

ĐẠI HỌC QUỐC GIA THÀNH PHỐ HỒ CHÍ MINH
TRƯỜNG ĐẠI HỌC CÔNG NGHỆ THÔNG TIN



BÁO CÁO CUỐI KỲ
MÔN: QUẢN LÝ THÔNG TIN
ĐỀ TÀI: TÌM HIỂU VỀ MONGODB

Giảng viên:

Tiến Sĩ - Nguyễn Gia Tuấn Anh

Cử Nhân - Trần Quốc Khánh

Sinh viên thực hiện:

Nguyễn Văn Toàn	21521549
-----------------	----------

Nguyễn Minh Lý	21521108
----------------	----------

Nguyễn Thị Kim Ngân	21521174
---------------------	----------

Thành phố Hồ Chí Minh, tháng 5 năm 2023

BÁO CÁO TÓM TẮT

1. Tiêu đề báo cáo: MONGODB.

2. Danh sách thành viên

MSSV	Họ tên	Ghi chú
21521549	Nguyễn Văn Toàn.	
21521108	Nguyễn Minh Lý.	
21521174	Nguyễn Thị Kim Ngân	

3. Nội dung chi tiết

Lời mở đầu

Chương 1: Tìm hiểu về Mongoddb.

Chương 2: Mô hình dữ liệu trong MongoDB

Chương 3: Các thao tác trên Mongoddb.

Chương 4: Kết luận và định hướng phát triển.

4. Phân công công việc

MSSV	Họ tên	Nội dung được phân công
21521549	Nguyễn Văn Toàn.	Chương 1: mục 1.2, 1.3 Chương 3: mục 3.1, mục 3.2
21521108	Nguyễn Minh Lý.	Chương 1: mục 1.4 Chương 3: mục 3.2 Chương 4
21521174	Nguyễn Thị Kim Ngân	Chương 1: Mục 1 Chương 2 Chương 3: Mục 3.2 Lời mở đầu

Mục lục

BÁO CÁO TÓM TẮT	1
LỜI MỞ ĐẦU	6
1. Đặt vấn đề.....	6
2. Mục tiêu.....	6
3. Phương pháp nghiên cứu.....	6
CHƯƠNG 1: GIỚI THIỆU VỀ MONGODB	7
1. So sánh giữa NoSQL và SQL	7
2. Khái quát sơ lược về MongoDB	9
3. Ưu điểm và nhược điểm của Mongodb.	9
3.1. <i>Ưu điểm của MongoDB:</i>	9
3.2. <i>Nhược điểm của MongoDB:</i>	10
4. Cách cài đặt và sử dụng Mongodb:	10
CHƯƠNG 2: MÔ HÌNH DỮ LIỆU TRONG MONGODB.....	14
1. Mô hình dữ liệu nhúng	14
2. Mô hình dữ liệu chuẩn hóa	15
CHƯƠNG 3: CÁC THAO TÁC CỦA MONGODB	16
1. Các thao tác cơ bản.....	16
1.1. <i>Sử dụng giao diện màn hình</i>	16
1.2. <i>Sử dụng bằng vscode:</i>	19
2. Các câu lệnh cơ bản	20
2.1. <i>Tạo database</i>	20
2.2. <i>Tạo collection</i>	20
2.3. <i>Thêm dữ liệu</i>	21
2.4. <i>Tìm kiếm dữ liệu</i>	22
2.5. <i>Xóa 1 database</i>	22
2.6. <i>Xóa 1 collection</i>	23
2.7. <i>Xóa dữ liệu trong collection</i>	23
2.8. <i>Update dữ liệu</i>	24
2.9. <i>Toán tử \$project</i>	25
2.10. <i>Toán tử \$match</i>	27
2.11. <i>Toán tử \$sort</i>	28
2.12. <i>Toán tử \$group</i>	29
2.13. <i>Toán tử \$unwind</i>	30
2.14. <i>Toán tử \$lookup</i>	32

3. Các thao tác nâng cao trong MongoDB	35
3.1. Cú pháp tạo user:	35
3.2. Xóa một user	36
3.3. Backup data	36
3.4. Restore data	38
3.5. Embeded Relationship	38
3.6. Sử dụng DBRefs	40
3.7. Chỉ Mục trong Mongodb.....	41
CHƯƠNG 4: KẾT LUẬN VÀ ĐỊNH HƯỚNG PHÁT TRIỂN	43
1. Kết Luận:	43
2. Một số định hướng của MongoDB gồm:.....	43
TÀI LIỆU THAM KHẢO.....	45
PHỤ LỤC.....	45

DANH MỤC HÌNH ẢNH

Hình 1. Dịch vụ MongoDB.....	12
Hình 2. Mô hình dữ liệu chuẩn hóa	15
Hình 3: Giao diện MongoDB.....	16
Hình 4: Các database cơ bản của MongoDB	17
Hình 5: Tạo database	17
Hình 6: Thêm dữ liệu	18
Hình 7: Truy xuất dữ liệu.....	18
Hình 8; Tải môi trường	19
Hình 9: Tạo kết nối	19
Hình 10: Tạo Playground.....	20
Hình 11: Tạo database	20
Hình 12: Tạo collection	21
Hình 13: Đặt tên cho collection	21
Hình 14: Insert dữ liệu	22
Hình 15: Chỉ định bao gồm một trường trong document	26
Hình 16: Loại trừ trường được chỉ định ra khỏi document.....	27
Hình 17: Ví dụ cho lệnh \$unwind()	31
Hình 18: Export collection.....	36
Hình 19: Các lựa chọn của Export.....	37
Hình 20: Lựa chọn trường để Export.....	38
Hình 21: Restore data.....	38

DANH MỤC BẢNG

Bảng 1. So sánh NoSQL và SQL.....	7
Bảng 2: Các đặc tả của \$project.....	26
Bảng 3: So sánh lệnh \$project	27
Bảng 4: So sánh lệnh \$match.....	28
Bảng 5: So sánh lệnh \$sort.....	29
Bảng 6: So sánh lệnh \$group	30
Bảng 7: Dữ liệu của 2 collection.....	33
Bảng 8; So sánh lệnh lookup	34

LỜI MỞ ĐẦU

1. Đặt vấn đề

Với sự phát triển mạnh mẽ của Internet hiện nay, cùng với lượng truy cập, tìm kiếm thông tin của người dùng mạng xã hội ngày càng nhiều. Đặc biệt, với sự phát triển mạnh mẽ của mạng xã hội, thì nhu cầu tìm kiếm thông tin và thêm nữa là người dùng có thể tự do sáng tạo nội dung, vì vậy mà ngày có nhiều lượng lớn thông tin cần, nhu cầu lưu trữ dữ liệu tăng lên. Vì vậy mà việc lưu trữ và khai thác lượng dữ liệu khổng lồ này là một thách thức lớn mà ta đang gặp phải.

Các hệ quản trị cơ sở dữ liệu quan hệ hiện tại chỉ đáp ứng được cho việc lưu trữ một lượng thông tin nhất không quá lớn, thêm vào đó là những hạn chế khi tổ chức dữ liệu. Vấn đề đặt ra là không chỉ đáp ứng việc lưu trữ mà còn phải có phương pháp quản lý, khai thác lượng thông tin hiệu quả. Do đó những năm gần đây, nhiều loại cơ sở dữ liệu phi quan hệ được nghiên cứu. Những CSDL này đặc biệt thích hợp cho các ứng dụng cực lớn, giảm thiểu tối đa các phép tính toán, tác vụ đọc-ghi với khả năng chịu tải, chịu lỗi cao nhưng đòi hỏi tài nguyên phần cứng khá thấp.

Nhóm quyết định chọn MongoDB để thực hiện tìm hiểu vì đây là hệ quản trị cơ sở dữ liệu phi quan hệ phổ biến hiện nay (top 4 theo đánh giá từ db-engines) và được dùng làm backend cho rất nhiều website như eBay, SourceForge, The New York Times,...

2. Mục tiêu

- Tìm hiểu về cách tổ chức, quản lý và truy vấn dữ liệu của 1 hệ quản trị cơ sở dữ liệu NoSQL – điển hình là MongoDB.
- Đánh giá cơ bản giữa hệ quản trị cơ sở dữ liệu quan hệ và phi quan hệ.

3. Phương pháp nghiên cứu

- Nhóm đã tạo 1 database Quản lý bán hàng với 20 dòng dữ liệu. Chúng em sẽ tiến hành tìm hiểu về MongoDB, sau đó sẽ chạy demo các tính năng của MongoDB

CHƯƠNG 1: GIỚI THIỆU VỀ MONGODB

1. So sánh giữa NoSQL và SQL

Bảng 1. So sánh NoSQL và SQL

Đặc điểm so sánh	NoSQL(Not Only SQL)	SQL(Structured Query Language)
Khái niệm	Được gọi là cơ sở dữ liệu phi quan hệ hoặc phân tán	Được gọi là RDBMS hoặc Cơ sở dữ liệu quan hệ
Lịch sử hình thành	Được phát triển vào cuối những năm 2000 để khắc phục các vấn đề và hạn chế của SQL databases	Được phát triển vào những năm 1970 để giải quyết các vấn đề với lưu trữ tệp phẳng
Lược đồ (Schema)	Sử dụng lược đồ động cho dữ liệu phi cấu trúc	Có lược đồ được xác định trước
Khả năng mở rộng	Có thể mở rộng theo chiều ngang	Có thể mở rộng theo chiều dọc
Các cơ sở dữ liệu	MongoDB, Redis, Neo4j, Cassandra, Hbase	Oracle, Postgres, MS-SQL

Open – source	Open-source	Sự kết hợp của mã nguồn mở như Postgres & MySQL, và thương mại như Oracle Database
Mô hình ACID và BASE	BASE (Về cơ bản có sẵn(Basically Available,), trạng thái mềm(Soft state), phù hợp cuối cùng (Eventual consistency) là một mô hình của nhiều hệ thống NoSQL	ACID (Atomicity, nhất quán(Consistent), cách ly(Isolation)và độ bền(Durability)) là một chuẩn cho RDBMS
Môi trường phù hợp	Không phù hợp với môi trường truy vấn phức tạp	Phù hợp với môi trường truy vấn phức tạp
Lưu trữ dữ liệu phân cấp	Thích hợp cho việc lưu trữ dữ liệu phân cấp vì có hỗ trợ phương thức cặp khóa-giá trị	Không thích hợp cho việc lưu trữ dữ liệu phân cấp
Dạng lưu trữ dữ liệu	Biểu diễn theo dạng tài liệu, cặp khóa – giá trị, đồ thị, trong bộ nhớ, tìm kiếm	Biểu diễn theo dạng bảng

2. Khái quát sơ lược về MongoDB

MongoDB là một hệ quản trị cơ sở dữ liệu phi quan hệ (NoSQL) mã nguồn mở phổ biến. Nó được phát hành lần đầu tiên vào năm 2009 và được thiết kế linh hoạt, có thể mở rộng và nhanh chóng. MongoDB lưu trữ dữ liệu ở định dạng tài liệu linh hoạt được gọi là BSON (JSON nhị phân), có thể lưu trữ dữ liệu ở nhiều định dạng khác nhau, bao gồm các cặp khóa-giá trị, mảng và tài liệu lồng nhau.

MongoDB không cần phải có lược đồ hoặc cấu trúc được xác định trước cho dữ liệu mà nó lưu trữ. Có nghĩa là ta không cần thay đổi mô hình cơ sở dữ liệu cơ bản mà vẫn có thể dễ dàng thêm, xóa các trường khỏi tài liệu khi cần. Ngoài ra, MongoDB cũng hỗ trợ các ngôn ngữ truy vấn phong phú và cung cấp các tính năng như lập chỉ mục, sao chép và phân đoạn để cho phép hiệu suất cao và khả năng mở rộng.

Các công ty như eBay, The New York Times và Expedia đã ứng dụng MongoDB vào trong việc phát triển các trang web hiện đại với hiệu suất cao và có khả năng mở rộng.

3. Ưu điểm và nhược điểm của MongoDB.

3.1. Ưu điểm của MongoDB:

- + Lưu trữ dưới dạng: MongoDB lưu trữ dữ liệu dưới dạng tài liệu JSON, cho phép lưu trữ dữ liệu có cấu trúc linh hoạt, phù hợp với các ứng dụng có yêu cầu thay đổi dữ liệu nhanh chóng và linh hoạt.
- + Khả năng mở rộng: khả năng mở rộng theo chiều ngang (horizontal scaling), cho phép mở rộng hệ thống dựa trên khả năng xử lý tương đối dễ dàng, giúp tăng hiệu suất và khả năng chịu tải của hệ thống.
- + Phù hợp cho dữ liệu lớn: MongoDB được thiết kế nhằm xử lý các tập dữ liệu lớn (big data), có khả năng chịu tải cao và hiệu suất tốt đối với ứng dụng có dữ liệu lớn và phức tạp.

+ Tính năng linh hoạt: MongoDB cung cấp nhiều tính năng linh hoạt như khả năng truy vấn phong phú, hỗ trợ đa định dạng dữ liệu, hỗ trợ replica set và sharding cho khả năng sao lưu và khôi phục dữ liệu đáng tin cậy.

+ Cộng đồng đông đảo: MongoDB có một cộng đồng người dùng và phát triển đông đảo, cung cấp nhiều tài liệu, tài nguyên và hỗ trợ từ cộng đồng.

3.2. *Nhược điểm của MongoDB:*

Không hỗ trợ giao thức quan hệ: Vì là hệ quản trị cơ sở dữ liệu NoSQL nên MongoDB không hỗ trợ giao thức quan hệ như SQL; không có tính năng join, transaction, foreign key, dẫn đến truy vấn dữ liệu phức tạp hoặc phân tích dữ liệu đòi hỏi công việc phức tạp hơn so với các hệ quản trị cơ sở dữ liệu quan hệ.

+ Yêu cầu quản trị dữ liệu chặt chẽ hơn: MongoDB không có tính năng transaction, làm cho việc quản lý dữ liệu và đồng bộ dữ liệu càng phức tạp hơn so với các hệ quản trị cơ sở dữ liệu quan hệ.

+ Dễ xảy ra duplicate data: MongoDB không hỗ trợ ràng buộc duy nhất (unique constraint) trên trường dữ liệu, dẫn đến khả năng xảy ra duplicate data (dữ liệu trùng lặp) nếu không được kiểm soát cẩn thận.

+ Không phải lựa chọn tốt cho các ứng dụng có yêu cầu ACID: MongoDB không hỗ trợ ACID (Atomicity, Consistency, Isolation, Durability), làm cho việc đảm bảo tính nhất quán và độ tin cậy của dữ liệu trong các ứng dụng có yêu cầu ACID trở nên khó khăn hơn so với các hệ quản trị cơ sở dữ liệu quan hệ.

➔ Vậy những trường hợp cần sử dụng NoSQL: khi không cần hỗ trợ ACID, khi dữ liệu quá lớn, khi dữ liệu đưa vào cần có lược đồ linh hoạt, các ràng buộc không cần phải bắt buộc thực hiện trong cơ sở dữ liệu, được sử dụng để lưu trữ dữ liệu tạm thời như giỏ mua hàng

4. Cách cài đặt và sử dụng MongoDB:

Bước 1: Tải MongoDB

- Tải xuống trình cài đặt Cộng đồng MongoDB từ liên kết sau: [.msi](#)

► Trung tâm tải xuống MongoDB ([MongoDB Community Server Download](#))

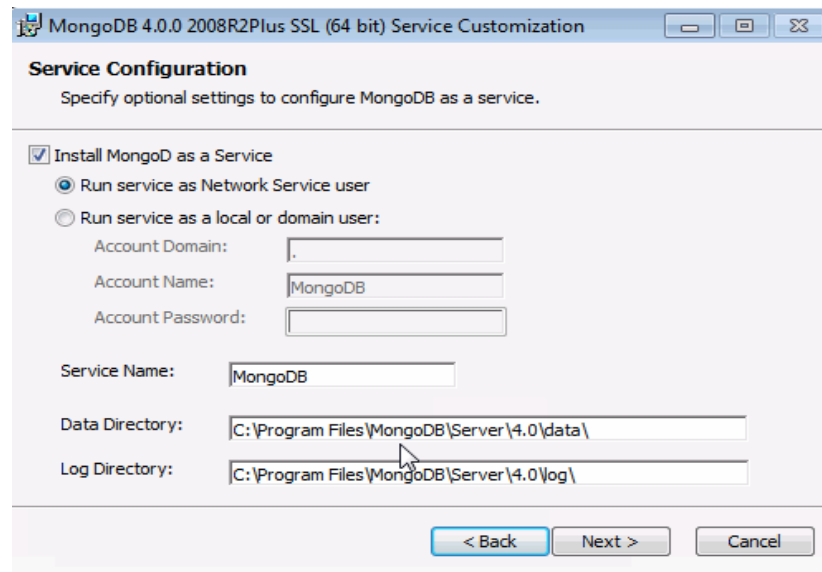
- Trong menu thả xuống Phiên bản , hãy chọn phiên bản MongoDB để tải xuống.
- Trong danh sách thả xuống Nền tảng , hãy chọn Windows .
- Trong menu thả xuống Gói , chọn msi .
- Nhấp vào Tải xuống .

Bước 2: Chạy trình cài đặt MongoDB.

- Chuyển đến thư mục mà bạn đã tải xuống trình cài đặt MongoDB (.msi tệp). Theo mặc định, đây là Downloads thư mục của bạn.
- Bấm đúp vào .msi tệp.

Bước 3: Làm theo hướng dẫn cài đặt MongoDB Community Edition.

- Trình hướng dẫn sẽ hướng dẫn bạn cài đặt MongoDB và MongoDB Compass.
- Chọn Loại thiết lập: tại đây bạn có thể chọn thiết lập Hoàn chỉnh (mọi cài đặt và các tool MongoDB được cài vào vị trí mặc định) hoặc có thể chọn thiết lập tùy chỉnh (ta có thể chỉ định tệp thực thi được cài đặt và ở đâu)
- Bắt đầu từ MongoDB 4.0, bạn có thể thiết lập MongoDB như một dịch vụ Windows trong quá trình cài đặt hoặc chỉ cài đặt các tệp nhị phân.
- Dịch vụ MongoDB



Hình 1. Dịch vụ MongoDB

- Chọn Cài đặt dịch vụ MongoDB
- Chọn một trong hai:
 - Chạy dịch vụ với tư cách là người dùng Dịch vụ mạng (Mặc định). Tại đây là tài khoản người dùng Windows được tích hợp sẵn trong Windows.
 - Chạy dịch vụ với tư cách là người dùng cục bộ hoặc miền.
 - + Đối với tài khoản người dùng cục bộ hiện tại, hãy chỉ định một khoảng thời gian cho Miền tài khoản và chỉ định Tên tài khoản và Mật khẩu tài khoản cho người dùng.
 - + Đối với người dùng miền hiện tại, hãy chỉ định Miền tài khoản, Tên tài khoản và Mật khẩu tài khoản cho người dùng đó.
- Tên dịch vụ. Chỉ định tên dịch vụ. Tên mặc định là MongoDB. Nếu bạn đã có một dịch vụ với tên được chỉ định, bạn phải chọn một tên khác.

- Thư mục dữ liệu . Chỉ định thư mục dữ liệu, tương ứng với tệp `--dbpath`. Nếu thư mục không tồn tại, trình cài đặt sẽ tạo thư mục và đặt quyền truy cập thư mục cho người dùng dịch vụ.
- Thư mục nhật ký . Chỉ định thư mục Nhật ký, tương ứng với tệp `--logpath`. Nếu thư mục không tồn tại, trình cài đặt sẽ tạo thư mục và đặt quyền truy cập thư mục cho người dùng dịch vụ.
- Cài đặt La bàn MongoDB: Click *Tùy chọn* để có được hướng dẫn cài đặt [MongoDB La bàn](#) → chọn Cài đặt MongoDB Compass (Mặc định).
- Khi đã sẵn sàng, hãy nhấp vào Cài đặt .

CHƯƠNG 2: MÔ HÌNH DỮ LIỆU TRONG MONGODB

- MongoDB cung cấp 2 kiểu dữ liệu mô hình : **Mô hình dữ liệu nhúng và mô hình dữ liệu chuẩn hoá**. Tùy vào trường hợp, bạn có thể sử dụng model phù hợp trong quá trình chuẩn bị document

1. Mô hình dữ liệu nhúng

- Trong MongoDB, ta có thể nhúng dữ liệu liên quan vào một cấu trúc hoặc tài liệu. Document cho phép nhúng các cấu trúc tài liệu vào một *trường* hoặc *mảng* trong tài liệu → *mô hình dữ liệu chưa chuẩn hóa*
- **Ví dụ:** Lấy thông tin chi tiết của 1 sinh viên từ 3 document khác nhau: *Info_details*, *Contact*, *Address*. Từ 3 document này ta có thể đem đi nhúng trong một tài liệu

```
{  
  
  _id: ObjectId("64494fabe1dcb1b3c9edd42b"),  
  
  Info_details: {name: "Ngan", school: "UIT"},  
  
  Contact: {email: "21521174@gm.uit.edu.vn", phone:"05229959"},  
  
  Address:{city: "Da Nang", street: "Duy Tan"}  
  
}
```

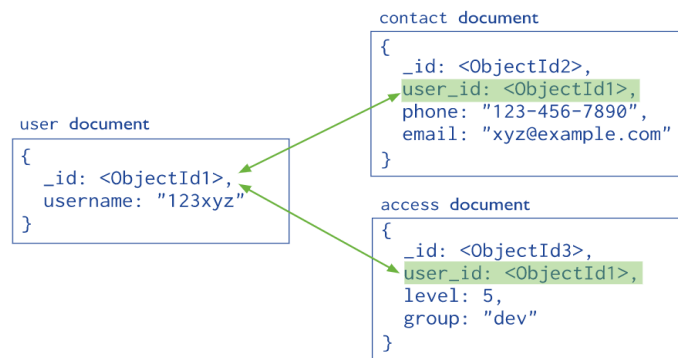
- Mô hình này cho phép lưu trữ các mẫu tin liên quan trong cùng một bản ghi. Vì vậy, các ứng dụng có thể đưa ra ít truy vấn và cập nhật hơn để hoàn thành hoạt động thông thường. Thiết kế dữ liệu theo kiểu này, chúng ta không cần phải tạo ra quá nhiều collection. Có thể gộp nhiều collection vào thành 1.

2. Mô hình dữ liệu chuẩn hóa

- Mô hình này sử dụng các tham chiếu giữa các tài liệu để mô tả các mối quan hệ giữa các tài liệu - được kết nối với nhau bằng các tham chiếu.

nối → Mô
chuẩn

Ví dụ:



Hình 2. Mô hình dữ liệu chuẩn hóa

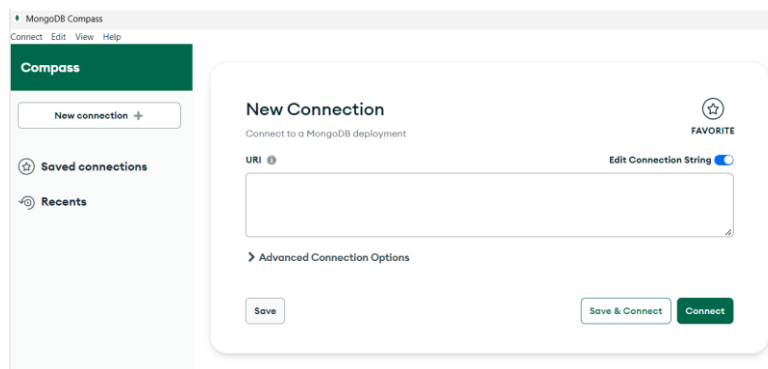
CHƯƠNG 3: CÁC THAO TÁC CỦA MONGODB

1. Các thao tác cơ bản

1.1. Sử dụng giao diện màn hình

Bước 1: Kết nối với máy chủ:

- Ở giao diện của MongoDB Compass, ta gõ tên máy chủ của ta tại mục URI để thiết lập kết nối. (Máy chủ mặc định thường có địa chỉ là localhost:27017)

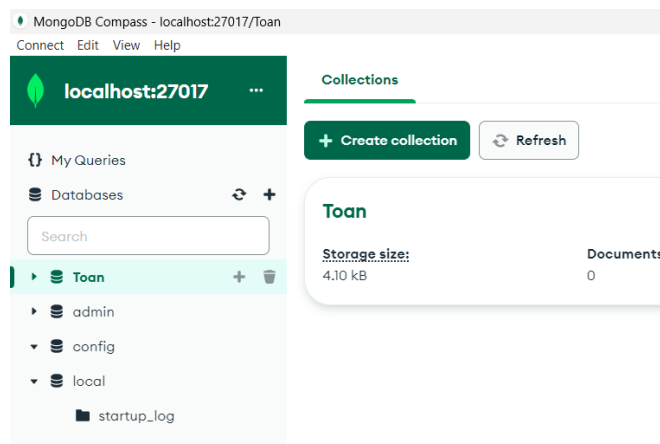


Hình 3: Giao diện MongoDB

- Trong MongoDB, có 3 cơ sở dữ liệu mặc định được tạo tự động khi khởi động máy chủ cơ sở dữ liệu lần đầu tiên: admin, config và local (hình 4).
 - + admin: Cơ sở dữ liệu quản trị là cơ sở dữ liệu mặc định cho các tác vụ quản trị như quản lý người dùng và xác thực. Nó lưu trữ thông tin tài khoản người dùng. Mặc định, chỉ người dùng có quyền quản trị mới có thể truy cập cơ sở dữ liệu quản trị.
 - + config: Cơ sở dữ liệu cấu hình được sử dụng để lưu trữ siêu dữ liệu và thông tin cấu hình cho các cụm phân đoạn. Nó chứa thông tin về các phân đoạn, chẳng hạn như vị trí và bộ bản sao của chúng, cũng như thông tin về các khối dữ liệu được lưu trữ trên các phân đoạn.
 - + local: Cơ sở dữ liệu cục bộ được MongoDB sử dụng cho mục đích nội bộ. Nó lưu trữ thông tin về phiên bản hiện tại của máy chủ

MongoDB, chẳng hạn như trạng thái sao chép và trạng thái của oplog.

- Cần lưu ý là không nên sửa đổi trực tiếp các cơ sở dữ liệu này, vì làm như vậy có thể gây ra sự cố với hoạt động bình thường của máy chủ MongoDB. Thay vào đó, bạn nên sử dụng các lệnh và công cụ quản trị thích hợp do MongoDB cung cấp để quản lý cơ sở dữ liệu và bộ sưu tập.



Hình 4: Các database cơ bản của MongoDB

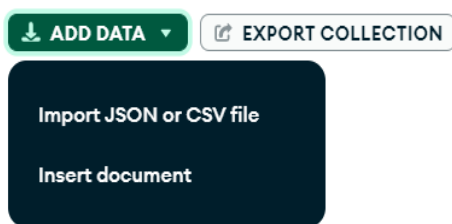
Bước 2: Cách tạo 1 Database:

- + Ở giao diện hình 4 tại mục Databases ta chọn vào mục dấu cộng, MongoDB sẽ hiện ra một cửa sổ (hình 5).

The image shows a 'Create Database' dialog box. It has a title bar with a close button. Inside, there are two input fields: 'Database Name' and 'Collection Name'. Below these is a checkbox labeled 'Time-Series' with a description: 'Time-series collections efficiently store sequences of measurements over a period of time. [Learn More](#)'. At the bottom, there is a section for 'Additional preferences' with a right-pointing arrow and the text '(e.g. Custom collation, Capped, Clustered collections)'. At the very bottom are two buttons: 'Cancel' and 'Create Database'.

Hình 5: Tạo database

- + Sau đó chúng ta đặt tên cho Database và Collection.
 - Database: Một cơ sở dữ liệu dùng để lưu dữ liệu từ các collection.
 - Collection: Một bộ sưu tập là một nhóm các tài liệu MongoDB. Các tài liệu trong một bộ sưu tập có thể có các trường khác nhau. Một bộ sưu tập tương đương với một bảng trong hệ thống cơ sở dữ liệu quan hệ. Một bộ sưu tập tồn tại trong một cơ sở dữ liệu duy nhất.
- + Ở giao diện người dùng chúng ta có thể thêm dữ liệu của một thư mục bằng cách:
 - Bước 1: chọn vào ADD DATA
 - Bước 2: Ở đây chúng ta có 2 lựa chọn (hình 6) :

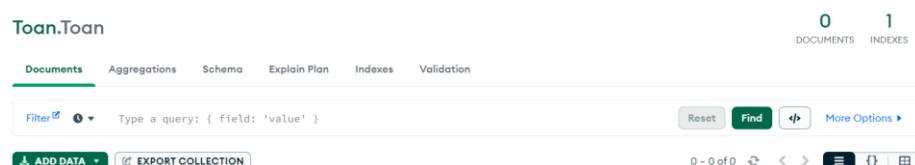


-Thêm từ file json (add file json)

-Thêm bằng tay (add documents)

Hình 6: Thêm dữ liệu

- + Để truy xuất dữ liệu chúng ta có thể dùng thanh tìm kiếm trong documents (hình 7).



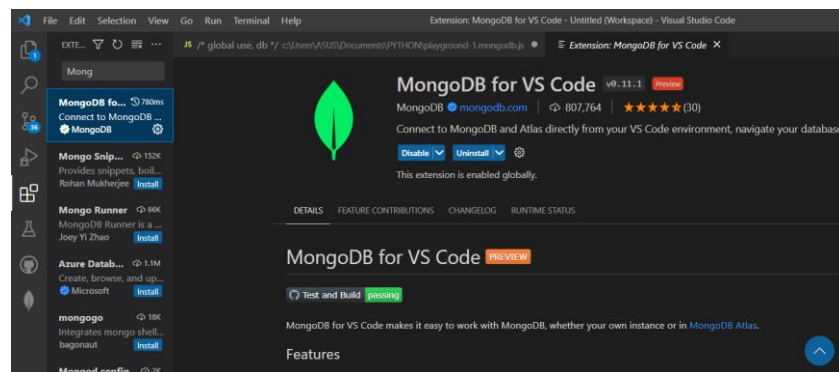
Hình 7: Truy xuất dữ liệu

- Tại thanh Filter (hình 7) chúng ta gõ trường dữ liệu và giá trị để tìm ra giá trị mà mình muốn.

1.2. Sử dụng bằng vscode:

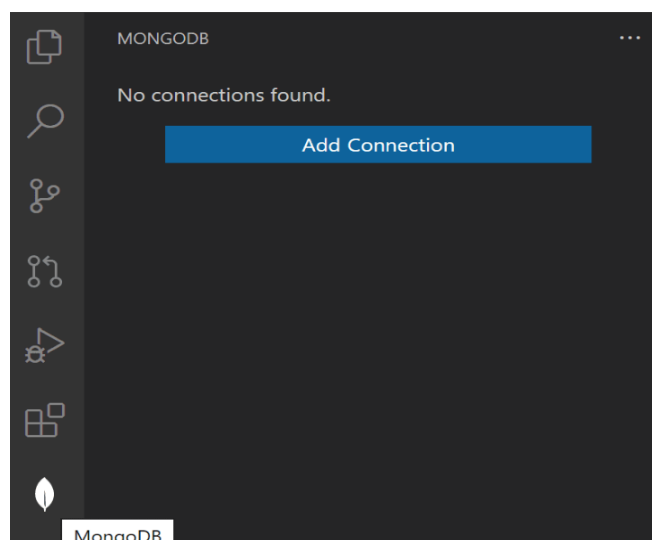
Tải môi trường:

- + Chúng ta vào **vscode** tìm kiếm và tải MONGODB for VS CODE (hình)



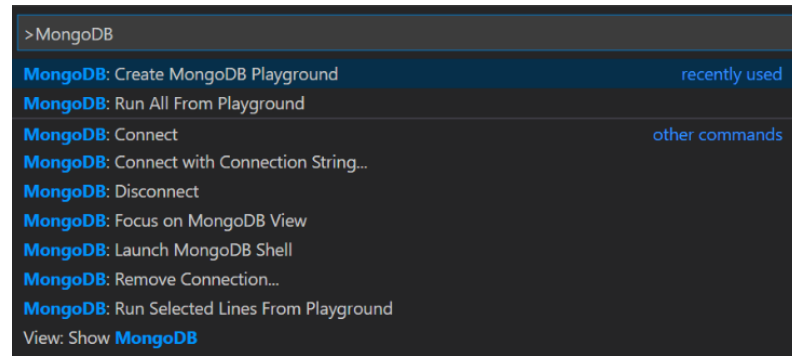
Hình 8: Tải môi trường

- + Sau khi tải về xong chúng ta tạo kết nối với máy chủ của chúng ta (hình 9).



Hình 9: Tạo kết nối

- + Sau đó chúng ta tạo Playground cho vscode (hình 10).

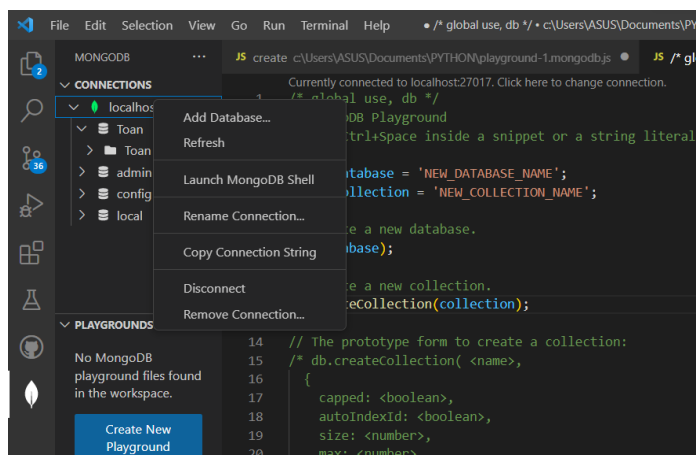


Hình 10: Tạo Playground

2. Các câu lệnh cơ bản

2.1. Tạo database

- Bấm chuột phải vào localhost phía tay trái và chọn new Database (hình 11)

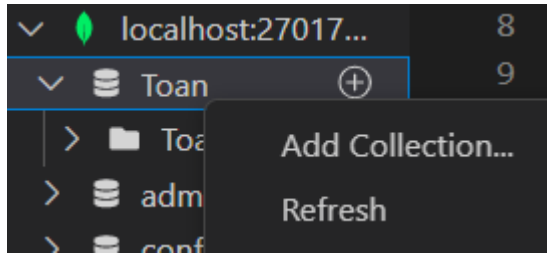


Hình 11: Tạo database

- Tại dòng const database, ta thay 'NEW_DATABASE_NAME' bằng tên database muốn tạo.
- Bôi đen các dòng lệnh và nhấn nút chạy.

2.2. Tạo collection

- Bấm chuột phải vào database mà muốn tạo collection → chọn add collection (hình 12).



Hình 12: Tạo collection

- Tại dòng const collection, ta thay “NEW_COLLECTION_NAME” bằng tên collection mà ta muốn đặt (hình 13).

```
const database = 'Toan';
const collection = 'NEW_COLLECTION_NAME';

// The current database to use.
use(database);

// Create a new collection.
db.createCollection(collection);
```

Hình 13: Đặt tên cho collection

- Bôi đen các dòng lệnh rồi nhấn nút chạy.

2.3. Thêm dữ liệu

- Sử dụng lệnh **use('Name')**, trong đó Name là tên Database mà chúng ta muốn sử dụng.
- **Cú pháp:** `db.getcollection('Collection_name').insertMany(['Field' : 'Data'])` (hình 14)

+ Collection_name: tên collection muốn insert dữ liệu

```

use('Toan');

// Insert a few documents into the sales collection.
db.getCollection('Toan').insertMany([
  { 'item': 'abc', 'price': 10, 'quantity': 2, 'date': new Date('2014-03-01T08:00:00Z') },
  { 'item': 'jkl', 'price': 20, 'quantity': 1, 'date': new Date('2014-03-01T09:00:00Z') },
  { 'item': 'xyz', 'price': 5, 'quantity': 10, 'date': new Date('2014-03-15T09:00:00Z') },
  { 'item': 'xyz', 'price': 5, 'quantity': 20, 'date': new Date('2014-04-04T11:21:39.736Z') },
  { 'item': 'abc', 'price': 10, 'quantity': 10, 'date': new Date('2014-04-04T21:23:13.331Z') },
  { 'item': 'def', 'price': 7.5, 'quantity': 5, 'date': new Date('2015-06-04T05:08:13Z') },
  { 'item': 'def', 'price': 7.5, 'quantity': 10, 'date': new Date('2015-09-10T08:43:00Z') },
  { 'item': 'abc', 'price': 10, 'quantity': 5, 'date': new Date('2016-02-06T20:20:13Z') },
]);

```

Hình 14: Insert dữ liệu

2.4. Tìm kiếm dữ liệu

- **Cú pháp:** `db.getCollection.find()`
- Một số kí hiệu so sánh khi sử dụng tìm kiếm:
 - + `$lt`: bé hơn
 - + `$gt`: lớn hơn
 - + `$lte`: bé hơn hoặc bằng
 - + `$gte`: lớn hơn hoặc bằng

Ví dụ: Tìm dữ liệu có ngày thêm vào là 04/04/2014

```

+ db.getCollection('Toan').find({
  date: {
    $gte: new Date('2014-04-04T00:00:00Z'),
    $lt: new Date('2014-04-05T00:00:00Z')
  }
});

```

2.5. Xóa 1 database

- **Cú pháp:** `db.dropDatabase('DATABASE_NAME')`

+ DATABASE_NAME: TÊN của database muốn xóa

- **Ví dụ:** Xóa cơ sở dữ liệu tên “Toan”

```
db.dropDatabase('Toan')
```

2.6. Xóa 1 collection

- **Cú pháp:** db. COLLECTION_NAME.drop()

+ COLLECTION_NAME: tên collection muốn xóa

- **Ví dụ:** Xóa Collection tên “Toan”

```
db.Toan.drop()
```

2.7. Xóa dữ liệu trong collection

- **Cú pháp:** db. Collection.deleteOne hoặc

```
db.collection.deleteMany(
```

```
Query,
```

```
{
```

```
  JustOne: <boolean>,
```

```
  writeConcern: <document>,
```

```
  collation: <document>
```

```
})
```

+ collectionName: tên của collection muốn thực thi lệnh xóa document

+ query: là object (hay document) chứa các câu truy vấn để lọc dữ liệu

+ justOne: là tham số cấu hình số lượng bản ghi có thể xóa khi query thực thi khớp

- justOne: true thì nó sẽ chỉ xóa 1 bản ghi duy nhất
 - justOne: false thì sẽ xóa tất cả các bản ghi khớp với điều kiện query
- + writeConcern: là 1 document chứa write concern.
- + collation: là 1 document chứa các quy tắc
- **Ví dụ:** Xóa một admin có name = “Toidicode” và có age = 18.

```
db.admin.remove(
{
name: “toidicode”,
age: 18 })
```

2.8. Update dữ liệu

- Để sửa đổi 1 bản ghi duy nhất trong MongoDB thì sử dụng phương thức updateOne() theo cú pháp sau:

```
db.collection.updateOne(
filter,
update,
{
Upsert: <Boolean>,
writeConcern: <document>,
collation: <document>
})
```

+ filter là một object chứa các tiêu chí lựa chọn bản ghi update (sử dụng cú pháp selector).

+ update là object chứa dữ liệu sửa đổi trên bản ghi.

+ upsert là một boolean cấu hình điều gì sẽ xảy ra khi không có bản ghi khớp với filter. Nếu upsert = true thì nó sẽ thêm mới bản ghi đó nếu không có bản ghi 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 dữ liệu khớp với filter nhiều hơn 1 bản ghi thì nó sẽ chỉ sửa đổi cho 1 bản ghi đầu tiên

- **Ví dụ:** Sửa name của admin có age = 18 thành Toan.

```
db.admin.updateOne(
  {age: 18},
  {
    $set:{
      name: 'Toan'
    }
  }
)
```

2.9. Toán tử \$project

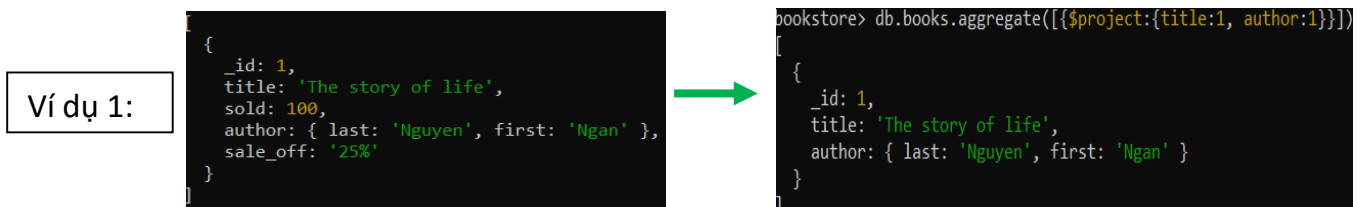
- **Cú pháp:** { **\$project:** { <các trường thông tin> } }
- **Tác dụng:** + Xác định trường mong muốn thực hiện

+ Kết quả khi sử dụng \$project sẽ tương tự như SELECT trong truy vấn SQL

- Các \$project đặc tả có các dạng sau:

Bảng 2: Các đặc tả của \$project

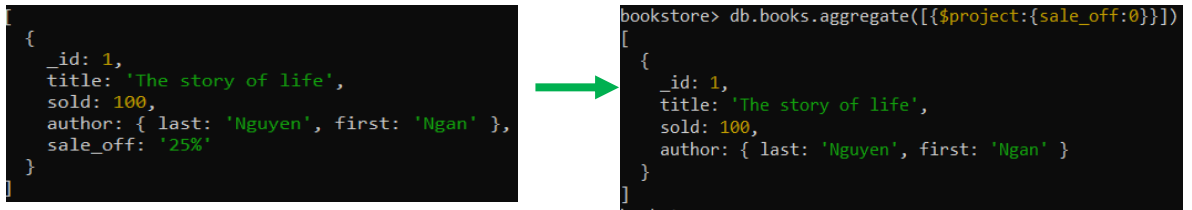
Hình thức	Ý nghĩa
<field X>: <1 or true>	Field X sẽ xuất hiện trong output document
_id: <0 or false>	Field _id sẽ không xuất hiện trong output document
<field X>: <expression>	Thêm trường mới hoặc đặt lại giá trị của trường hiện có



Hình 15: Chỉ định bao gồm một trường

Ở ví dụ 1, thực hiện truy vấn lấy title và author nên ta đặt là 1. Còn _id trong document là 1 nên kết quả trả về là _id, title, author

Ví dụ 2:



```
{
  _id: 1,
  title: 'The story of life',
  sold: 100,
  author: { last: 'Nguyen', first: 'Ngan' },
  sale_off: '25%'
}
```

bookstore> db.books.aggregate([{\$project:{sale_off:0}}])

```
[
  {
    _id: 1,
    title: 'The story of life',
    sold: 100,
    author: { last: 'Nguyen', first: 'Ngan' }
  }
]
```

Hình 16: Loại trừ trường được chỉ định ra khỏi document

Ở ví dụ 2, thực hiện truy vấn loại bỏ trường 'sale_off'

Bảng 3: So sánh lệnh \$project

MongoDB	SQL
db.books.aggregate([{\$project:{title:1, author:1}}])	SELECT _id, title, author FROM books

2.10. Toán tử \$match

Chức năng:

- Dùng để lọc các document theo một điều kiện nào đó
- Dùng tương đương với mệnh đề **WHERE/HAVING** trong câu lệnh truy vấn SQL
- \$match dùng giống như cú pháp **find()**

Cú pháp: \$match { \$match: { <điều kiện> } }

Ví dụ:

Bảng 4: So sánh lệnh \$match

MongoDB	SQL	Giải thích
<pre>db.books.aggregate([\$match: { title: 'The story of life' }])</pre>	<pre>SELECT * FROM books WHERE title = 'The story of life'</pre>	<p>Hiện ra thông tin của quyển sách có title là The story of life</p>

2.11. Toán tử \$sort

Chức năng: - Dùng để sắp xếp các document trong output theo điều kiện

- \$sort tương tự như ORDER BY trong SQL

Cú pháp: { \$sort: {<trường 1>:<thứ tự>, <trường 2>:<thứ tự>,... } }

Trong đó, <thứ tự> có thể có các giá trị sau:

1: sắp xếp theo thứ tự tăng dần

-1: sắp xếp theo thứ tự giảm dần

Ví dụ so sánh với truy vấn SQL

Bảng 5: So sánh lệnh \$sort

MongoDB	SQL	Giải thích
db.books.aggregate([<div>\$sort: {title: -1}</div> <div>]])</div>	SELECT * FROM books ORDER BY SL DESC	Hiện ra các quyển sách có tên tựa đề theo thứ tự giảm dần

2.12. Toán tử \$group

Cú pháp: \$group:{_id: <expression>,

<field>: {<accumulator> : <expression>}}

Chức năng:

- Dùng để gom nhóm các document đầu vào theo expression _id. Mỗi nhóm tương ứng với 1 document output
- \$group tương đương với mệnh đề GROUP BY trong SQL

Nếu **_id : null**, thì sẽ truy vấn tất cả các document đầu vào

Trong đó:

+ <expression>: tập hợp các nhóm thông tin cần thực hiện nhóm để thực hiện thao tác nào đó

+ <new_field>: đặt tên cho trường mới để hiển thị kết quả truy vấn (trong SQL thì: SELECT <field> AS <new_field>). Tại trường mới này, ta có thể thực hiện các lệnh tính toán: \$sum, \$avg, \$count, \$max, \$min, \$addToSet, \$first, \$last, \$push

Ví dụ:

Bảng 6: So sánh lệnh \$group

MongoDB	SQL	Giải thích
<pre>db.products.aggregate([{\$group:{_id: "\$category",total: { \$sum:"\$price"}}}])</pre>	<pre>SELECT category, SUM(price) AS total FROM products GROUP BY category</pre>	Hiện ra tổng giá của các loại hàng được nhóm theo loại sản phẩm

2.13. Toán tử \$unwind

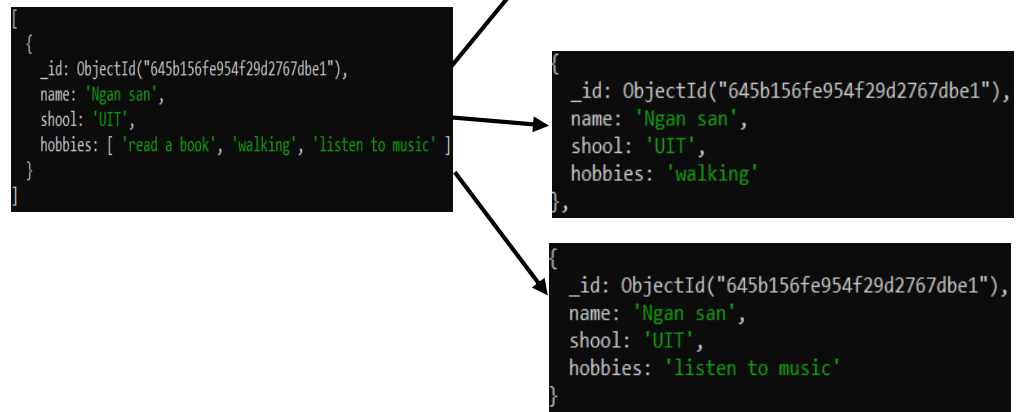
Chức năng: - Dùng để phân tách giá trị của một mảng các trường thông tin trong các input document. Nếu mảng các trường thông tin của một input document có N phần tử thì trong output sẽ có N document

Cú pháp: {\$unwind:<field_name>}

Ví dụ:

Phân tách đưa mảng các hobbies

thành 3 document khác nhau



Hình 17: Ví dụ cho lệnh \$unwind()

2.14. Toán tử \$lookup

Chức năng: - Cho phép thực hiện phép kết trái giữa 2 collection trong cùng database. Với mỗi input document, \$lookup sẽ thêm một array field chứa các phần tử matching với collection được join.

- Toán tử này tương đương với mệnh đề JOIN trong SQL.

Cú pháp:

```
{  
  
  $lookup:  
  
    {  
  
      from: <collection để kết>,  
  
      localField: <tên trường (của collection hiện tại) cần kết>  
  
      foreignField: <field from the documents of the "from" collection>,  
  
      as: <output array field>  
  
    }  
  
}
```

Ví dụ: Cho 2 collection SinhVien và DiemThi như dưới:

Bảng 7: Dữ liệu của 2 collection

SinhVien	DiemThi
<pre>{ "_id": ObjectId("629d4dbc9bd84f1054fee4fb"), "MSSV" : 21521174, "Ten": "Nguyen Ngan", "Lop": "CNCL2021.1", "SoDT": "0896265999", "Dia chi": 123 Tran Phu }</pre>	<pre>{ "_id": ObjectId("629d5e219bd84f1054fee502"), "MSSV" : 21521174, "Ten": "Nguyen Ngan", "Mon hoc": "Toan", "Diem": 8, "Lop": "CNCL2021.1" }</pre>

Bảng 8; So sánh lệnh lookup

MongoDB	SQL	Giải thích
<pre>db.SinhVien.aggregate([{ "\$lookup": { "from": "DiemThi", "localField": "MSSV", "foreignField": "MSSV", "as": "SV_DiemThi" } }]</pre>	<pre>SELECT * FROM SinhVien, DiemThi WHERE SinhVien. MSSV = DiemThi.MSSV</pre>	Thực hiện phép kết giữa 2 collection SinhVien và DiemThi thông qua thuộc tính MSSV

Kết quả sau khi thực hiện phép nối trên:

```
{
  "_id": ObjectId("629d4dbc9bd84f1054fee4fb"),
  "MSSV": 21521174,
  "Ten": "Nguyen Ngan",
  "Lop": "CNCL2021.1",
  "SoDT": "0896265999",
  "Dia chi": "123 Tran Phu",
  "SV_DiemThi": [
    {
```

```

        ObjectId("629d5e219bd84f1054fee502"),

        "MSSV" : 21521174,

        "Ten": "Nguyen Ngan",

        "Mon hoc": "Toan",

        "Diem": 8,

        "Lop": "CNCL2021.1"

    }

]

}

```

3. Các thao tác nâng cao trong MongoDB

3.1. Cú pháp tạo user:

```

db.createUser

({

    user: "username",          // user: xác định tên người dùng

    pwd: "password",          // pwd: xác định mật khẩu của người dùng

    roles: [{role: "role_name", db: "db_name"}]

    //role: xác định vai trò của người dùng khi đăng nhập vào tài khoản

})

```

- **Ví dụ:** Ở đây mình phân quyền cho người dùng “Toan” là userAdmin, người dùng này sẽ quản lý cơ sở dữ liệu duy nhất là “admin”

```

db.createUser({

```

```
user: "Toan", pwd: "123456",  
  
roles:[{ role: "root", db: "admin"}]  
  
})
```

3.2. Xóa một user

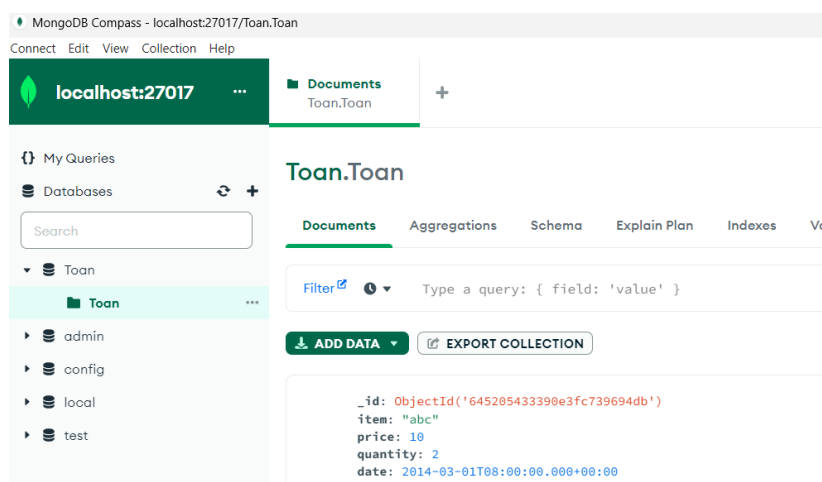
- Để xóa một user, ta sử dụng method `dropUser()`
- Ví dụ: xóa user tên tài khoản là "Toan"

use admin

```
db.dropUser('Toan')
```

3.3. Backup data

- Để export data của một collection thành 1 file chúng ta có thể dùng cách sau đây:
 - + Đầu tiên chúng ta vào app MongoDB và chọn collection mà mình muốn export.
 - + Tại giao diện chúng ta sẽ thấy nút export (hình 18).



Hình 18: Export collection

- + Sau đó chúng ta click vào nút “EXPORT COLLECTION”.
- + Sẽ có 2 lựa chọn là “Export Full Collection” và “Export query with filters” (hình 19).

Export

Collection Toan.Toan

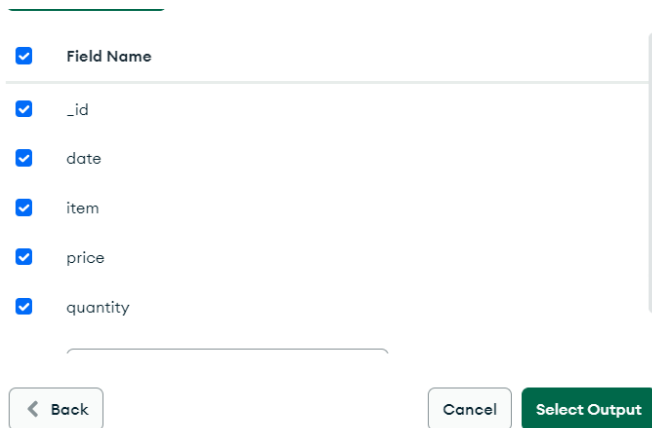
☒ Export query with filters — 8 results (Recommended)

```
db.Toan.find(
  {}
)
```

☐ Export Full Collection

Hình 19: Các lựa chọn của Export

- + Export query with filters dùng để export data qua hàm tìm kiếm mà người dùng nhập vào giúp người dùng có thể export data trong trường hợp chỉ cần một số data nhất định.
 - Sau khi chúng ta đã viết hàm tìm kiếm của mình, chúng ta có thể chọn trường mà chúng ta muốn lấy (hình 20).
- + Export Full Collection dùng để export toàn bộ data của collection.
 - Sau khi ấn vào Export Full Collection chúng ta sẽ được lựa chọn trường mà chúng ta muốn export (Hình 20).

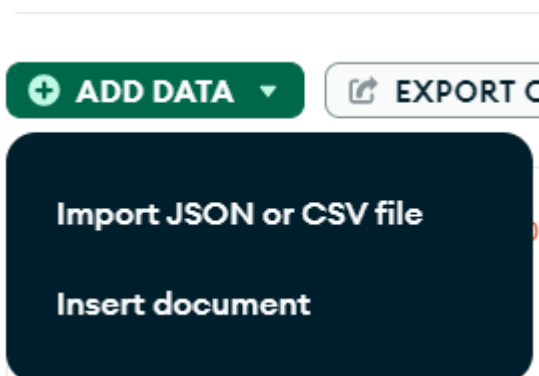


Hình 20: Lựa chọn trường để Export

- Sau khi chọn chúng ta chọn “Select Output” và chúng ta nhấn vào “select a file” để chọn file hoặc đặt tên file mà chúng ta muốn lưu data vào.

3.4. Restore data

- Để restore bằng 1 file có sẵn thì chúng ta click vào “ADD DATA” tại giao diện màn hình (hình 21).
- Sau đó chúng ta chọn import file JSON or CSV file (hình 21).



Hình 21: Restore data

- Sau khi nhấn vào thì chúng ta chọn file mà chúng ta đã backup để restore.

3.5. Embeded Relationship

- Chúng ta có thể nhúng address document vào trong user document.

```
+ VD:

{

  "_id":ObjectId("52ffc33cd85242f436000001"),

  "contact": "987654321",

  "dob": "01-01-1991",

  "name": "Tom Benzamin",

  "address": [

    {

      "building": "22 A, Indiana Apt",

      "pincode": 123456,

      "city": "Los Angeles",

      "state": "California"

    },

    {

      "building": "170 A, Acropolis Apt",

      "pincode": 456789,

      "city": "Chicago",

      "state": "Illinois"

    }

  ]

}
```


- Điều này giúp hạn chế trùng lặp dữ liệu khi một document đơn có thể hoàn toàn lưu trữ nhiều dữ liệu liên quan và qua đó nó sẽ giúp cho chúng ta trong việc tránh trùng lặp dữ liệu, truy xuất dễ dàng hơn. Tuy nhiên chúng ta không nên lưu quá nhiều vì khi đó sẽ ảnh hưởng đến tốc độ truy xuất của document.

3.6. Sử dụng DBRefs

- Khi chúng ta tìm kiếm một dữ liệu nào đó thì DBRefs sẽ giúp chúng ta xác định collection đang tham chiếu là địa chỉ id đang tham chiếu.
- Cú pháp khai báo: chúng ta thêm \$ vào trước biến chứa dữ liệu.
- Cú pháp tìm kiếm: Sau khi có 1 câu lệnh tìm kiếm.

VD: `var user = db.users.findOne({"name":"Tom Benzamin"})`

- chúng ta sử dụng thêm 1 biến để lưu address bằng cú pháp:

Tên biến = “Tên database”.address

- Cuối cùng để hiện thị những gì mình đã lưu bằng:

`db[“Tên Biến”].$[“Tên biến lưu collection”].findOne({"_id":(“Tên biến”. “Tên biến lưu id”)}))`

- VD: {

`"_id":ObjectId("53402597d852426020000002"),`

`"address": {`

`"$ref": "address_home",`

`"$id": ObjectId("534009e4d852427820000002"),`

`"$db": "tutorialspoint"},`

`"contact": "987654321",`

```
"dob": "01-01-1991",  
  
"name": "Tom Benzamin"  
  
}
```

Khi chúng ta tìm kiếm màn hình sẽ hiện thị:

```
>var user = db.users.findOne({"name":"Tom Benzamin"})  
  
>var dbRef = user.address  
  
>db[dbRef.$ref].findOne({"_id":(dbRef.$id)})
```

Code trên trả về address document sau đây, mà có mặt trong address_home collection:

```
{  
  
  "_id" : ObjectId("534009e4d852427820000002"),  
  
  "building" : "22 A, Indiana Apt",  
  
  "pincode" : 123456,  
  
  "city" : "Los Angeles",  
  
  "state" : "California"  
  
}
```

3.7. Chỉ Mục trong MongoDB

- Tạo chỉ mục (index) cho các trường dữ liệu mà bạn muốn truy vấn. Chỉ mục giúp MongoDB tìm kiếm dữ liệu nhanh hơn bằng cách sắp xếp và đánh chỉ mục các giá trị của trường đó.

- Xác định các trường dữ liệu mà bạn muốn trả về trong kết quả truy vấn. Đảm bảo rằng các trường này đã được định nghĩa trong chỉ mục hoặc là một phần của chỉ mục được sử dụng.
- Sử dụng truy vấn có điều kiện (query) và chỉ rõ các trường dữ liệu trong projection để chỉ trả về các trường mong muốn.
- VD: `db.collection.find({ "field1": value }, { "field2": 1, "field3": 1 })`

Trong đó, "field1" là trường dùng để điều kiện, và "field2" và "field3" là các trường muốn trả về.

- Lưu ý: Chỉ mục chỉ hoạt động tốt nhất trong vùng nhớ đệm nên khi tạo chúng ta nên tạo chỉ mục đảm bảo nhỏ hơn vùng nhớ đệm của Ram dành cho MongoDB.

CHƯƠNG 4: KẾT LUẬN VÀ ĐỊNH HƯỚNG PHÁT TRIỂN

1. Kết Luận:

- MongoDB là một hệ quản trị cơ sở dữ liệu phi quan hệ (NoSQL) phổ biến, được phát triển để xử lý dữ liệu phi cấu trúc và mở rộng linh hoạt.
- MongoDB sử dụng mô hình dữ liệu JSON-like (BSON) để lưu trữ dữ liệu, cho phép lưu trữ và truy xuất các dữ liệu có cấu trúc linh hoạt và thay đổi theo thời gian.
- MongoDB hỗ trợ tính năng replica set và sharding, cho phép mở rộng khả năng lưu trữ và tăng cường khả năng chịu tải của hệ thống.
- MongoDB cung cấp các tính năng linh hoạt như index phân tán, truy vấn mạnh mẽ và MapReduce để xử lý dữ liệu lớn và phân tích dữ liệu.
- MongoDB cũng đi kèm với các công cụ quản lý, như MongoDB Compass, giúp quản lý và thao tác với cơ sở dữ liệu MongoDB một cách dễ dàng.

2. Một số định hướng của MongoDB gồm:

- Tăng tính bảo mật: MongoDB đang phát triển các tính năng bảo mật mới để đảm bảo an toàn cho dữ liệu của người dùng, bao gồm mã hóa dữ liệu và kiểm soát truy cập nâng cao.
- Tăng cường tính năng quản trị: MongoDB cung cấp các công cụ quản trị mạnh mẽ để giúp quản trị viên dễ dàng theo dõi và quản lý các cụm MongoDB lớn.
- Phát triển tính năng tích hợp: MongoDB đang tích hợp với các công nghệ khác, bao gồm các công nghệ đám mây, công nghệ container và các nền tảng tích hợp khác, để cung cấp cho người dùng một trải nghiệm tốt hơn khi sử dụng MongoDB trong các môi trường phức tạp.
- Cải thiện khả năng mở rộng: MongoDB đang phát triển các tính năng mới để tăng cường khả năng mở rộng, bao gồm tính năng tự động phân mảnh dữ liệu và

hỗ trợ mạng lưới lớn hơn để đáp ứng nhu cầu lưu trữ dữ liệu ngày càng tăng của người dùng.

- Nâng cao hiệu suất: MongoDB đang tập trung vào việc cải thiện hiệu suất của nó, bao gồm tối ưu hóa truy vấn và cải tiến các tính năng như tìm kiếm văn bản đầy đủ (full-text search).
- Trong tương lai, MongoDB và các hệ quản trị cơ sở dữ liệu NoSQL khác dự kiến sẽ tiếp tục được sử dụng và phát triển.
- Với sự phát triển của dữ liệu phi cấu trúc, dữ liệu lớn và ứng dụng thời gian thực, MongoDB có tiềm năng để đáp ứng nhu cầu lưu trữ và xử lý dữ liệu phức tạp.

=> Tuy nhiên, hệ quản trị cơ sở dữ liệu quan hệ (SQL) vẫn được sử dụng rộng rãi và vẫn phù hợp cho các ứng dụng có yêu cầu về tính nhất quán cao, giao dịch phức tạp và truy vấn phức tạp.

TÀI LIỆU THAM KHẢO

[1] <https://www.mongodb.com>

[2] <https://code.visualstudio.com/docs/azure/mongodb>

[3] <https://viblo.asia/p/tim-hieu-ve-mongodb-4P856ajGIY3>

[4] <https://vietnix.vn/acid-la-gi/>

....

PHỤ LỤC

[1] Source code và data phục vụ nghiên cứu: https://github.com/Ngansan/QLTT_LT

[2] Drive lưu video demo:

<https://drive.google.com/drive/folders/1qQsLgfPcj0bP1xjadNCAMtGyerKvRWj8?usp=sharing>

