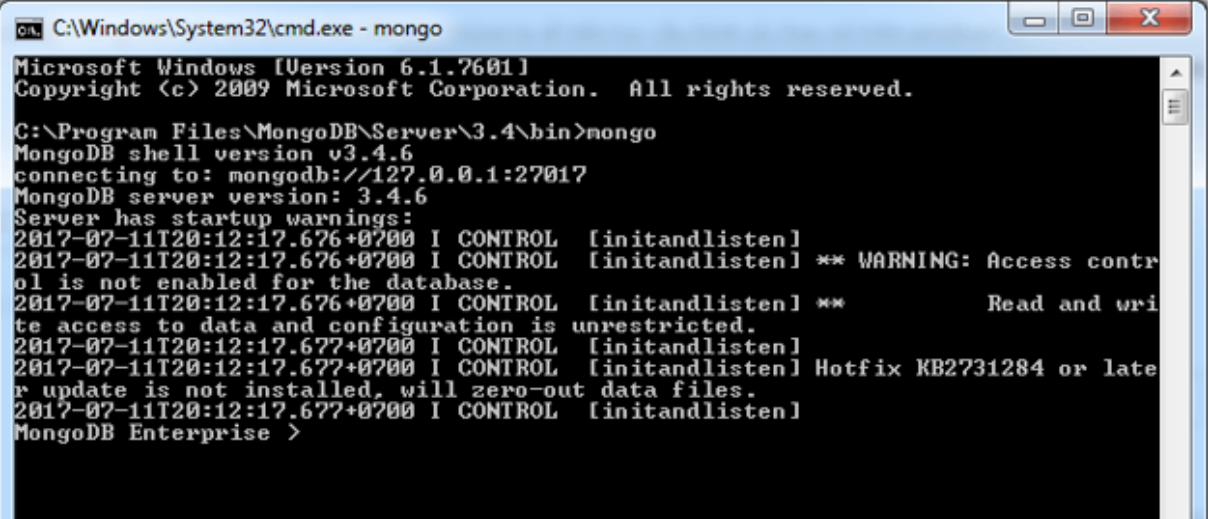
Chú ý: Khi nào muốn dừng MongoDB thì chỉ cần đóng cửa sổ cmd đi, hoặc nhấn tổ hợp phím CTRL+C.

3. Xem thông tin MongoDB

- Để xem thông tin của MongoDB đang chạy thì chúng ta chạy tiếp 1 cửa sổ cmd nữa và cd đến thư mục bin của MongoDB như ở trên (**Chú ý: cmd chạy MongoDB vẫn phải được mở, tức là MongoDB phải đang ở trạng thái run**) và gõ lệnh:

```
mongo
```

Lúc này cửa sổ cmd sẽ hiển thị ra các thông số của MongoDB đang chạy.



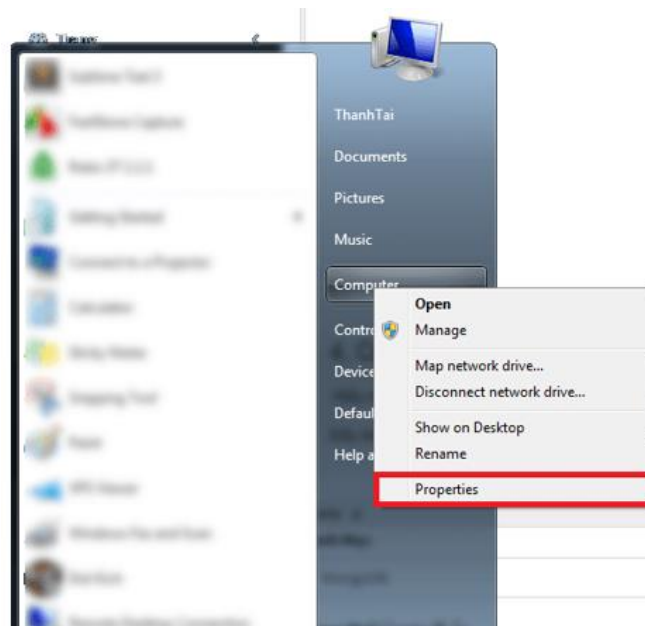
```
C:\Windows\System32\cmd.exe - mongo
Microsoft Windows [Version 6.1.7601]
Copyright (c) 2009 Microsoft Corporation. All rights reserved.

C:\Program Files\MongoDB\Server\3.4\bin>mongo
MongoDB shell version v3.4.6
connecting to: mongodb://127.0.0.1:27017
MongoDB server version: 3.4.6
Server has startup warnings:
2017-07-11T20:12:17.676+0700 I CONTROL [initandlisten]
2017-07-11T20:12:17.676+0700 I CONTROL [initandlisten] ** WARNING: Access control
ol is not enabled for the database.
2017-07-11T20:12:17.676+0700 I CONTROL [initandlisten] **          Read and wri
te access to data and configuration is unrestricted.
2017-07-11T20:12:17.677+0700 I CONTROL [initandlisten]
2017-07-11T20:12:17.677+0700 I CONTROL [initandlisten] Hotfix KB2731284 or late
r update is not installed, will zero-out data files.
2017-07-11T20:12:17.677+0700 I CONTROL [initandlisten]
MongoDB Enterprise >
```

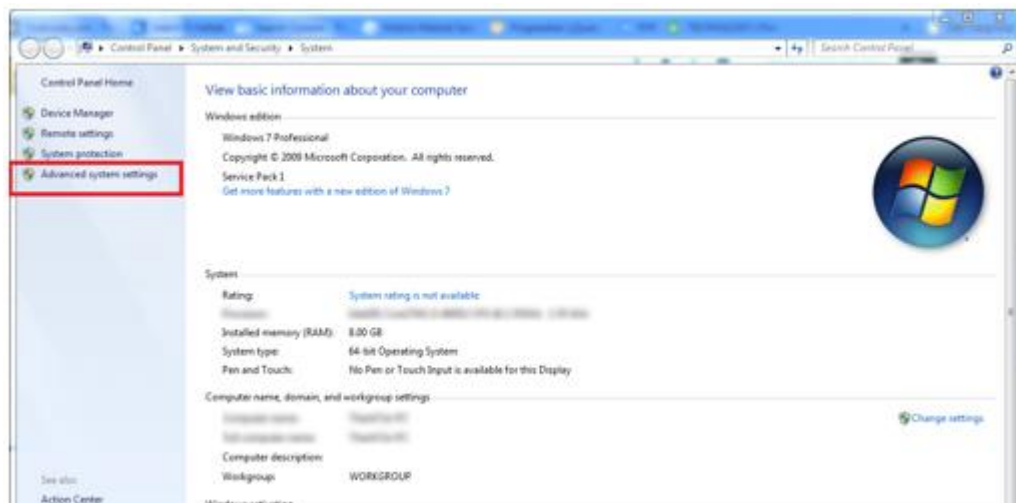
4. Cấu hình biến môi trường để thực thi MongoDB

- Nếu như việc sử dụng MongoDB như trên khá là bất tiện, thì chúng ta có thể cấu hình biến môi trường để có thể sử dụng lệnh MongoDB ở bất kỳ thư mục nào trên Windows mà không cần phải vào thư mục bin nữa.

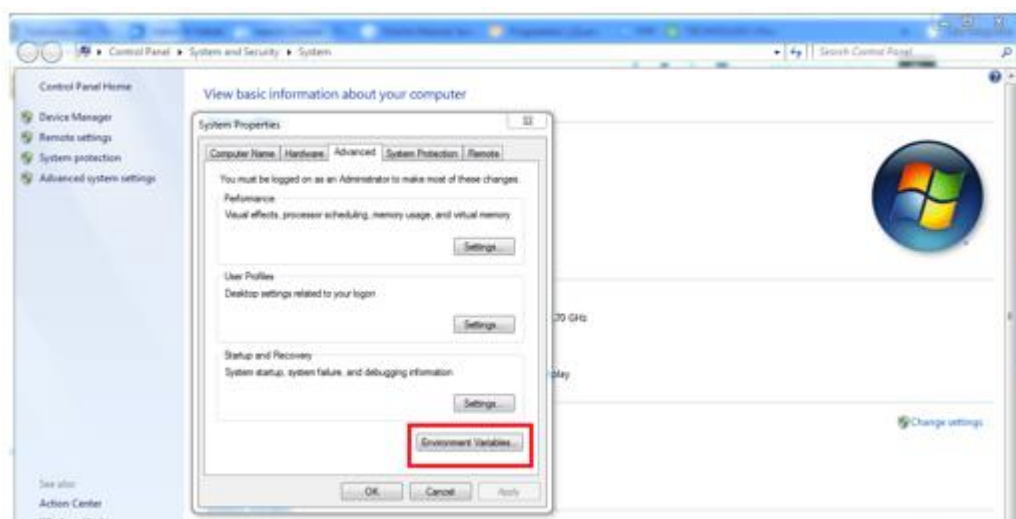
Đầu tiên chúng ta chọn vào biểu tượng Windows (hoặc nhấn phím Windows), chuột phải vào Computer và chọn **Properties**.



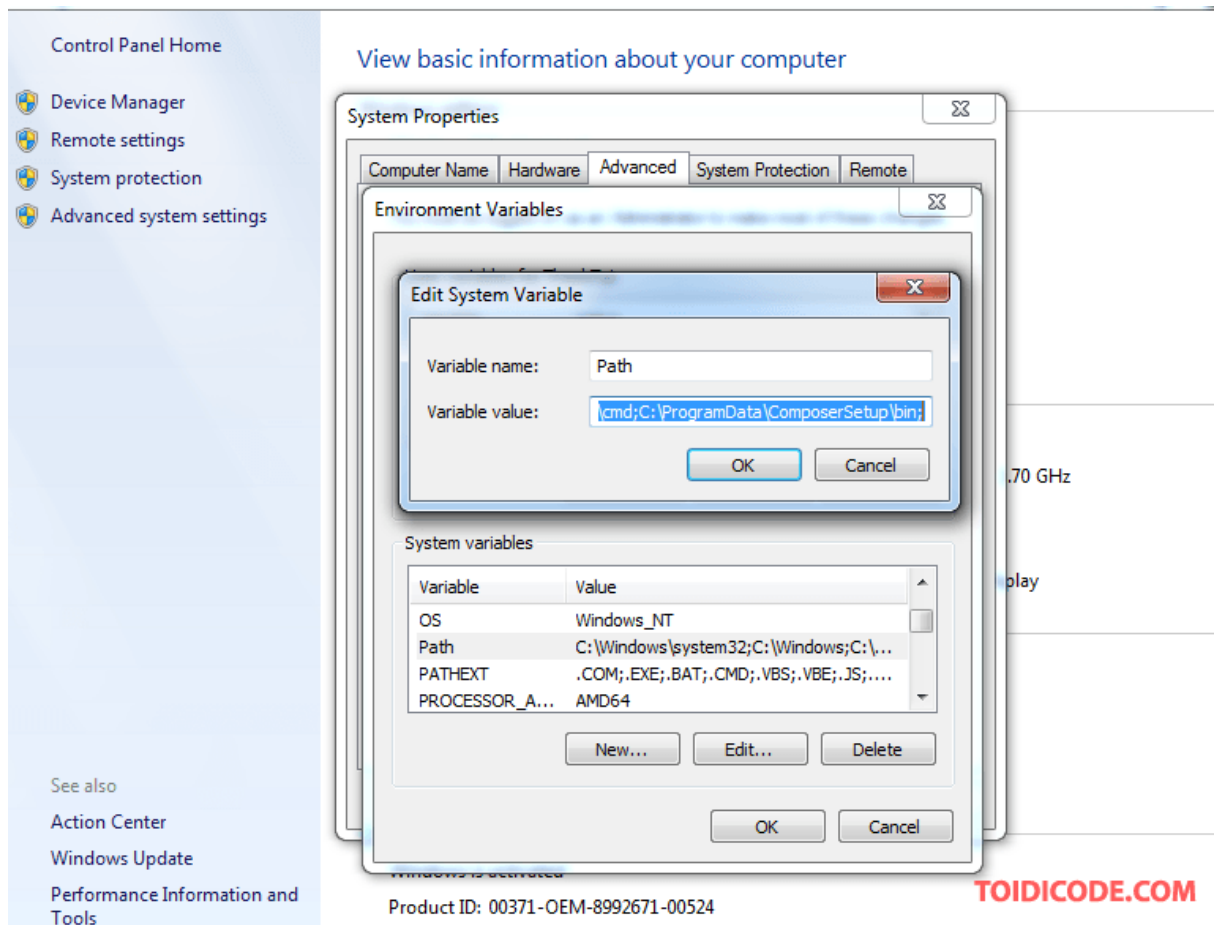
- Ngay sau đó một cửa sổ sẽ hiện ra, và bạn chọn tiếp **Advanced system settings**



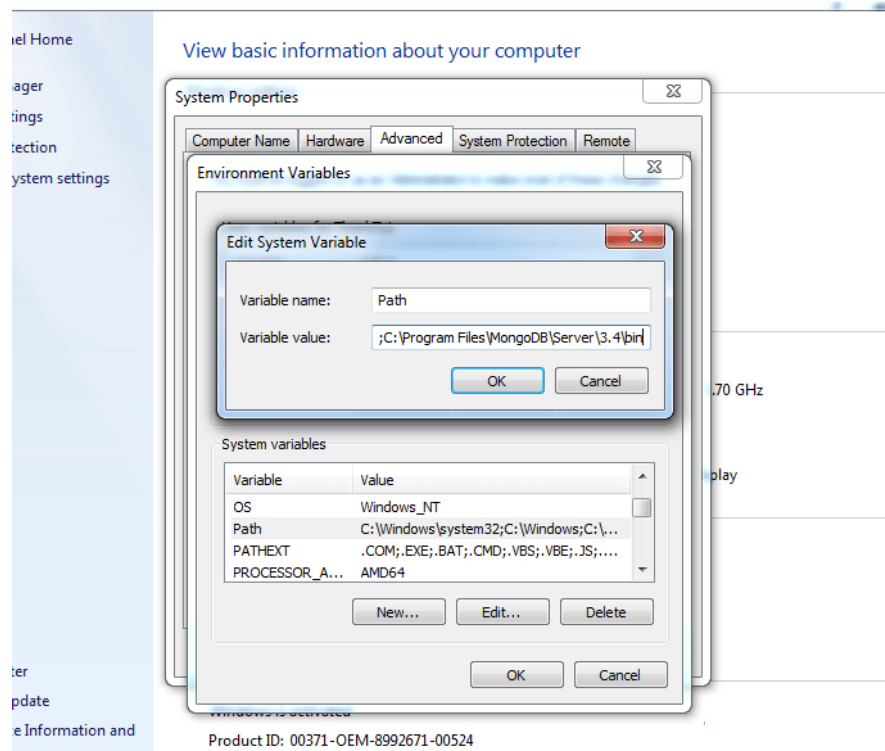
- Một cửa sổ mới hiện lên, chúng ta lại tiếp tục chọn **Environment Variables**



- Tiếp tục có một cửa sổ mới hiện lên, ở cửa sổ **Environment Variables**, tìm đến Variable có tên là **PATH** rồi click vào nó.



- Ở ô Variable value chúng ta điền thêm đường dẫn tới thư mục bin của mình vào (Chú ý phải có dấu **;** ngăn giữa path trước đó và path sẽ thêm vào). Ví dụ như **C:\Program Files\MongoDB\Server\3.4\bin**



- Sau đó chọn OK để lưu lại và hoàn tất quá trình cấu hình.

Sau khi đã cấu hình thành công biến môi trường rồi, thì chúng ta chỉ cần bật cmd lên ở bất kỳ đâu trên Windows và gõ các lệnh của MongoDB mà không cần cd đến thư mục bin nữa.

Ví dụ: Mình bật cmd ở thư mục ThanhTai vẫn có thể sử dụng được các lệnh MongoDB.

```

C:\Windows\system32\cmd.exe - mongo
Microsoft Windows [Version 6.1.7601]
Copyright (c) 2009 Microsoft Corporation. All rights reserved.

C:\Users\ThanhTai>mongo
MongoDB shell version v3.4.6
connecting to: mongod://127.0.0.1:27017
MongoDB server version: 3.4.6
Server has startup warnings:
2017-07-11T20:12:17.676+0700 I CONTROL [initandlisten]
2017-07-11T20:12:17.676+0700 I CONTROL [initandlisten] ** WARNING: Access control
is not enabled for the database.
2017-07-11T20:12:17.676+0700 I CONTROL [initandlisten] **      Read and write
access to data and configuration is unrestricted.
2017-07-11T20:12:17.677+0700 I CONTROL [initandlisten]
2017-07-11T20:12:17.677+0700 I CONTROL [initandlisten] Hotfix KB2731284 or later
update is not installed, will zero-out data files.
2017-07-11T20:12:17.677+0700 I CONTROL [initandlisten]
MongoDB Enterprise >

```

Bài 4: Tạo database trong MongoDB

1. Tạo database trong MongoDB

- Để tạo một database trong MongoDB chúng ta sử dụng cú pháp sau:

```
use DatabaseName
```

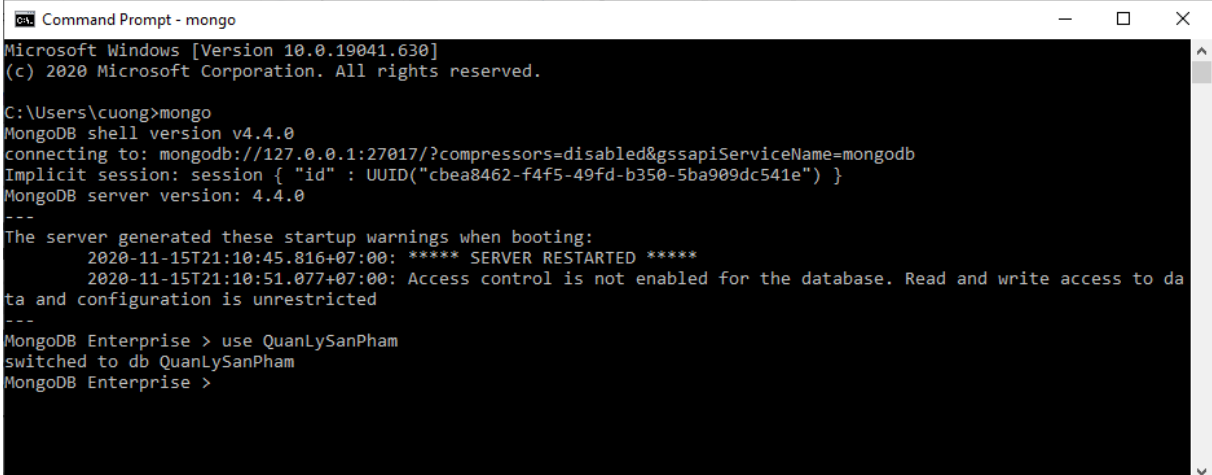
Trong đó: **DatabaseName** là tên của database muốn tạo.

Chú ý: Nếu như database muốn tạo đã tồn tại rồi thì nó sẽ không được tạo nữa mà sử dụng luôn database đó.

Ví dụ: Tạo database có tên **QuanLySanPham**

```
use QuanLySanPham
```

- Nếu như thành công thì cmd sẽ trả về *"switched to db QuanLySanPham"*.



```
Command Prompt - mongo
Microsoft Windows [Version 10.0.19041.630]
(c) 2020 Microsoft Corporation. All rights reserved.

C:\Users\cuong>mongo
MongoDB shell version v4.4.0
connecting to: mongodb://127.0.0.1:27017/?compressors=disabled&gssapiServiceName=mongodb
Implicit session: session { "id" : UUID("cbea8462-f4f5-49fd-b350-5ba909dc541e") }
MongoDB server version: 4.4.0
---
The server generated these startup warnings when booting:
  2020-11-15T21:10:45.816+07:00: ***** SERVER RESTARTED *****
  2020-11-15T21:10:51.077+07:00: Access control is not enabled for the database. Read and write access to da
ta and configuration is unrestricted
---
MongoDB Enterprise > use QuanLySanPham
switched to db QuanLySanPham
MongoDB Enterprise >
```

2. Xem database đang sử dụng

-Để xem database đang sử dụng (current database) thì chúng ta sử dụng lệnh:

```
db
```

Ví dụ: Xem database đang sử dụng


```
Command Prompt - mongo
Microsoft Windows [Version 10.0.19041.630]
(c) 2020 Microsoft Corporation. All rights reserved.

C:\Users\cuong>mongo
MongoDB shell version v4.4.0
connecting to: mongodb://127.0.0.1:27017/?compressors=disabled&gssapiServiceName=mongodb
Implicit session: session { "id" : UUID("cbea8462-f4f5-49fd-b350-5ba909dc541e") }
MongoDB server version: 4.4.0
---
The server generated these startup warnings when booting:
  2020-11-15T21:10:45.816+07:00: ***** SERVER RESTARTED *****
  2020-11-15T21:10:51.077+07:00: Access control is not enabled for the database. Read and write access to da
ta and configuration is unrestricted
---
MongoDB Enterprise > use QuanLySanPham
switched to db QuanLySanPham
MongoDB Enterprise > db
QuanLySanPham
MongoDB Enterprise >
```

3. Xem tất cả các database trong hệ thống

- Để xem tất cả các database đã được tạo trên MongoDB thì chúng ta sử dụng lệnh:

```
show dbs
```

CHÚ Ý: Lệnh này sẽ chỉ hiện ra các database đã có ít nhất một collection (hiểu như table ở trong [SQL Server](#)), còn nếu rỗng thì nó sẽ không hiện.

Ví dụ: Xem tất cả các database đã được tạo trên mongoDB

```
Select Command Prompt - mongo
MongoDB Enterprise > show dbs
QLNV          0.000GB
QuanLySanPham 0.000GB
admin         0.000GB
config        0.000GB
local         0.000GB
peopledb      0.000GB
MongoDB Enterprise >
```

Bài 5: Xóa Database trong MongoDB

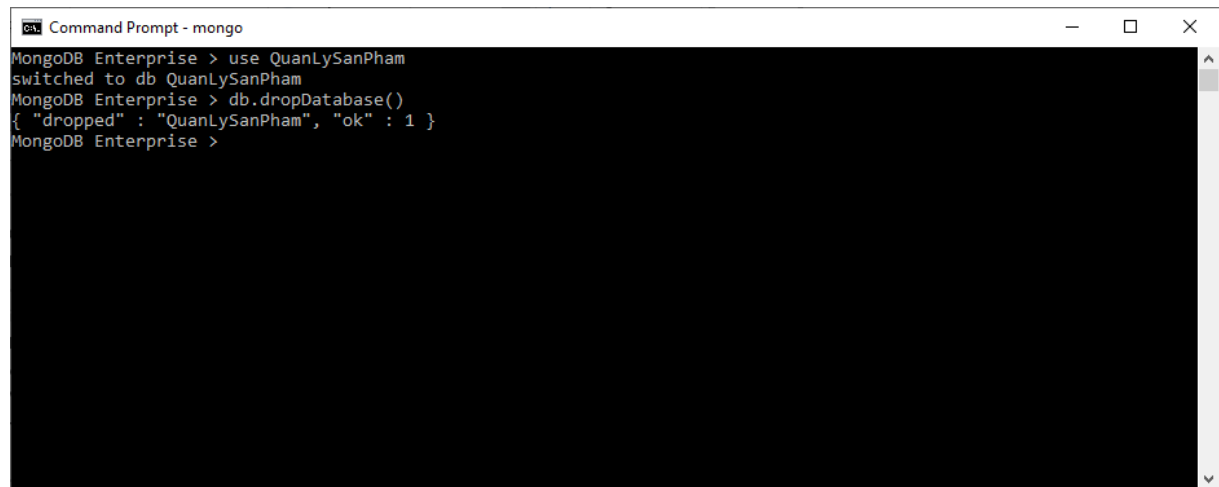
1. Lệnh Xóa database trong MongoDB

-Để xóa database trong MongoDB các bạn sử dụng cú pháp sau:

```
db.dropDatabase()
```

CHÚ Ý: Lệnh này dùng để xóa current database, nên trước khi sử dụng bắt buộc bạn phải switch vào database cần xóa thì mới thành công.

Ví dụ: Xóa database **QuanLySanPham**

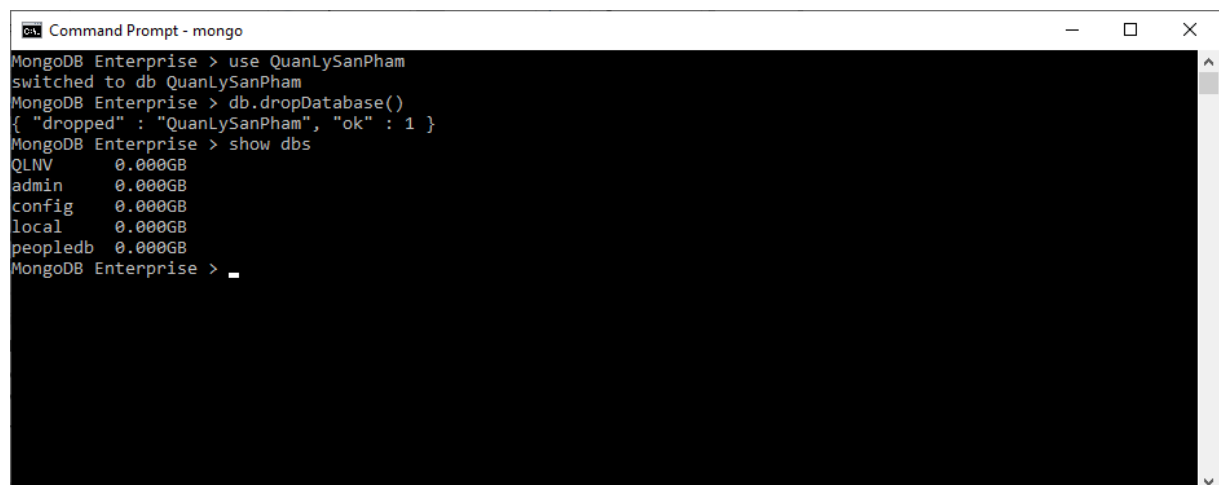


```
Command Prompt - mongo
MongoDB Enterprise > use QuanLySanPham
switched to db QuanLySanPham
MongoDB Enterprise > db.dropDatabase()
{ "dropped" : "QuanLySanPham", "ok" : 1 }
MongoDB Enterprise >
```

- Kết quả trả về dạng:

```
{ "dropped" : DatabaseName, "ok" : 1 }
```

Là bạn đã xóa thành công database trên MongoDB rồi, nhưng nếu như bạn cần kiểm tra một cách chắc chắn thì có thể dùng lệnh **show dbs** để kiểm tra lại:



```
Command Prompt - mongo
MongoDB Enterprise > use QuanLySanPham
switched to db QuanLySanPham
MongoDB Enterprise > db.dropDatabase()
{ "dropped" : "QuanLySanPham", "ok" : 1 }
MongoDB Enterprise > show dbs
QLNV      0.000GB
admin     0.000GB
config    0.000GB
local     0.000GB
peopledb  0.000GB
MongoDB Enterprise > _
```

Bài 6: Tạo và xóa collection trong MongoDB

1. Tạo collection trong MongoDB

- Nếu như trong các hệ quản trị cơ sở RDBMS trước đây có các table để lưu trữ dữ liệu, thì trong MongoDB chúng được thay thế với một khái niệm hoàn toàn mới là các Collection.

Để tạo collection trong MongoDB chúng ta sử dụng cú pháp:

```
db.createCollection(collectionName, option)
```

Trong đó:

- **collectionName** là tên của collection muốn tạo.
- **option** là một đối tượng chứa các tùy chọn riêng cho collection. Các tùy chọn bao gồm:
 - **capped** - đây là thông số cấu hình hành động sẽ xảy ra khi collection vượt quá dung lượng cho phép (thông số size). Nếu capped: true thì khi dung lượng quá hạn mức cho phép nó sẽ ghi đè các dữ liệu cũ nhất.
 - **autoIndexId** - đây là thông số cấu hình xem có đánh chỉ mục cho trường `_id` không. Nếu autoIndexId: true thì sẽ đánh chỉ mục cho trường `_id` (Phiên bản 3.4 tùy chọn này sẽ bị xóa).
 - **size** - xác định kích cỡ tối đa collection có thể chứa (đơn vị byte).
 - **max** - xác định số tài liệu tối đa mà một capped collection có thể chứa.
 - **storageEngine** - cấu hình storageEngine cho collection
 - **validator** - cấu hình định dạng cho dữ liệu của các trường
 - **validationLevel** - xác định độ nghiêm ngặt của validator ở trên. Giá trị có thể là:
 - ☞ "off" - không validator khi insert hoặc update.
 - ☞ "strict" - đây là giá trị mặc định, thiết lập validator với mọi câu lệnh insert và update.
 - ☞ "moderate" - thiết lập validator cho các rule được liệt kê ở validator, nếu trường nào không có thì sẽ không áp dụng.

- **validationAction** - thiết lập trạng thái khi dữ liệu không khớp với validator. Giá trị có thể điền vào là "error" hoặc "warn".
- **indexOptionDefaults**
- **viewOn**
- **pipeline**
- **collation**

Lưu ý: Để có thể thực hiện được lệnh này thì chúng ta cần khai báo sử dụng database.

Ví dụ 1: Tạo một collection có tên là Products trong database QuanLySanPham.

```
MongoDB Enterprise > use QuanLySanPham
switched to db QuanLySanPham
MongoDB Enterprise > db.createCollection("Products")
{ "ok" : 1 }
MongoDB Enterprise >
```

Ví dụ 2: Tạo một collection có tên là admin, và đồng thời validator dữ liệu cho collection.

```
MongoDB Enterprise >
db.createCollection("admin",{validator:{$or:[{name: {$type:
"string" }},{password: {$type: "string"}},{email: { $regex:
"/@gmail\\.com"}}]}}
)
{ "ok" : 1 }
MongoDB Enterprise >
```

Trong đoạn trên chúng ta đã ràng buộc dữ liệu cho trường name và password có kiểu dữ liệu là **string**, email bắt buộc phải có đuôi là **@gmail.com**

2. Xóa collection trong MongoDB

Để xóa một collection trong MongoDB chúng ta sử dụng cú pháp:

```
db.collectionName.drop()
```

Ví dụ: Xóa collection users

```
MongoDB Enterprise > db.users.drop()  
  
true  
  
MongoDB Enterprise >
```

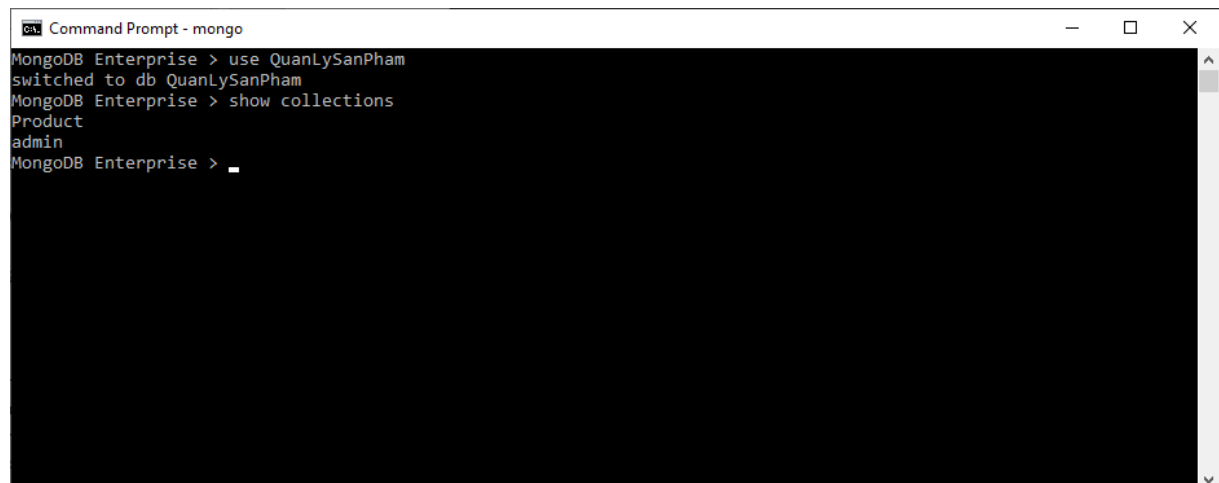
3. Xem danh sách các collection có trong database

Để xem danh sách các collection đang có trong database chúng ta sử dụng cú pháp:

```
show collections
```

Ví dụ: xem danh sách các collection có trong database QuanLySanPham

```
MongoDB Enterprise > use QuanLySanPham  
  
switched to db QuanLySanPham  
  
MongoDB Enterprise > show collections  
  
Product  
  
admin  
  
MongoDB Enterprise >
```



Bài 7: Insert dữ liệu trong MongoDB

1. Thêm mới dữ liệu vào trong MongoDB

MongoDB cung cấp 3 phương thức để thực hiện việc thêm mới dữ liệu vào trong collection:

- insert
- insertOne
- insertMany

Insert

Phương thức insert trong MongoDB dùng để thêm mới một hoặc nhiều dữ liệu (document) vào trong MongoDB.

Cú pháp:

```
db.conlectionName.insert(data)
```

Trong đó:

- collectionName là tên của collection chúng ta cần thêm dữ liệu vào.
- data có thể là 1 object chứa các trường và giá trị của nó hoặc cũng có thể là một mảng đối tượng (nếu như chúng ta muốn thêm nhiều document trong một lệnh).

Ví dụ 1: Thêm mới một dữ liệu vào collection có tên là admin

```
db.admin.insert({  
  name: "Vu Thanh Tai",  
  password: "admin",  
  email: "thanhtai96nd@gmail.com"  
})
```

Nếu như nInserted trả về là 1 tương đương với việc chúng ta đã thêm thành công một document vào trong MongoDB. Ứng với ví dụ trên thì có nghĩa là chúng ta đã thêm thành công.

Ví dụ 2: Thêm mới nhiều document vào collection có tên là admin

```
db.admin.insert([
  {
    name: "Vu Thanh Tai",
    password: "admin",
    email: "thanhtai96nd@gmail.com"
  },
  {
    name: "administrator",
    password: "admin123",
    email: "thanhtaivtt@toidicode.com"
  }
])
```

Với ví dụ này nếu như tham số `nInserted` trả về có giá trị là 2 thì tức là dữ liệu đã được thêm thành công.

insertOne

Phương thức `insertOne` trong MongoDB có tác dụng cho phép chúng ta insert **một document** trong một lệnh.

Cú pháp:

```
db.collectionName.insertOne(data)
```

Trong đó:

- `collectionName` là tên của collection chúng ta cần thêm document vào.
- `data` là một object chứa document chúng ta cần thêm vào.

Ví dụ 3: Thêm mới một document

```
db.admin.insertOne({
  name: "Vu Thanh Tai",
  password: "admin",
```

```
email: "thanhtai96nd@gmail.com"
})
```

- Nếu như thêm thành công thì hệ thống sẽ trả về `_id` của document vừa được thêm.

insertMany

Phương thức `insertMany` cho phép chúng ta thêm mới **nhiều document** vào trong MongoDB.

Cú pháp:

```
db.collectionName.insertMany(data)
```

- `collectionName` là tên của collection chúng ta cần thêm document vào.
- `data` là một mảng object chứa document chúng ta cần thêm vào.

Ví dụ 4: Thêm nhiều document

```
db.admin.insertMany([
  {
    name: "Vu Thanh Tai",
    password: "admin",
    email: "thanhtai96nd@gmail.com"
  },
  {
    name: "administrator",
    password: "admin123",
    email: "thanhtaivtt@toidicode.com"
  }
])
```

Nếu như thành công thì nó sẽ trả về `_id` của các dữ liệu vừa được thêm.

2. Chú ý

- Với cả ba phương thức trên nếu như `collectionName` chưa tồn tại trong hệ thống thì mặc định MongoDB sẽ tự động tạo mới và insert document vào.

Ví dụ: Giả sử trong hệ thống chưa tồn tại collection **posts** mà bạn thực hiện câu lệnh insert dữ liệu vào trong collection **posts** thì câu lệnh đó sẽ thực hiện tạo collection **posts** và thêm dữ liệu vào trong collection đó.

- Vì vậy, chúng ta phải hết sức chú ý khi thực hiện thêm mới dữ liệu.

Bài 8: Truy vấn dữ liệu trong MongoDB

1. Lấy tất cả document trong Collection

- Để lấy tất cả document trong collection chúng ta sử dụng phương thức `find()` với cú pháp:

```
db.collectionName.find()
```

Trong đó: `collectionName` là tên của collection mà chúng ta muốn truy vấn.

- Tuy nhiên, khi chỉ sử dụng mỗi phương thức `find()` thì các document trả về sẽ dưới dạng object, không theo một cấu trúc nào cả.

Ví dụ: Lấy tất cả document đang có trong Collection admin

```
db.admin.find()
```

- Dựa vào chức năng của phương thức `find()` thì chúng ta có thể so sánh nó tương đương như câu lệnh `Select * from` trong SQL.

- Và nếu như bạn muốn dữ liệu được trả về được hiển thị theo cấu trúc đã được định sẵn thì chỉ cần thêm phương thức `pretty()` vào phía sau phương thức `find()`

Cú pháp:

```
db.collectionName.find().pretty()
```

Ví dụ: Lấy hết document trong collection admin

```
db.admin.find().pretty()
```

```
C:\Windows\system32\cmd.exe - mongo
{ "_id" : ObjectId("5968a588363c4c2d9c4bbd05"), "name" : "Uu Thanh Tai", "password" : "admin", "email" : "thanhtai96nd@gmail.com" }
{ "_id" : ObjectId("5968a588363c4c2d9c4bbd06"), "name" : "administrator", "password" : "admin123", "email" : "thanhtai96nd@gmail.com" }
MongoDB Enterprise > db.admin.find().pretty()
{
  "_id" : ObjectId("59689cfb363c4c2d9c4bbd01"),
  "name" : "Uu Thanh Tai",
  "password" : "admin",
  "email" : "thanhtai96nd@gmail.com"
}
{
  "_id" : ObjectId("59689e8f363c4c2d9c4bbd02"),
  "name" : "Uu Thanh Tai",
  "password" : "admin",
  "email" : "thanhtai96nd@gmail.com"
}
{
  "_id" : ObjectId("59689e8f363c4c2d9c4bbd03"),
  "name" : "administrator",
  "password" : "admin123",
  "email" : "thanhtai96nd@gmail.com"
}
{
  "_id" : ObjectId("5968a093363c4c2d9c4bbd04"),
  "name" : "Uu Thanh Tai",
  "password" : "admin",
  "email" : "thanhtai96nd@gmail.com"
}
{
  "_id" : ObjectId("5968a588363c4c2d9c4bbd05"),
  "name" : "Uu Thanh Tai",
  "password" : "admin",
  "email" : "thanhtai96nd@gmail.com"
}
{
  "_id" : ObjectId("5968a588363c4c2d9c4bbd06"),
  "name" : "administrator",
  "password" : "admin123",
  "email" : "thanhtai96nd@gmail.com"
}
MongoDB Enterprise >
```

2. Truy vấn có điều kiện trong MongoDB

- Để truy vấn có điều kiện trong MongoDB thì bạn cũng sử dụng cú pháp tương tự như phần 1, nhưng lúc này chúng ta sẽ chèn thêm điều kiện vào trong phương thức find() với cú pháp sau:

```
db.collection.find(condition)
```

Trong đó:

- **collectionName** là tên của collection muốn truy vấn.
- **condition** là object chứa mệnh đề điều kiện. Theo các cú pháp sau đây:

Phép Toán	Cú Pháp	Ví dụ	Câu lệnh tương ứng trong SQL
Bằng (Equality)	{key: value}	db.admin.find({name: "Vu Thanh Tai"}).pretty()	.. WHERE name = "Vu Thanh Tai"
Nhỏ hơn (Less Than)	{key: {\$lt: value}}	db.admin.find({age: {\$lt: 18}}).pretty()	... WHERE age < 18
Nhỏ hơn bằng (Less Than Equals)	{key: {\$lte: value}}	db.admin.find({age: {\$lte: 18}}).pretty()	... WHERE age <= 18
Lớn hơn (Greater Than)	{key: {\$gt: value}}	db.admin.find({age: {\$gt: 12}}).pretty()	... WHERE age > 12
Lớn hơn bằng (Greater Than Equals)	{key: {\$gte: value}}	db.admin.find({age: {\$gte: 12}}).pretty()	... WHERE age >= 12
Khác (Not Equals)	{key: {\$ne: value}}	db.admin.find({age: {\$ne: 12}}).pretty()	... WHERE age != 12
Trong (In)	{key: {\$in: [value1, value2...]}}	db.admin.find({age: {\$in: [12, 18]}}).pretty()	... WHERE age IN (12, 18)
Không Thuộc (Not In)	{key: {\$nin: [value1, value2...]}}	db.admin.find({age: {\$nin: [12, 18]}}).pretty()	... WHERE age NOT IN (12, 18)

Ví dụ: in ra tất cả các admin có tên là **Vu Thanh Tai** có trong collection admin

```
db.admin.find({name: "Vu Thanh Tai"}).pretty()
```

```
C:\Windows\system32\cmd.exe - mongo
{"password": "admin",
"email": "thanhtai96nd@gmail.com"}
>
<
{"_id": ObjectId("5968a093363c4c2d9c4bbd04"),
"name": "Uu Thanh Tai",
"password": "admin",
"email": "thanhtai96nd@gmail;.com"}
>
<
{"_id": ObjectId("5968a588363c4c2d9c4bbd05"),
"name": "Uu Thanh Tai",
"password": "admin",
"email": "thanhtai96nd@gmail.com"}
>
MongoDB Enterprise > db.admin.insert<<name: "Uu Thanh Tai",age :18>>
WriteResult<< "ninserted" : 1 >>
MongoDB Enterprise > db.admin.find<<name: "Uu Thanh Tai">>.pretty<>
<
{"_id": ObjectId("59689cfb363c4c2d9c4bbd01"),
"name": "Uu Thanh Tai",
"password": "admin",
"email": "thanhtai96nd@gmail;.com"}
>
<
{"_id": ObjectId("59689e8f363c4c2d9c4bbd02"),
"name": "Uu Thanh Tai",
"password": "admin",
"email": "thanhtai96nd@gmail.com"}
>
<
{"_id": ObjectId("5968a093363c4c2d9c4bbd04"),
"name": "Uu Thanh Tai",
"password": "admin",
"email": "thanhtai96nd@gmail;.com"}
>
<
{"_id": ObjectId("5968a588363c4c2d9c4bbd05"),
"name": "Uu Thanh Tai",
"password": "admin",
"email": "thanhtai96nd@gmail.com"}
>
```

3. Truy vấn nhiều điều kiện trong MongoDB

- Trong MongoDB cũng có hỗ trợ chúng ta truy vấn nhiều điều kiện trong một lệnh, với các toán tử AND, OR như trong SQL.

AND

- Để thực hiện phép toán này thì các bạn chỉ cần thêm các điều kiện câu truy vấn vào trong object chứa điều kiện tìm kiếm vào trong phương thức find.

Ví dụ: Lấy ra admin có tên Vũ Thanh Tài và có tuổi là 18 trong collection admin

```
db.admin.find({
  name: "Vu Thanh Tai",
  age: 18
})
```

```
C:\Windows\system32\cmd.exe - mongo
MongoDB Enterprise > db.admin.find(<<name: "Vu Thanh Tai",age:18>>).pretty()
<
  "_id" : ObjectId<"5968de7f52f29fd1f78d16f2">,
  "name" : "Vu Thanh Tai",
  "age" : 18
>
MongoDB Enterprise >
```

- Tương tự, nếu như bạn muốn AND bao nhiêu điều kiện thì thêm bấy nhiêu vào trong object.

OR

- Để sử dụng mệnh đề OR (hoặc) trong MongoDB thì chúng ta cần phải truyền một key scope có tên là **\$or** vào làm key chứa mảng các điều kiện OR, theo cú pháp:

```
db.collectionName.find({
  $or : [
    {key1: value1},
    {key2: value2},
    ...,
    {keyn: valuen}
  ]
}).pretty()
```

Ví dụ: Lấy ra tất cả các admin có **tuổi bằng 12 hoặc name là Vu Thanh Tai** trong collection admin

```
db.admin.find({
  $or : [
    {age: 12},
    {name: "Vu Thanh Tai"},
  ]
}).pretty()
```

```
C:\Windows\system32\cmd.exe - mongo
MongoDB Enterprise > db.admin.find(<<$or: [ <age: 12>, <name: "Vu Thanh Tai"> ]>>
.pretty()
{
  "_id" : ObjectId("59689cfb363c4c2d9c4bbd01"),
  "name" : "Vu Thanh Tai",
  "password" : "admin",
  "email" : "thanhtai96nd@gmail;.com"
}
{
  "_id" : ObjectId("59689e8f363c4c2d9c4bbd02"),
  "name" : "Vu Thanh Tai",
  "password" : "admin",
  "email" : "thanhtai96nd@gmail.com"
}
{
  "_id" : ObjectId("5968a093363c4c2d9c4bbd04"),
  "name" : "Vu Thanh Tai",
  "password" : "admin",
  "email" : "thanhtai96nd@gmail;.com"
}
{
  "_id" : ObjectId("5968a588363c4c2d9c4bbd05"),
  "name" : "Vu Thanh Tai",
  "password" : "admin",
  "email" : "thanhtai96nd@gmail.com"
}
{
  "_id" : ObjectId("5968de7f52f29fd1f78d16f2"),
  "name" : "Vu Thanh Tai",
  "age" : 18
}
{
  "_id" : ObjectId("5968de8a52f29fd1f78d16f3"),
  "name" : "Vu Thanh Tai",
  "age" : 12
}
MongoDB Enterprise >
```

Kết hợp cả AND và OR

- Để kết hợp giữa AND và OR thì bạn chỉ cần làm tương tự như cách thực hiện truy vấn AND, và nếu truy vấn nào là OR thì object đó lại làm tương tự như truy vấn OR.

Ví dụ: Lấy ra tất cả các admin có **tuổi bằng 12 hoặc password bằng admin và có name là Vu Thanh Tai** trong collection admin.

```
db.admin.find({
  $or : [
    {age: 12},
    {password: "admin"},
  ],
  name: "Vu Thanh Tai"
}).pretty()
```

```
C:\Windows\system32\cmd.exe - mongo
"age" : 12
MongoDB Enterprise > db.admin.find(<<$or: [ <age: 12>, <password: "admin"> ], name: "Uu Thanh Tai">).pretty()
{
  "_id" : ObjectId<"59689cfb363c4c2d9c4bbd01">,
  "name" : "Uu Thanh Tai",
  "password" : "admin",
  "email" : "thanhtai96nd@gmail;.com"
}
{
  "_id" : ObjectId<"59689e8f363c4c2d9c4bbd02">,
  "name" : "Uu Thanh Tai",
  "password" : "admin",
  "email" : "thanhtai96nd@gmail.com"
}
{
  "_id" : ObjectId<"5968a093363c4c2d9c4bbd04">,
  "name" : "Uu Thanh Tai",
  "password" : "admin",
  "email" : "thanhtai96nd@gmail;.com"
}
{
  "_id" : ObjectId<"5968a588363c4c2d9c4bbd05">,
  "name" : "Uu Thanh Tai",
  "password" : "admin",
  "email" : "thanhtai96nd@gmail.com"
}
{
  "_id" : ObjectId<"5968de8a52f29fd1f78d16f3">,
  "name" : "Uu Thanh Tai",
  "age" : 12
}
MongoDB Enterprise >
```

4. Chọn lọc các trường cần lấy ra trong MongoDB

-Để chọn lọc các trường cần hiển thị ra trong 1 collection thì các bạn sử dụng phương thức find() với cú pháp sau:

```
db.collectionName.find(objectwhere, objectselect)
```

Trong đó:

- **objectwhere** là object chứa các điều kiện ở các phần trên. Nếu bạn không muốn lọc theo điều kiện thì bạn để một object rỗng vào.
- **objectselect** là object chứa các trường dữ liệu cần lấy ra. Mặc định thì nó sẽ lấy cả `_id`, nên nếu như bạn không muốn hiển thị `_id` thì bạn cần thêm `_id: 0` vào object.

Ví dụ: Lấy ra trường `_id`, `name`, `password` của tất cả các document có trong collection `admin`.

```
db.admin.find({}, {name: 1, password: 1}).pretty()
```



```
Windows Powershell
MongoDB Enterprise > db.admin.find(<<name: Vu Thanh Tai>>).pretty()
{
  "_id" : ObjectId("5968de8a52f29fd1f78d16f3"), "name" : "Vu Thanh Tai" }
{
  "_id" : ObjectId("59689cfb363c4c2d9c4bbd01"),
  "name" : "Vu Thanh Tai",
  "password" : "admin"
}
{
  "_id" : ObjectId("59689e8f363c4c2d9c4bbd02"),
  "name" : "Vu Thanh Tai",
  "password" : "admin"
}
{
  "_id" : ObjectId("59689e8f363c4c2d9c4bbd03"),
  "name" : "administrator",
  "password" : "admin123"
}
{
  "_id" : ObjectId("5968a093363c4c2d9c4bbd04"),
  "name" : "Vu Thanh Tai",
  "password" : "admin"
}
{
  "_id" : ObjectId("5968a588363c4c2d9c4bbd05"),
  "name" : "Vu Thanh Tai",
  "password" : "admin"
}
{
  "_id" : ObjectId("5968a588363c4c2d9c4bbd06"),
  "name" : "administrator",
  "password" : "admin123"
}
{
  "_id" : ObjectId("5968de7f52f29fd1f78d16f2"), "name" : "Vu Thanh Tai" }
{
  "_id" : ObjectId("5968de8a52f29fd1f78d16f3"), "name" : "Vu Thanh Tai" }
MongoDB Enterprise >
```

Ví dụ: Lấy ra name,password của những document có name = Vu Thanh Tai có trong admin collection

```
db.admin.find({name: "Vu Thanh Tai"},{name: 1, password: 1, _id: 0}).pretty()
```

```
Windows Powershell
MongoDB Enterprise > db.admin.find(<<name: Vu Thanh Tai>>,{name: 1, password: 1, _id: 0}).pretty()
{ "name" : "Vu Thanh Tai", "password" : "admin" }
{ "name" : "Vu Thanh Tai", "password" : "admin" }
{ "name" : "Vu Thanh Tai", "password" : "admin" }
{ "name" : "Vu Thanh Tai", "password" : "admin" }
{ "name" : "Vu Thanh Tai", "password" : "admin" }
{ "name" : "Vu Thanh Tai" }
MongoDB Enterprise >
```

Bài 9: Limit và order by document trong MongoDB

1. Giới hạn số lượng document trong MongoDB

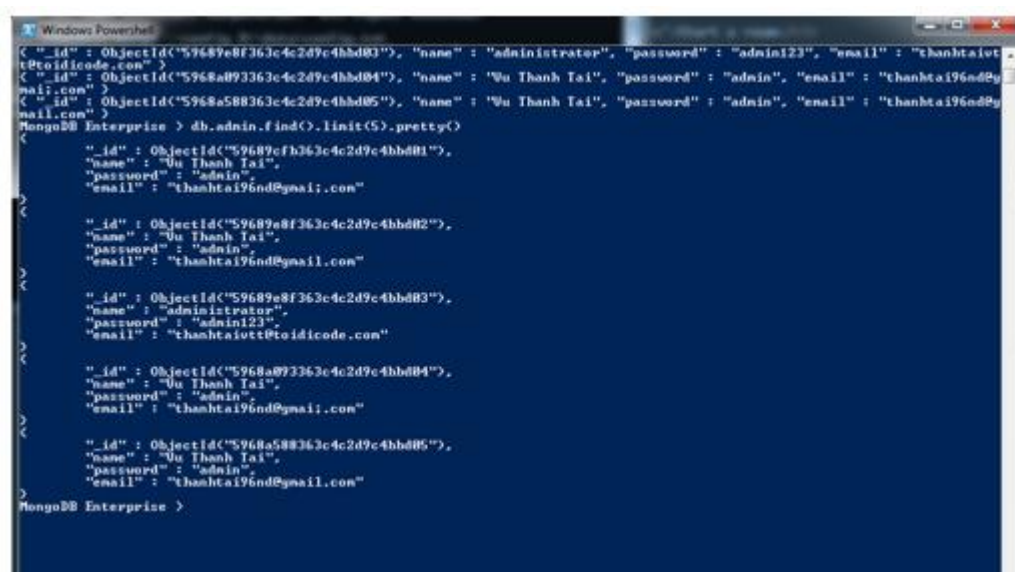
- Để có thể giới hạn số lượng document được lấy ra khi truy vấn thì bạn chỉ cần thêm phương thức **limit()** vào phía sau phương thức **find()** theo cú pháp sau:

```
db.collectionName.find().limit(number)
```

Trong đó: **number** là số lượng document muốn lấy ra.

Ví dụ 1: Lấy ra tối đa 5 document trong collection admin

```
db.admin.find().limit(5).pretty()
```



```
Windows PowerShell
PS C:\> mongo
MongoDB Enterprise > use admin
MongoDB Enterprise > db.admin.find().limit(5).pretty()
{
  "_id" : ObjectId("59689e8f363c4c2d9c4bbd03"), "name" : "administrator", "password" : "admin123", "email" : "thanhtaioct@toidicode.com"
}
{
  "_id" : ObjectId("5968a093363c4c2d9c4bbd04"), "name" : "Vu Thanh Tai", "password" : "admin", "email" : "thanhtai196nd@gmail.com"
}
{
  "_id" : ObjectId("5968a588363c4c2d9c4bbd05"), "name" : "Vu Thanh Tai", "password" : "admin", "email" : "thanhtai196nd@gmail.com"
}
{
  "_id" : ObjectId("59689ef363c4c2d9c4bbd01"), "name" : "Vu Thanh Tai", "password" : "admin", "email" : "thanhtai196nd@gmail.com"
}
{
  "_id" : ObjectId("59689e8f363c4c2d9c4bbd02"), "name" : "Vu Thanh Tai", "password" : "admin", "email" : "thanhtai196nd@gmail.com"
}
{
  "_id" : ObjectId("59689e8f363c4c2d9c4bbd03"), "name" : "administrator", "password" : "admin123", "email" : "thanhtaioct@toidicode.com"
}
{
  "_id" : ObjectId("5968a093363c4c2d9c4bbd04"), "name" : "Vu Thanh Tai", "password" : "admin", "email" : "thanhtai196nd@gmail.com"
}
{
  "_id" : ObjectId("5968a588363c4c2d9c4bbd05"), "name" : "Vu Thanh Tai", "password" : "admin", "email" : "thanhtai196nd@gmail.com"
}
MongoDB Enterprise >
```

2. Bỏ qua số lượng document trong MongoDB

- Trong một số trường hợp, bạn muốn bỏ qua một hoặc nhiều document đầu trong số document được lấy ra thì trong MongoDB đã cung cấp cho chúng ta phương thức **skip()** để thực hiện điều đó.

```
db.collectionName.find().skip(numberSkip)
```

Trong đó: **numberSkip** là số lượng document mà bạn muốn bỏ qua. Mặc định thì skip bằng 0

Ví dụ 2: Bỏ qua 3 document đầu trong số document được lấy ra

```
db.admin.find().skip(3)
```

```
Windows PowerShell
MongoDB Enterprise > db.admin.find().skip(3).pretty()
{
  "_id" : ObjectId<"5968a093363c4c2d9c4bbd04">,
  "name" : "Vu Thanh Tai",
  "password" : "admin",
  "email" : "thanhtai96nd@gmail.com"
}
{
  "_id" : ObjectId<"5968a588363c4c2d9c4bbd05">,
  "name" : "Vu Thanh Tai",
  "password" : "admin",
  "email" : "thanhtai96nd@gmail.com"
}
{
  "_id" : ObjectId<"5968a588363c4c2d9c4bbd06">,
  "name" : "administrator",
  "password" : "admin123",
  "email" : "thanhtaiott@toidicode.com"
}
{
  "_id" : ObjectId<"5968de7f52f29fd1f78d16f2">,
  "name" : "Vu Thanh Tai",
  "age" : 18
}
{
  "_id" : ObjectId<"5968de7f52f29fd1f78d16f3">,
  "name" : "Vu Thanh Tai",
  "age" : 12
}
MongoDB Enterprise >
```

3. Sắp xếp dữ liệu trong MongoDB

- Để có thể sắp xếp dữ liệu được trả về trong MongoDB thì các bạn sử dụng phương thức **sort()** theo cú pháp sau:

```
db.admin.find().sort({field: orderMode})
```

Trong đó:

- **field** là trường chọn để sắp xếp dữ liệu.
- **orderMode** là kiểu sắp xếp. Trong đó:
 - Nếu orderMode = 1 là sắp xếp theo chiều tăng dần.
 - Nếu orderMode = -1 là sắp xếp theo chiều giảm dần.

Ví dụ: Lấy ra tất cả các document có trong collection và sắp xếp theo độ dài của trường name giảm dần.

```
db.admin.find().sort({name: -1})
```

```
Windows PowerShell
"age" : 12
MongoDB Enterprise > db.admin.find().sort((name: -1)).pretty()
{
  "_id" : ObjectId("59689e8f363c4c2d9c4bbd83"),
  "name" : "administrator",
  "password" : "admin123",
  "email" : "thanhtaiott@toidicod.com"

  "_id" : ObjectId("5968a588363c4c2d9c4bbd86"),
  "name" : "administrator",
  "password" : "admin123",
  "email" : "thanhtaiott@toidicod.com"

  "_id" : ObjectId("59689cfb363c4c2d9c4bbd81"),
  "name" : "Vu Thanh Tai",
  "password" : "admin",
  "email" : "thanhtai96nd@gmail.com"

  "_id" : ObjectId("59689e8f363c4c2d9c4bbd82"),
  "name" : "Vu Thanh Tai",
  "password" : "admin",
  "email" : "thanhtai96nd@gmail.com"

  "_id" : ObjectId("5968a093363c4c2d9c4bbd84"),
  "name" : "Vu Thanh Tai",
  "password" : "admin",
  "email" : "thanhtai96nd@gmail.com"

  "_id" : ObjectId("5968a588363c4c2d9c4bbd85"),
  "name" : "Vu Thanh Tai",
  "password" : "admin",
  "email" : "thanhtai96nd@gmail.com"

  "_id" : ObjectId("5968de7f52f29fd1f78d16f2"),
  "name" : "Vu Thanh Tai",
  "age" : 18

  "_id" : ObjectId("5968de8a52f29fd1f78d16f3"),
  "name" : "Vu Thanh Tai",
  "age" : 12
}
MongoDB Enterprise >
```

Bài 10: Update dữ liệu trong MongoDB

1. Sửa đổi một document trong MongoDB

- Để sửa đổi một document duy nhất trong MongoDB thì các bạn sử dụng phương thức `updateOne()` theo cú pháp sau:

```
db.collectionName.updateOne(  
  filter,  
  update,  
  {  
    upsert: <boolean>,  
    writeConcern: <document>  
    collation: <document>,  
  }  
)
```

Trong đó:

- filter là một object chứa các tiêu chí lựa chọn document update (sử dụng cú pháp `selector`).
- update là object chứa dữ liệu sửa đổi trên document.
 - upsert là một boolean cấu hình điều gì sẽ xảy ra khi không có document khớp với filter. Nếu upsert = true thì nó sẽ thêm mới document đó nếu không có document nào khớp với filter và sẽ không có điều gì xảy ra nếu upsert = false. Mặc định thì upsert = false.
 - writeConcern là một document chứa write concern.
 - collation là một document chứa các quy tắc.

Lưu ý: Khi sử dụng phương thức `updateOne()` nếu như dữ liệu khớp với filter nhiều hơn một document thì nó sẽ chỉ sửa đổi cho một document đầu tiên.

Ví dụ 1: Sửa name của admin có tuổi = 18 thành Toidicode

```
db.admin.updateOne(  
  {age: 18},
```

```
{
  $set: {
    name: "Toidicode"
  }
}
```

```
PS C:\Users> mongo
MongoDB shell version v3.4.6
connecting to: mongod://127.0.0.1:27017
MongoDB server version: 3.4.6
Server has startup warnings:
2017-07-16T10:40:32.962+0700 I CONTROL [initandlisten]
2017-07-16T10:40:32.962+0700 I CONTROL [initandlisten] ** WARNING: Access control is not enabled for the database.
2017-07-16T10:40:32.962+0700 I CONTROL [initandlisten] **   Read and write access to data and configuration is u
restricted.
2017-07-16T10:40:32.962+0700 I CONTROL [initandlisten]
MongoDB Enterprise > use toidicode
switched to db toidicode
MongoDB Enterprise > db.admin.find()
{ "_id" : ObjectId("59689cfb363c4c2d9c4bbd01"), "name" : "Vu Thanh Tai", "password" : "admin", "email" : "thanhtai96nd@y
mail.com" }
{ "_id" : ObjectId("59689e8f363c4c2d9c4bbd02"), "name" : "Vu Thanh Tai", "password" : "admin", "email" : "thanhtai96nd@y
mail.com" }
{ "_id" : ObjectId("59689e8f363c4c2d9c4bbd03"), "name" : "administrator", "password" : "admin123", "email" : "thanhtai96
nd@y-mail.com" }
{ "_id" : ObjectId("5968a093363c4c2d9c4bbd04"), "name" : "Vu Thanh Tai", "password" : "admin", "email" : "thanhtai96nd@y
mail.com" }
{ "_id" : ObjectId("5968a588363c4c2d9c4bbd05"), "name" : "Vu Thanh Tai", "password" : "admin", "email" : "thanhtai96nd@y
mail.com" }
{ "_id" : ObjectId("5968a588363c4c2d9c4bbd06"), "name" : "administrator", "password" : "admin123", "email" : "thanhtai96
nd@y-mail.com" }
{ "_id" : ObjectId("5968de2f52f29fd1f78d16f2"), "name" : "Vu Thanh Tai", "age" : 18 }
{ "_id" : ObjectId("5968de8a52f29fd1f78d16f3"), "name" : "Vu Thanh Tai", "age" : 12 }
MongoDB Enterprise > db.admin.updateOne( { "_id" : ObjectId("5968de2f52f29fd1f78d16f2"), "name" : "Vu Thanh Tai", "age" : 18 }, { "$set" : { "age" : 12 } })
{ "acknowledged" : true, "matchedCount" : 1, "modifiedCount" : 1 }
MongoDB Enterprise > db.admin.find()
{ "_id" : ObjectId("59689cfb363c4c2d9c4bbd01"), "name" : "Vu Thanh Tai", "password" : "admin", "email" : "thanhtai96nd@y
mail.com" }
{ "_id" : ObjectId("59689e8f363c4c2d9c4bbd02"), "name" : "Vu Thanh Tai", "password" : "admin", "email" : "thanhtai96nd@y
mail.com" }
{ "_id" : ObjectId("59689e8f363c4c2d9c4bbd03"), "name" : "administrator", "password" : "admin123", "email" : "thanhtai96
nd@y-mail.com" }
{ "_id" : ObjectId("5968a093363c4c2d9c4bbd04"), "name" : "Vu Thanh Tai", "password" : "admin", "email" : "thanhtai96nd@y
mail.com" }
{ "_id" : ObjectId("5968a588363c4c2d9c4bbd05"), "name" : "Vu Thanh Tai", "password" : "admin", "email" : "thanhtai96nd@y
mail.com" }
{ "_id" : ObjectId("5968a588363c4c2d9c4bbd06"), "name" : "administrator", "password" : "admin123", "email" : "thanhtai96
nd@y-mail.com" }
{ "_id" : ObjectId("5968de2f52f29fd1f78d16f2"), "name" : "Toidicode", "age" : 12 }
{ "_id" : ObjectId("5968de8a52f29fd1f78d16f3"), "name" : "Vu Thanh Tai", "age" : 12 }
```

2. Sửa đổi nhiều bản ghi trong MongoDB

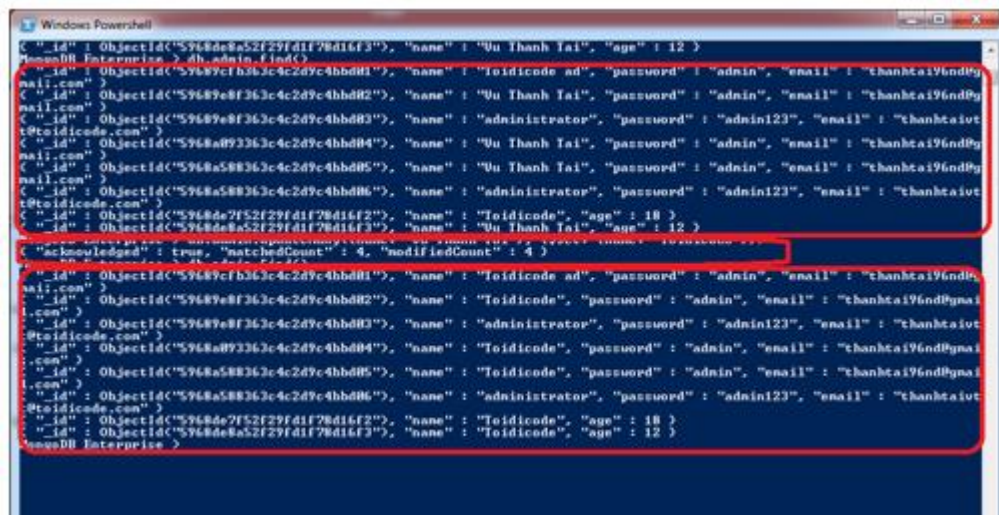
- Để sửa nhiều document trong một lệnh trong MongoDB thì chúng ta sử dụng phương thức `updateMany()` với cú pháp tương tự như phương thức `updateOne()`:

```
db.collectionName.updateOne(
  filter,
  update,
  {
    upsert: <boolean>,
    writeConcern: <document>,
    collation: <document>,
  }
)
```


Lưu ý: Phương thức này chỉ khác với phương thức `updateOne()` ở chỗ: Nếu như số lượng document so khớp với filter lớn hơn 1 document thì nó sẽ sửa dữ liệu trên tất cả các document đó.

Ví dụ 2: Sửa name của admin có name = "Vu Thanh Tai" thành Toidicode

```
db.admin.updateMany(  
  {name: "Vu Thanh Tai"},  
  {  
    $set: {  
      name: "Toidicode"  
    }  
  }  
)
```



```
Windows PowerShell  
PS C:\> mongoDB Enterprise > db.admin.find()  
{ "_id" : ObjectId("5968de8a52f29fd1f78d16f3"), "name" : "Vu Thanh Tai", "age" : 12 }  
{ "_id" : ObjectId("59689c1b363c4c2d9c4bbd81"), "name" : "Toidicode ad", "password" : "admin", "email" : "thanhtai96nd@mai.com", "age" : 12 }  
{ "_id" : ObjectId("59689c1b363c4c2d9c4bbd82"), "name" : "Vu Thanh Tai", "password" : "admin", "email" : "thanhtai96nd@mai.com", "age" : 12 }  
{ "_id" : ObjectId("59689c1b363c4c2d9c4bbd83"), "name" : "administrator", "password" : "admin123", "email" : "thanhtai96nd@mai.com", "age" : 12 }  
{ "_id" : ObjectId("5968a093363c4c2d9c4bbd84"), "name" : "Vu Thanh Tai", "password" : "admin", "email" : "thanhtai96nd@mai.com", "age" : 12 }  
{ "_id" : ObjectId("5968a588363c4c2d9c4bbd85"), "name" : "Vu Thanh Tai", "password" : "admin", "email" : "thanhtai96nd@mai.com", "age" : 12 }  
{ "_id" : ObjectId("5968a588363c4c2d9c4bbd86"), "name" : "administrator", "password" : "admin123", "email" : "thanhtai96nd@mai.com", "age" : 12 }  
{ "_id" : ObjectId("5968de2f52f29fd1f78d16f2"), "name" : "Toidicode", "age" : 18 }  
{ "_id" : ObjectId("5968de8a52f29fd1f78d16f3"), "name" : "Vu Thanh Tai", "age" : 12 }  
{ "acknowledged" : true, "matchedCount" : 4, "modifiedCount" : 4 }  
PS C:\> mongoDB Enterprise > db.admin.updateMany(  
  {name: "Vu Thanh Tai"},  
  {  
    $set: {  
      name: "Toidicode"  
    }  
  }  
)  
{ "_id" : ObjectId("59689c1b363c4c2d9c4bbd81"), "name" : "Toidicode ad", "password" : "admin", "email" : "thanhtai96nd@mai.com", "age" : 12 }  
{ "_id" : ObjectId("59689c1b363c4c2d9c4bbd82"), "name" : "Toidicode", "password" : "admin", "email" : "thanhtai96nd@mai.com", "age" : 12 }  
{ "_id" : ObjectId("59689c1b363c4c2d9c4bbd83"), "name" : "administrator", "password" : "admin123", "email" : "thanhtai96nd@mai.com", "age" : 12 }  
{ "_id" : ObjectId("5968a093363c4c2d9c4bbd84"), "name" : "Toidicode", "password" : "admin", "email" : "thanhtai96nd@mai.com", "age" : 12 }  
{ "_id" : ObjectId("5968a588363c4c2d9c4bbd85"), "name" : "Toidicode", "password" : "admin", "email" : "thanhtai96nd@mai.com", "age" : 12 }  
{ "_id" : ObjectId("5968a588363c4c2d9c4bbd86"), "name" : "administrator", "password" : "admin123", "email" : "thanhtai96nd@mai.com", "age" : 12 }  
{ "_id" : ObjectId("5968de2f52f29fd1f78d16f2"), "name" : "Toidicode", "age" : 18 }  
{ "_id" : ObjectId("5968de8a52f29fd1f78d16f3"), "name" : "Toidicode", "age" : 12 }  
PS C:\> mongoDB Enterprise >
```

3. Sửa đổi document trong MongoDB

Ngoài 2 phương thức trên thì trong MongoDB còn cung cấp cho chúng ta một phương thức `update()` có thể cấu hình `updateOne()` hoặc `updateMany()`, sử dụng theo cú pháp sau:

```
db.collection.update(  
  filter,  
  update,  
  {
```

```

    upsert: <boolean>,
    multi: <boolean>,
    writeConcern: <document>,
    collation: <document>
  }
)

```

Trong đó:

- filter là một object chứa các tiêu chí lựa chọn document update (sử dụng cú pháp selector).
- update là object chứa dữ liệu sửa đổi trên document.
- upsert là một boolean cấu hình điều gì sẽ xảy ra khi không có document khớp với filter. Nếu upsert = true thì nó sẽ thêm mới document đó nếu không có document nào khớp với filter và sẽ không có điều gì xảy ra nếu upsert = false. Mặc định thì upsert = false.
- multi là một boolean cấu hình xem có cho phép sửa đổi nhiều document hay không, multi bằng true là cho phép và ngược lại bằng false thì là không. Mặc định thì thuộc tính này có giá trị là false.
- writeConcern là một document chứa write concern.
- collation là một document chứa các quy tắc.

Ví dụ 3: Sửa đổi name của một document duy nhất có name là "Toidicode" thành "Vu Thanh Tai"

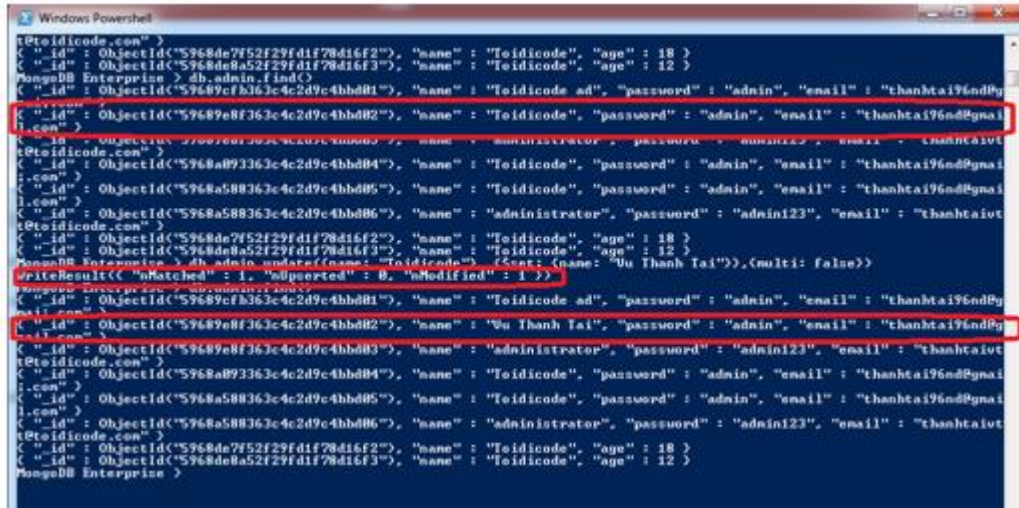
```

db.admin.updateOne(
  {name: "Vu Thanh Tai"},
  {
    $set: {
      name: "Toidicode"
    }
  },
  {

```



```
multi : false
```



```
Windows PowerShell
PS C:\> mongo
MongoDB Enterprise > use teidicode
MongoDB Enterprise > db.admin.find()
{ "_id" : ObjectId("5968de7f52f29fd1f78d16f2"), "name" : "Teidicode", "age" : 18 }
{ "_id" : ObjectId("5968de8a52f29fd1f78d16f3"), "name" : "Teidicode", "age" : 12 }
MongoDB Enterprise > db.admin.find()
{ "_id" : ObjectId("59689cfb363c4c2d9c4bbd81"), "name" : "Teidicode ad", "password" : "admin", "email" : "thanhtai96nd@gmail.com" }
{ "_id" : ObjectId("59689cfb363c4c2d9c4bbd82"), "name" : "Teidicode", "password" : "admin", "email" : "thanhtai96nd@gmail.com" }
{ "_id" : ObjectId("5968a093363c4c2d9c4bbd84"), "name" : "Teidicode", "password" : "admin", "email" : "thanhtai96nd@gmail.com" }
{ "_id" : ObjectId("5968a588363c4c2d9c4bbd85"), "name" : "Teidicode", "password" : "admin", "email" : "thanhtai96nd@gmail.com" }
{ "_id" : ObjectId("5968a588363c4c2d9c4bbd86"), "name" : "administrator", "password" : "admin123", "email" : "thanhtai96nd@gmail.com" }
{ "_id" : ObjectId("5968de7f52f29fd1f78d16f2"), "name" : "Teidicode", "age" : 18 }
{ "_id" : ObjectId("5968de8a52f29fd1f78d16f3"), "name" : "Teidicode", "age" : 12 }
MongoDB Enterprise > db.admin.find({'name': 'Vu Thanh Tai'}).multi: false
WriteResult<< "nMatched" : 1, "nUpserted" : 0, "nModified" : 1 >>
MongoDB Enterprise > db.admin.find()
{ "_id" : ObjectId("59689cfb363c4c2d9c4bbd81"), "name" : "Teidicode ad", "password" : "admin", "email" : "thanhtai96nd@gmail.com" }
{ "_id" : ObjectId("59689cfb363c4c2d9c4bbd82"), "name" : "Vu Thanh Tai", "password" : "admin", "email" : "thanhtai96nd@gmail.com" }
{ "_id" : ObjectId("5968a093363c4c2d9c4bbd84"), "name" : "Teidicode", "password" : "admin", "email" : "thanhtai96nd@gmail.com" }
{ "_id" : ObjectId("5968a588363c4c2d9c4bbd85"), "name" : "Teidicode", "password" : "admin", "email" : "thanhtai96nd@gmail.com" }
{ "_id" : ObjectId("5968a588363c4c2d9c4bbd86"), "name" : "administrator", "password" : "admin123", "email" : "thanhtai96nd@gmail.com" }
{ "_id" : ObjectId("5968de7f52f29fd1f78d16f2"), "name" : "Teidicode", "age" : 18 }
{ "_id" : ObjectId("5968de8a52f29fd1f78d16f3"), "name" : "Teidicode", "age" : 12 }
MongoDB Enterprise >
```

Bài 11: Xóa dữ liệu trong MongoDB

1. Xóa dữ liệu trong MongoDB

- Để có thể xóa dữ liệu trong MongoDB thì các bạn sử dụng phương thức `remove()` với cú pháp sau:

```
db.collectionName.remove(
    query,
    {
        justOne: <boolean>,
        writeConcern: <document>,
        collation: <document>
    }
)
```

Trong đó:

- **collectionName** là tên của collection mà bạn muốn thực thi lệnh xóa document.
- **query** là object (hay còn gọi là document) chứa các **câu truy vấn** để lọc dữ liệu.

- **justOne** là tham số cấu hình số lượng document có thể xóa khi query thực thi khớp.
 - Nếu **justOne: true** thì nó sẽ chỉ xóa 1 document duy nhất.
 - Nếu **justOne: false** thì nó sẽ xóa tất cả các document khớp với điều kiện query.
- **writeConcern** là một document chứa write concern.
- **collation** là một document chứa các quy tắc.

2. Ví dụ

Ví dụ: Xóa một admin có name = "NguyenVanA" và có age = 18

```
db.admin.remove(
{
  name: "NguyenVanA",
  age: 18
})
```

```
PS C:\Users\thanhtai> mongo
MongoDB shell version v3.4.6
connecting to: mongodb://127.0.0.1:27017
MongoDB server version: 3.4.6
Server has startup warnings:
2017-07-17T18:58:43.740+0700 I CONTROL [initandlisten]
2017-07-17T18:58:43.740+0700 I CONTROL [initandlisten] == WARNING: Access control is not enabled for the database.
2017-07-17T18:58:43.741+0700 I CONTROL [initandlisten] == Read and write access to data and configuration is unrestricted.
2017-07-17T18:58:43.741+0700 I CONTROL [initandlisten]
MongoDB Enterprise > use toidicode
switched to db toidicode
MongoDB Enterprise > db.admin.find()
{ "_id" : ObjectId("59687cfb363c4c2d9c4bbd81"), "name" : "Toidicode ad", "password" : "admin", "email" : "thanhtai96nd@gmail.com" }
{ "_id" : ObjectId("59687e8f363c4c2d9c4bbd82"), "name" : "Vu Thanh Tai", "password" : "admin", "email" : "thanhtai96nd@gmail.com" }
{ "_id" : ObjectId("59687e8f363c4c2d9c4bbd83"), "name" : "administrator", "password" : "admin123", "email" : "thanhtai96nd@gmail.com" }
{ "_id" : ObjectId("5968a093363c4c2d9c4bbd84"), "name" : "Toidicode", "password" : "admin", "email" : "thanhtai96nd@gmail.com" }
{ "_id" : ObjectId("5968a588363c4c2d9c4bbd85"), "name" : "Toidicode", "password" : "admin", "email" : "thanhtai96nd@gmail.com" }
{ "_id" : ObjectId("5968a588363c4c2d9c4bbd86"), "name" : "administrator", "password" : "admin123", "email" : "thanhtai96nd@gmail.com" }
{ "_id" : ObjectId("5968de7f52f29fd1f78d16f2"), "name" : "Toidicode", "age" : 18 }
MongoDB Enterprise > db.admin.remove({'name': 'Toidicode', 'age': 18})
WriteResult(1)
```