Nhóm 23

Đề tài: Phát triển service xử lý và lưu trữ video

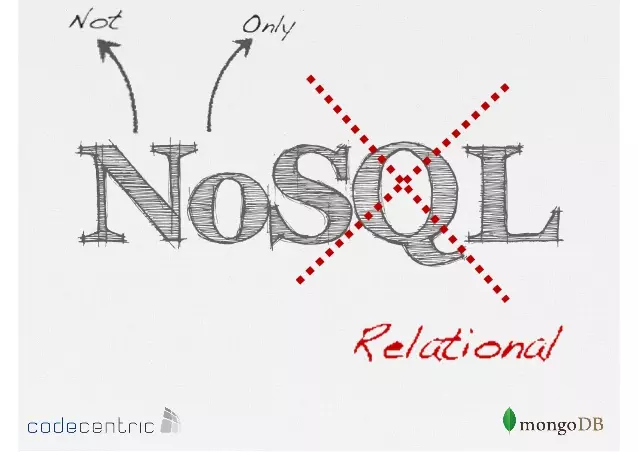
Họ và tên: Hà Vĩ Khang

MSSV: 20110657

Task 2: Tìm hiểu MongoDB và cài đặt MongoDB

**1. Định nghĩa 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++.
* 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.



*(\*) 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”) 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 về tốc độ, tính năng, khả năng mở rộng, memory cache,...*

**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.
* **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.

**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)

Ví dụ cho Embedded Documents:

\_id: ObjectId("5099803df3f4948bd2f98391"),

name: { first: "Alan", last: "Turing" },

contact: { phone: { type: "cell", number: "111-222-3333" } },

}

Để chỉ trường last trong trường name, ta dùng “name.last” Để chỉ number trong phone mà phone lại nằm trong trường contact, ta dùng “contact.phone.number”

(\*) key \_id là do hệ thống tự tạo ra để làm khóa chính.

**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” |
| 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" } } )

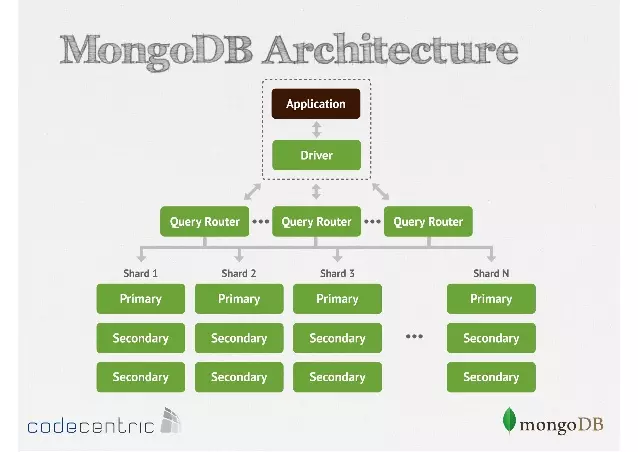
hoặc có thể viết

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

**5. Một số câu lệnh dùng trong MongoDB**

| **Câu lệnh** | **SQL** | **MongoDB** |
| --- | --- | --- |
| Create table | CREATE TABLE people (id MEDIUMINT NOT NULL AUTO\_INCREMENT, user\_id Varchar(30), age Number, status char(1), PRIMARY KEY (id)) | db.people.insertOne({User\_id: “abc123”, Age: 55, Status: “A”}) |
| Drop table | DROP TABLE people | db.people.drop() |
| Insert records into tables | INSERT INTO people(user\_id, age, status) VALUES ("bcd001", 45, "A") | db.people.insertOne( { user\_id: "bcd001", age: 45, status: "A" }) |
| Select | SELECT \*FROM people | db.people.find() |
|  | SELECT id,user\_id, status FROM people | db.people.find( { }, { user\_id: 1, status: 1 } ) |
|  | SELECT \* FROM people WHERE status = "A" | db.people.find( { status: "A" } ) |
|  | SELECT \* FROM people WHERE status = "A" AND age = 50 | db.people.find( { status: "A", age: 50 } ) |
|  | SELECT \* FROM people WHERE status = "A" OR age = 50 | db.people.find( { $or: [ { status: "A" } , { age: 50 } ] } ) |
|  | SELECT \* FROM people WHERE user\_id like "%bc%" | db.people.find( { user\_id: /bc/ } ) db.people.find( { user\_id: { $regex: /bc/ } } ) |
|  | SELECT COUNT(user\_id) FROM people | db.people.count( { user\_id: { $exists: true } } ) db.people.find( { user\_id: { $exists: true } } ).count() |
| Update records | UPDATE people SET status = "C" WHERE age > 25 | db.people.updateMany( { age: { $gt: 25 } }, { $set: { status: "C" } } ) |
|  | UPDATE people SET age = age + 3 WHERE status = "A" | db.people.updateMany( { status: "A" } , { $inc: { age: 3 } } ) |
| Delete Records | DELETE FROM people WHERE status = "D" | db.people.deleteMany( { status: "D" } ) |
|  | DELETE FROM people | db.people.deleteMany({}) |

**6. MongoDB hoạt động như thế nào**



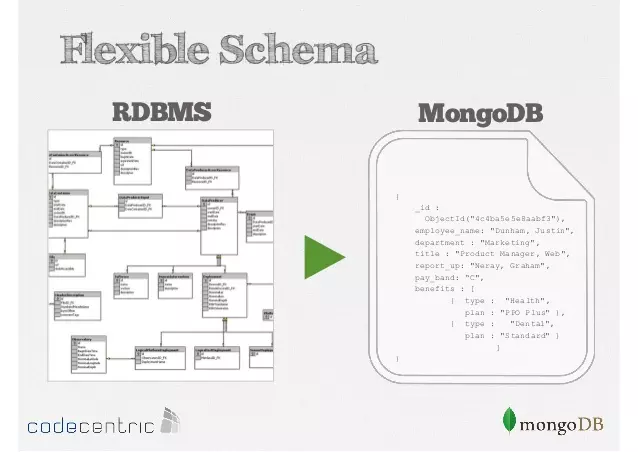
* MongoDB hoạt động dưới một tiến trình ngầm service, luôn mở một cổng (Cổng mặc định là 27017) để lắng nghe các yêu cầu truy vấn, thao tác từ các ứng dụng gửi vào sau đó mới tiến hành xử lý.
* Mỗi một bản ghi của MongoDB được tự động gắn thêm một field có tên “\_id” thuộc kiểu dữ liệu ObjectId mà nó 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.

**Từ đây có thể nhìn thấy nhược điểm của Mongodb như sau:**

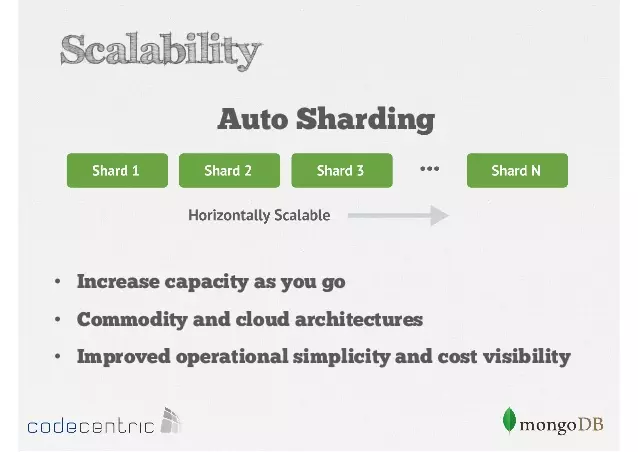
* Dữ liệu được caching, lấy RAM làm trọng tâm hoạt động vì vậy khi hoạt động yêu cầu một bộ nhớ RAM lớn
* Như đã giới thiệu ở trên, mọi thay đổi về dữ liệu mặc định đều chưa được ghi xuống ổ cứng ngay lập tức vì vậy khả năng bị mất dữ liệu từ nguyên nhân mất điện đột xuất là rất cao.

**7. Lợi thế của MongoDB**

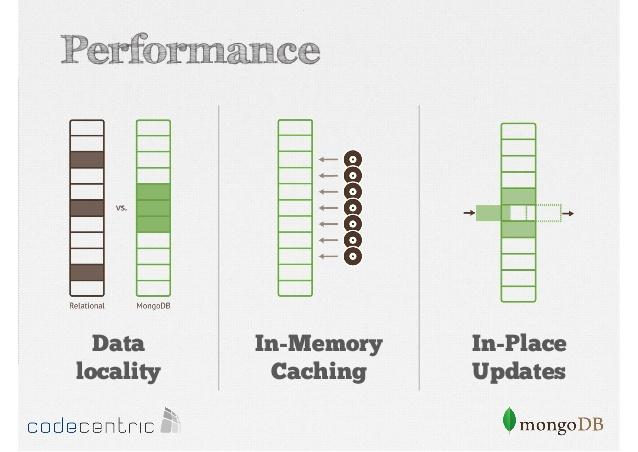
* Ít schema hơn: Vì schema được sinh ra là để nhóm các đối tượng vào 1 cụm, dễ quản lý. Ví dụ như tạo 1 schema tên là Students chẳng hạn thì chỉ có những gì liên quan đến student thì mới được cho vào schema này. Trong khi đó trong mongodb thì chỉ 1 collection ta có thể chứa nhiều document khác nhau .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



**8. Khi nào NÊN sử dụng MongoDB ?**

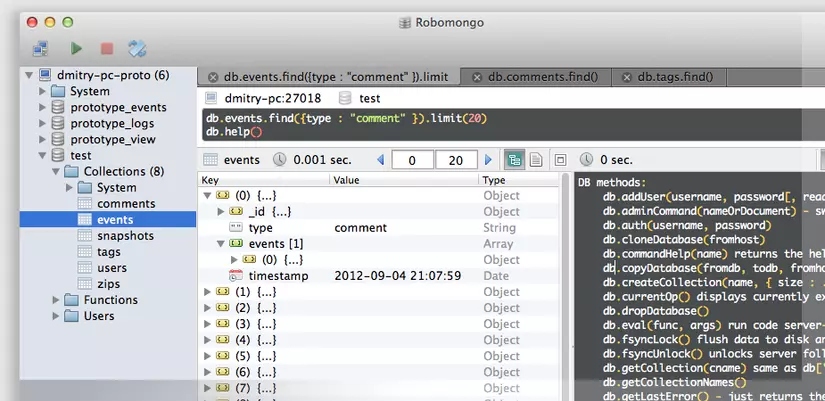
Sử dụng MongoDB trong trường hợp:

* 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.

**9. 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).

**10. Các công cụ quản trị MongoDB**

****

Một số công cụ điển hình như:

* RoboMongo
* UMongo(trước đây là JMongoBrower)
* MongoExplorer
* RockMongo

**11. Cài đặt MongoDB**

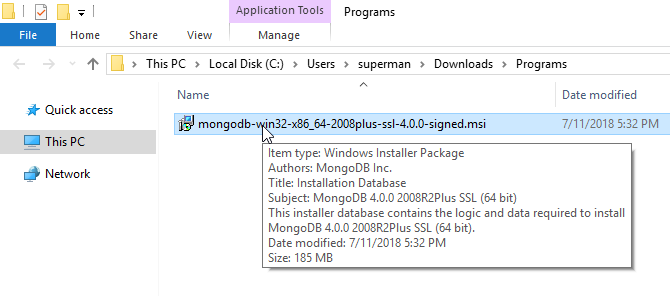
- Trên Windows:

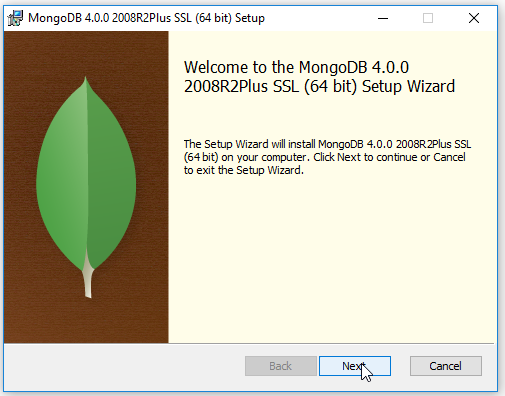
Tải MongoDB từ link bên dưới: https://www.mongodb.org/downloads

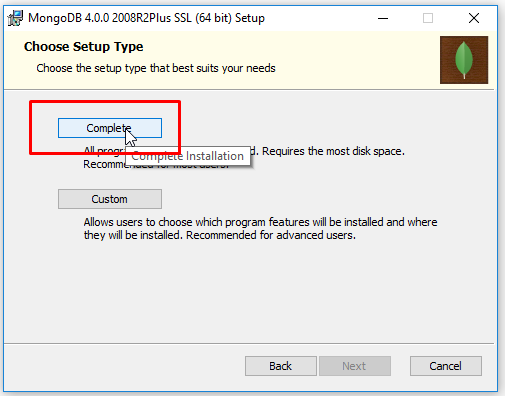
Chạy file vừa tải về

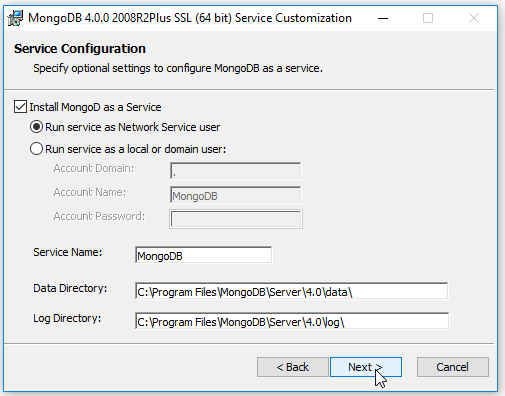
Sau khi chạy xong file, vào thư mục C:\Program Files\MongoDB\Server\3.4\bin , chạy file mongod để khởi động mongoDB

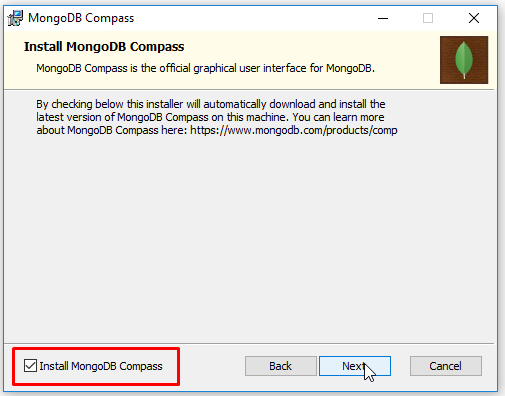
Dùng địa chỉ 127.0.0.1:27017 để tạo connection đến mongoDB

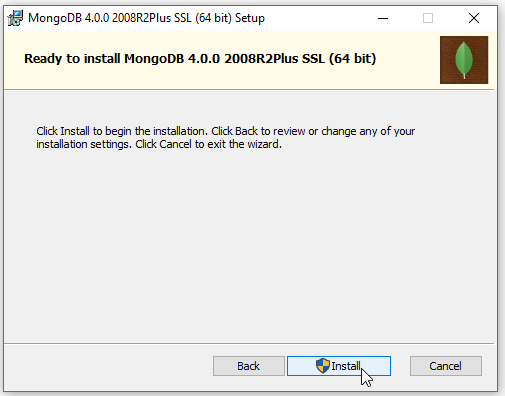


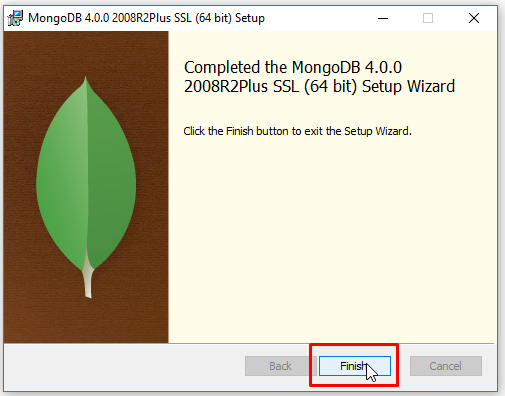




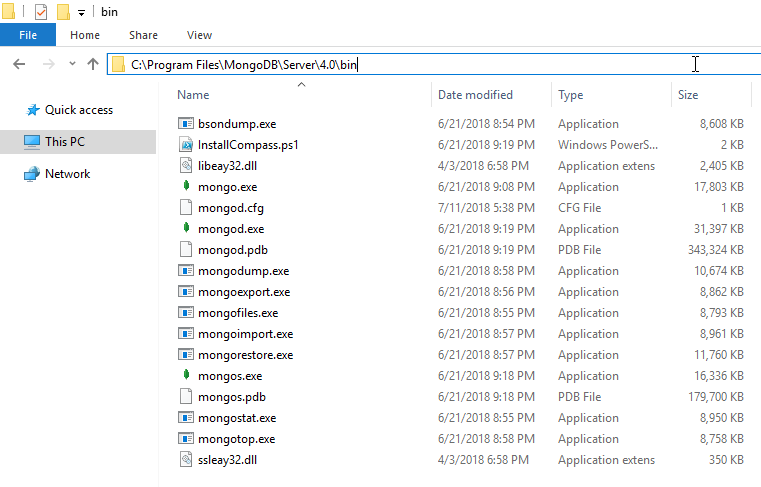






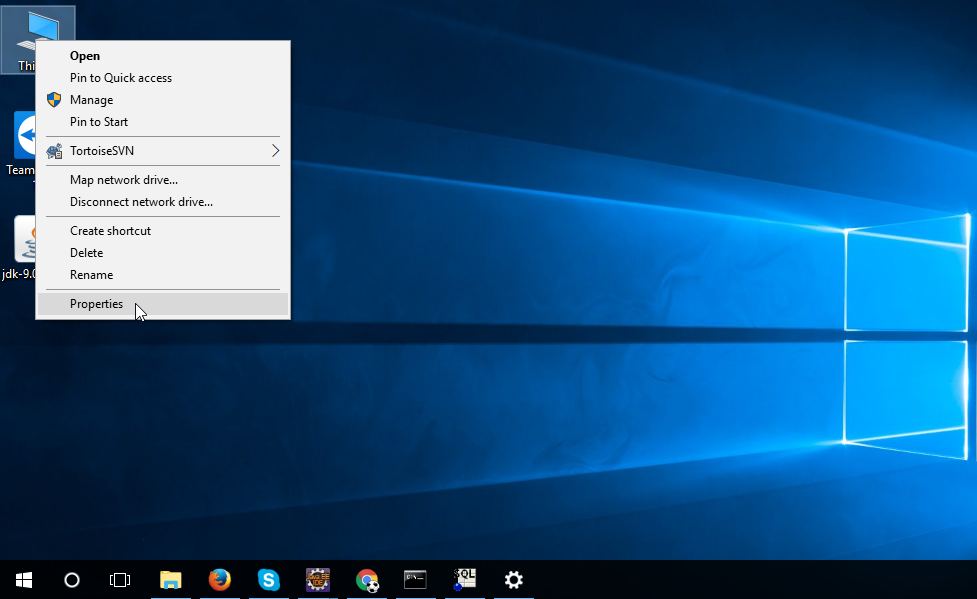


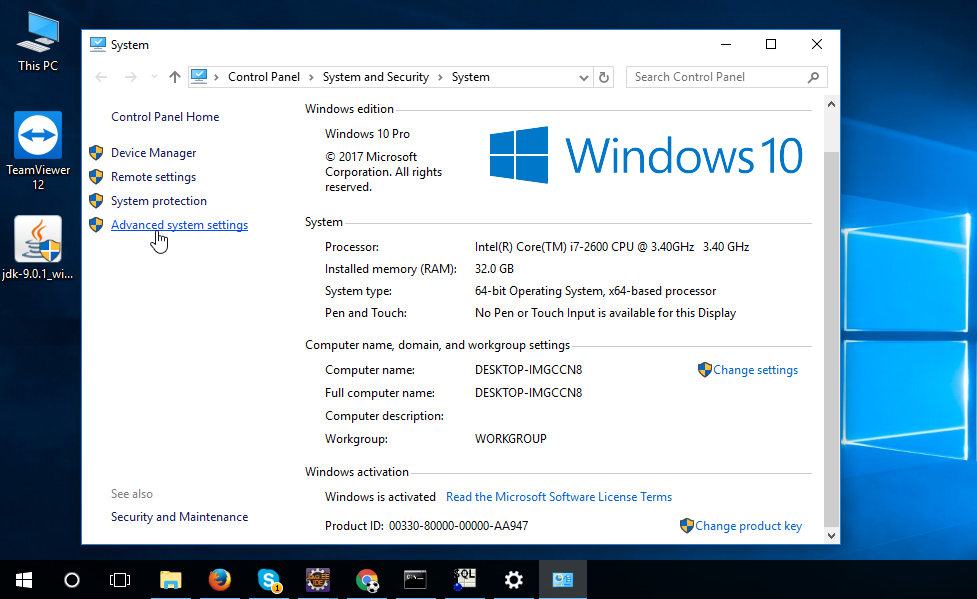
Nơi chứa phần mềm MongoDB sau khi cài đặt.



**Tạo biến môi trường cho MongoDB**

Chuột phải vào biểu tượng computer hoặc vào computer và click chuột phải -> chọn Properites





A screenshot of a computer

Description automatically generated

