A black background with a black square

AI-generated content may be incorrect.TRƯỜNG ĐẠI HỌC THỦY LỢI

KHOA CÔNG NGHỆ THÔNG TIN



**BÁO CÁO BÀI TẬP LỚN**

MÔN PHÂN TÍCH DỮ LIỆU LỚN

**Đề tài: Công cụ Elasticsearch và ứng dụng**

|  |  |  |
| --- | --- | --- |
| Giảng viên hướng dẫn: | Nguyễn Đắc Phương Thảo | |
| Nhóm thực hiện: | 64HTTT3 - 4 | |
| Thành viên nhóm: | Vương Đình Quốc Anh | 2251161948 |
|  | Phan Đức Tuấn | 2251162202 |
|  | Nguyễn Sinh Cung | 2251161960 |
|  | Đinh Thế Phước | 2251162112 |

**Hà Nội, 2025**

Mục lục

[1. Giới thiệu 4](#_Toc388)

[1.1. BigData và tầm quan trọng (Quốc Anh) 4](#_Toc7377)

[1.2. Elasticsearch là gì? (Cung) 6](#_Toc31595)

[1.3. Đặc điểm của elasticsearch (Tuấn) 6](#_Toc7307)

[1.4 Các thức hoạt động của Elasticsearch (Phước) 6](#_Toc22290)

[2. Kiến thức về Elasticsearch 7](#_Toc4024)

[2.1. Cấu trúc của Elasticsearch (Tuấn) 7](#_Toc10059)

[2.2. Cách Elasticsearch hoạt động (Quốc Anh) 9](#_Toc28016)

[2.3. Tại sao Elasticsearch lại có tốc độ truy vấn nhanh? (Cung) 10](#_Toc16404)

[2.3.1. Sử dụng cấu trúc “Inverted Index” (chỉ mục đảo) 10](#_Toc13792)

[2.3.2. Cơ chế phân tán dữ liệu và xử lý song song 10](#_Toc13442)

[2.3.3. Cơ chế lưu trữ tối ưu và nén dữ liệu hiệu quả 11](#_Toc23802)

[2.3.4. Bộ nhớ đệm (Caching) thông minh 11](#_Toc26924)

[2.3.5. Truy vấn gần như thời gian thực (Near Real-Time Search) 11](#_Toc7098)

[2.4. Các thành phần trong hệ sinh thái Elasticsearch (Phước) 11](#_Toc1746)

[3. Giới thiệu về MongoDB 12](#_Toc25776)

[3.1. Các feature của MongoDB gồm có: (Quốc Anh) 13](#_Toc14163)

[3.2. Ưu điểm của MongoDB là gì? (Cung) 13](#_Toc23973)

[3.3. Nhược điểm của MongoDB (Tuấn) 13](#_Toc10027)

[4. Áp dụng Elasticsearch trong hệ thống tìm kiếm nhạc 14](#_Toc3989)

[4.2. Mục tiêu bài toán 14](#_Toc22547)

[4.3. Nguồn dữ liệu (Tuấn) 14](#_Toc18263)

[4.4. Các bước thực hiện 15](#_Toc30077)

[4.4.1. Cài đặt Docker (Quốc Anh) 15](#_Toc13482)

[4.4.2. Tiền xử lý dữ liệu (Tuấn) 16](#_Toc8696)

[4.4.3. Import dữ liệu vào Elasticsearch (elasticsearch.py) (Quốc Anh) 21](#_Toc639)

[4.4.4. Thực hiện truy vấn với Elasticsearch Query để phân tích và tối ưu hóa hiệu suất của các truy vấn trước đó (Phước) 23](#_Toc11466)

[4.5. Kết quả và nhận xét 28](#_Toc15029)

[4.5.1. Hiệu quả của Elasticsearch trong phân tích log: 28](#_Toc22849)

[4.5.2. Nhận xét 29](#_Toc6312)

[5. Triển khai trên MongoDB 29](#_Toc28031)

[5.1. Cài đặt MongoDB 29](#_Toc18753)

[5.2. Cấu hình kết nối và nạp dữ liệu MongoDB (Cung) 30](#_Toc11482)

[5.3. Thực hiện truy vấn (Quốc Anh) 33](#_Toc19553)

[6. Thảo luận và phân tích 34](#_Toc16900)

[6.1. Ưu điểm của Elasticsearch 34](#_Toc2790)

[6.2. Nhược điểm của Elasticsearch 35](#_Toc32655)

[6.3. So sánh với MongoDB 36](#_Toc3483)

[6.4. So sánh thời gian chạy Elasticsearch và MongoDB 37](#_Toc20882)

[7. Kết luận 37](#_Toc25802)

[8. Tài liệu tham khảo 38](#_Toc20858)

# **1. Giới thiệu**

## 1.1. BigData và tầm quan trọng (Quốc Anh)

Dữ liệu lớn là khối lượng lớn, tốc độ cao và/hoặc loại hình thông tin rất đa dạng mà yêu cầu phương thức xử lý mới để cho phép tăng cường ra quyết định, khám phá bên trong và xử lý tối ưu

Đặc trưng của BigData gắn với 3 chữ V:

*Volume (độ lớn)*: Độ lớn của thiết bị lưu trữ tại thời điểm dữ liệu xuất hiện

*Variety (tính đa dạng):* bao gồm nhiều loại dữ liệu đa dạng, phong phú

*Velocity (vận tốc)*: tốc độ phân tích và xử lý dữ liệu nhanh chóng



Các thách thức trong việc quản lý và phân tích dữ liệu lớn.

1. Thách thức về hạ tầng và chi phí đầu tư

* Việc lưu trữ, xử lý và khai thác lượng dữ liệu khổng lồ đòi hỏi **hệ thống máy chủ, mạng và cơ sở dữ liệu phân tán mạnh mẽ**.
* Theo bài báo, năm 2024 các ngân hàng thương mại Việt Nam **đã chi tới 32.437 tỉ đồng cho công nghệ**, tương đương gần **15% tổng chi phí hoạt động** – cho thấy gánh nặng tài chính cực lớn.
* Các ngân hàng nhỏ và vừa gặp khó khăn về vốn, dẫn tới **khả năng triển khai hạn chế và phụ thuộc vào giải pháp thuê ngoài (cloud, data lake thuê)**.

2. Thách thức về chất lượng và tích hợp dữ liệu

* Dữ liệu ngân hàng đến từ **nhiều nguồn khác nhau** (giao dịch, mạng xã hội, hồ sơ tín dụng, ứng dụng di động, dữ liệu khách hàng cá nhân…).
* Việc chuẩn hóa, làm sạch và tích hợp những nguồn dữ liệu không đồng nhất này rất phức tạp, dễ dẫn tới **trùng lặp, sai lệch hoặc thiếu đồng bộ**.
* Một số ngân hàng tiên phong như **NCB** đã phải hợp tác với **Google Cloud triển khai Data Lake và AI/ML** để giải quyết bài toán tích hợp.

3. Thách thức về bảo mật và quyền riêng tư

* Khi dữ liệu khách hàng (giao dịch, hành vi, thông tin định danh) được tập trung, nguy cơ **rò rỉ, tấn công mạng và vi phạm dữ liệu cá nhân** tăng cao.
* Việc áp dụng AI, Big Data và Cloud yêu cầu tuân thủ chặt chẽ **chuẩn an toàn thông tin (ISO/IEC 27001, PCI DSS)** và các quy định về bảo mật dữ liệu (như Nghị định 13/2023/NĐ-CP về bảo vệ dữ liệu cá nhân).

4. Thách thức về nhân lực và năng lực phân tích

* Phân tích dữ liệu lớn đòi hỏi đội ngũ **chuyên gia dữ liệu, kỹ sư AI, chuyên viên phân tích nghiệp vụ (Data Analyst)** – những vị trí còn khan hiếm tại Việt Nam.
* Các ngân hàng nhỏ “loay hoay” vì **thiếu đội ngũ nhân sự công nghệ cao** và **không có chiến lược dữ liệu rõ ràng** nên chưa tận dụng được giá trị của Big Data.

5. Thách thức về mô hình và thuật toán phân tích

* Big Data không chỉ là thu thập dữ liệu, mà còn phải **chọn mô hình phù hợp để phân tích và ra quyết định tự động** (AI, Machine Learning, Predictive Analytics…).
* Nếu không có dữ liệu huấn luyện đủ lớn và sạch, các mô hình có thể **dự đoán sai, dẫn đến rủi ro tín dụng hoặc gian lận**.
* Ví dụ: Techcombank, TPBank, VIB sử dụng AI và Big Data để **phân tích hành vi khách hàng và xét duyệt tín dụng tự động**, nhưng các mô hình này vẫn cần giám sát thường xuyên để tránh sai lệch.

6. Thách thức về quản trị và chiến lược dữ liệu

* Nhiều tổ chức vẫn xem dữ liệu như **“chi phí công nghệ” thay vì “tài sản chiến lược”**, dẫn đến thiếu tầm nhìn dài hạn.
* Quản trị dữ liệu lớn đòi hỏi **xây dựng chính sách, quy trình quản lý vòng đời dữ liệu (Data Governance)**, phân quyền truy cập, và đo lường hiệu quả đầu tư công nghệ.

## 1.2. Elasticsearch là gì? (Cung)

Elasticsearch là một công cụ tìm kiếm dựa trên nền tảng Apache Lucene. Nó cung cấp một bộ máy tìm kiếm dạng phân tán, có đầy đủ công cụ với một giao diện web HTTP có hỗ trợ dữ liệu [JSON](https://topdev.vn/blog/json-la-gi/).Elasticsearch được xây dựng để hoạt động như một server cloud theo cơ chế của RESTful. Điều này giúp nó có thể tương tác và sử dụng bới rất nhiều ngôn ngữ , cũng chính do đó cũng là điểm yếu của nó khi độ bảo mật không cao.

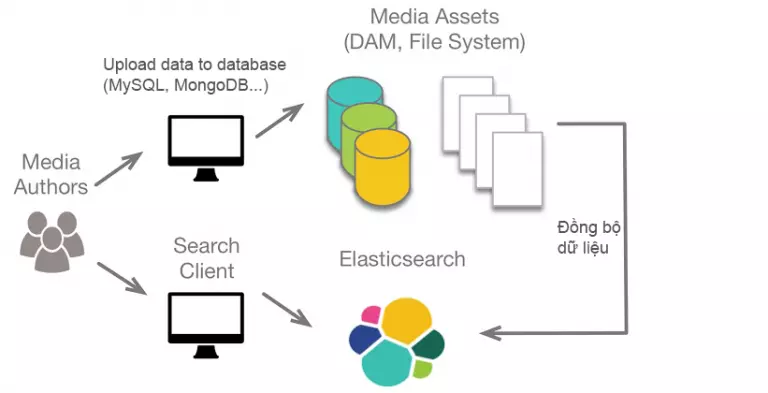
## 1.3. Đặc điểm của elasticsearch (Tuấn)

* Đầu tiên là tìm kiếm dữ liệu rất nhanh. Bởi vì Elasticsearch được xây dựng trên đỉnh Lucene, nó thực hiện rất tốt trong tìm kiếm toàn văn. Elasticsearch cũng là một nền tảng tìm kiếm gần thời gian thực, có nghĩa là sự chậm trễ từ lập chỉ mục tài liệu đến khả năng tìm kiếm tài liệu là rất ngắn, thường chỉ là một giây.
* Đặc điểm thiết yếu của việc được phân phối.Các tài liệu lưu trữ trong Elasticsearch được phân phối trong các thùng chứa khác nhau, được gọi là mảnh vỡ , có thể được sao chép vào cung cấp bản sao dự phòng dữ liệu trong trường hợp lỗi phần cứng. Bản chất phân tán của Elasticsearch cho phép nó mở rộng tới hàng trăm (thậm chí hàng nghìn) máy chủ và xử lý petabyte dữ liệu.
* Elasticsearch có một hệ sinh thái rất tốt, giúp đơn giản hóa quá trình thu thập dữ liệu, trực quan hóa và báo cáo. Bằng cách tích hợp với Beats và Logstash, người dùng có thể dễ dàng xử lý dữ liệu trước khi lập chỉ mục cho Elasticsearch. Đồng thời, Kibana không chỉ cung cấp trực quan hóa dữ liệu Elasticsearch theo thời gian thực mà còn cung cấp giao diện người dùng để người dùng truy cập nhanh dữ liệu

## 1.4 Các thức hoạt động của Elasticsearch (Phước)

Elasticsearch hoạt động như một [Cloud Server](https://vietnix.vn/cloud-server/) có khả năng tìm kiếm thông qua cơ chế [RESTful](https://vietnix.vn/restful-api-la-gi/). Trong đó, người dùng tạo ra các HTTP Request và dữ liệu dạng JSON, sau đó nhập vào Elasticsearch. Các dữ liệu này đều được đánh Index (Chỉ mục), giúp đem lại hiệu quả tìm kiếm rất cao.

Đầu tiên, dữ liệu được đưa vào Elasticsearch từ nhiều nguồn khác nhau, sau đó được phân tích, xử lý, trong quá trình nhập liệu. Tiếp theo, dữ liệu được phân loại và chỉ mục, được đẩy lên Server Elasticsearch. Cuối cùng, người dùng có thể tạo các truy vấn phức tạp và lấy dữ liệu được trả về từ server Elasticsearch.

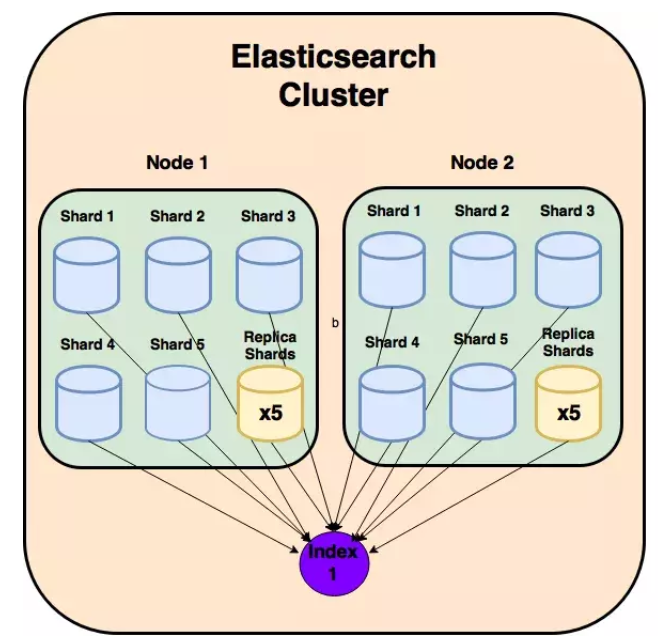


# **2. Kiến thức về Elasticsearch**

## 2.1. Cấu trúc của Elasticsearch (Tuấn)

Cluster:

* Là tập hợp của một hoặc nhiều nodes hoạt động cùng nhau, có khả năng tìm kiếm và lập chỉ mục trên tất cả các nodes cho toàn bộ dữ liệu. Chức năng chính của Cluster là quyết định xem shards nào được phân bổ cho node nào và khi nào thì di chuyển các node để cân bằng lại Cluster.
* Mỗi cluster được định danh bằng một unique name, sử dụng chung cho tất cả các nodes, do vậy việc định danh các cluster trùng tên sẽ gây nên lỗi cho các nodes.
* Mỗi cluster có một node chính (master) được lựa chọn tự động và có thể thay thế khi gặp sự cố.



**Nodes:**

* Là một instance (server) đơn lẻ duy nhất đang chạy của Elasticsearch. Nơi lưu trữ dữ liệu , tham gia thực hiện đánh index của cluster và thực hiện tìm kiếm
* Mỗi node được định danh bằng một *unique name*, được đặt mặc định ngẫu nhiên bởi Universally Unique IDentifier (UUID) tiến hành khi thiết lập hoặc tự định danh. Tên của node rất quan trọng trong việc xác định node này thuộc cluster nào trong Elasticsearch

**Shards và Replicas**:

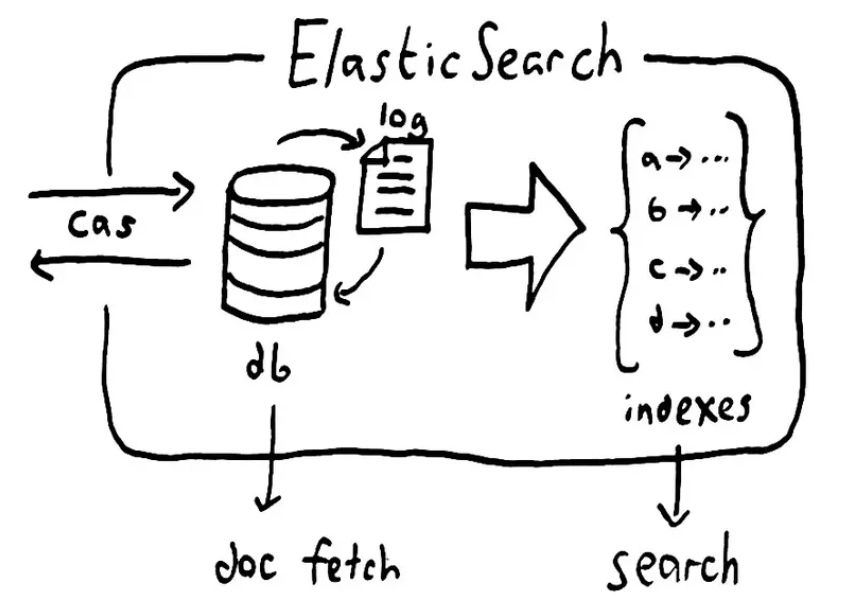
* **Khái niệm**: Các index được chia theo chiều ngang thành các shard, mỗi shard chứa tất cả các thuộc tính của document nhưng chứa ít đối tượng JSON hơn index. Sự phân tách ngang làm cho shard là một node độc lập, có thể được lưu trữ trong bất kỳ node nào. Do vậy, một node có thể có nhiều Shard, vì thế Shard sẽ là đối tượng nhỏ nhất, hoạt động ở mức thấp nhất, đóng vai trò lưu trữ dữ liệu
* **Phân loại**:
* Primary Shard: Primary Shard là nơi lưu trữ dữ liệu và đánh index. Sau khi đánh xong dữ liệu sẽ được vận chuyển tới các Replica Shard. Mặc định của Elasticsearch là mỗi index sẽ có 5 Primary shard và với mỗi Primary shard thì sẽ đi kèm với một Replica Shard
* Replica Shard: Replica Shard đúng như cái tên của nó, nó là nơi lưu trữ dữ liệu nhân bản của Primary Shard. Replica Shard có vai trò đảm bảo tính toàn vẹn của dữ liệu khi Primary Shard xảy ra vấn đề. Ngoài ra Replica Shard có thể giúp tăng cường tốc độ tìm kiếm vì chúng ta có thể setup lượng Replica Shard nhiều hơn mặc định của Elasticsearch

**Index**:

* Là một tập hợp các loại document khác nhau và các thuộc tính của chúng, giúp lưu trữ một lượng lớn dữ liệu có thể vượt qua giới hạn phần cứng của node làm chậm quá trình phản hồi các request từ những node đơn. Do vậy, Index sử dụng khái niệm shards (phân đoạn) để chia nhỏ thành nhiều phần giúp cải thiện hiệu suất
* Khi tạo index, có thể xác định số lượng shard mà bạn muốn
* Index cũng được định danh bằng tên, tên này được sử dụng khi thực hiện các hoạt động lập chỉ mục, tìm kiếm, cập nhật hoặc xóa các document trong index

## 2.2. Cách Elasticsearch hoạt động (Quốc Anh)

* Elasticsearch được xây dựng để hoạt động như một server riêng biệt theo cơ chế của RESTful phục vụ việc tìm kiếm dữ liệu.
* Đầu tiên, dữ liệu thô (raw data) vào Elasticsearch từ nhiều nguồn như log, system indicators và webapp sẽ được phân tích, xử lý, bình thường hóa (normalizes) và làm phong phú thêm (enriches) trong quá trình nhập liệu (Data ingestion) trước khi được lập chỉ mục (index) và đẩy lên Server Elasticsearch ở bước thứ 2 (cách lập chỉ mục mình sẽ có một bài viết chi tiết hơn, không nhắc đến trong này).
* Cuối cùng, sau khi dữ liệu được lập chỉ mục, người dùng có thể tạo các truy vấn phức tạp từ dữ liệu này và sử dụng các tập hợp (aggregations) để truy xuất các bản tóm tắt phức tạp của dữ liệu hay nói ngắn gọn là lấy data trả về từ server Elasticsearch.



## 2.3. Tại sao Elasticsearch lại có tốc độ truy vấn nhanh? (Cung)

* Elasticsearch đạt được **tốc độ truy vấn vượt trội** nhờ sự kết hợp của **cấu trúc dữ liệu đặc biệt, cơ chế lập chỉ mục thông minh và kiến trúc phân tán**. Cụ thể:

### 2.3.1. Sử dụng cấu trúc “Inverted Index” (chỉ mục đảo)

* Đây là yếu tố cốt lõi giúp Elasticsearch tìm kiếm nhanh hơn rất nhiều so với cơ sở dữ liệu truyền thống.
* Thay vì quét toàn bộ dữ liệu, Elasticsearch **lưu trữ mỗi từ khóa cùng danh sách các tài liệu chứa nó**.
* Khi người dùng nhập truy vấn, hệ thống chỉ cần **tra cứu vị trí của từ khóa trong chỉ mục đảo**, giúp phản hồi kết quả gần như tức thì, ngay cả với hàng triệu bản ghi.

### 2.3.2. Cơ chế phân tán dữ liệu và xử lý song song

* Dữ liệu được **chia nhỏ thành các shard** và **phân tán trên nhiều node** trong cluster.
* Khi truy vấn, Elasticsearch **gửi yêu cầu đồng thời tới tất cả các shard**, mỗi shard tìm kiếm trên phần dữ liệu của mình, rồi **hợp nhất kết quả**.
* Nhờ đó, quá trình tìm kiếm diễn ra **song song** và tận dụng tối đa sức mạnh của toàn bộ hệ thống.

### 2.3.3. Cơ chế lưu trữ tối ưu và nén dữ liệu hiệu quả

* Elasticsearch dựa trên **Apache Lucene**, sử dụng các cấu trúc dữ liệu như **segment file** và **term dictionary** để lưu trữ dữ liệu một cách gọn nhẹ, giúp truy cập nhanh hơn từ đĩa.
* Ngoài ra, các dữ liệu đã lập chỉ mục được **nén và sắp xếp** giúp giảm kích thước và tăng tốc độ đọc.

### 2.3.4. Bộ nhớ đệm (Caching) thông minh

* Elasticsearch có cơ chế **query cache và shard request cache**, lưu kết quả các truy vấn thường xuyên sử dụng.
* Nhờ đó, các truy vấn lặp lại hoặc tương tự sẽ được **phản hồi ngay từ bộ nhớ đệm**, không cần quét lại dữ liệu.

### 2.3.5. Truy vấn gần như thời gian thực (Near Real-Time Search)

* Mỗi khi dữ liệu được ghi vào, Elasticsearch nhanh chóng **refresh chỉ mục (mặc định mỗi 1 giây)**, giúp dữ liệu mới có thể được tìm kiếm ngay gần như tức thì.
* Cơ chế này giúp hệ thống vừa ghi – vừa truy vấn mà vẫn giữ được tốc độ cao.

## **2.4. Các thành phần trong hệ sinh thái Elasticsearch** (Phước)

Hệ sinh thái **Elastic Stack** (trước đây gọi là **ELK Stack**) gồm các công cụ mã nguồn mở do Elastic.co phát triển, hỗ trợ toàn bộ chu trình dữ liệu: **thu thập – xử lý – lưu trữ – tìm kiếm – phân tích – trực quan hóa**. Các thành phần hoạt động thống nhất tạo nên nền tảng xử lý dữ liệu mạnh mẽ, phục vụ giám sát, phân tích log và hỗ trợ ra quyết định theo thời gian thực.

**Elasticsearch** là **trung tâm của hệ thống**, chịu trách nhiệm **lưu trữ, lập chỉ mục và tìm kiếm dữ liệu**. Dữ liệu được lưu dưới dạng **document JSON**, được phân tích và lập **chỉ mục đảo (inverted index)** để tăng tốc độ truy vấn. Elasticsearch sử dụng mô hình **phân tán (distributed model)**, chia dữ liệu thành các **shard** và **replica**, cho phép mở rộng linh hoạt và đảm bảo độ tin cậy cao. Ngoài khả năng tìm kiếm văn bản, nó còn hỗ trợ **phân tích dữ liệu thời gian thực** và **ứng dụng machine learning** để phát hiện xu hướng hoặc bất thường.

**Logstash** đóng vai trò **bộ xử lý dữ liệu trung gian**, thu thập và chuyển đổi dữ liệu từ nhiều nguồn khác nhau (log hệ thống, API, cảm biến…). Dữ liệu được lọc, chuẩn hóa và gửi vào Elasticsearch để lưu trữ. Nhờ khả năng kết nối với nhiều nguồn và định dạng, Logstash giúp hợp nhất dữ liệu phân tán, tạo đầu vào nhất quán cho quá trình tìm kiếm và phân tích.

**Kibana** là **giao diện trực quan hóa** cho Elasticsearch, cho phép người dùng **xem, phân tích và trình bày dữ liệu** qua biểu đồ, bảng, bản đồ và dashboard. Kibana hỗ trợ **phân tích thời gian thực, truy vấn linh hoạt, giám sát hệ thống và thiết lập cảnh báo**, đồng thời tích hợp mô-đun **Machine Learning** để phát hiện bất thường trong dữ liệu. Đây là công cụ giúp biến dữ liệu kỹ thuật thô thành thông tin trực quan, dễ hiểu.

Ở tầng thu thập, **Beats** là **bộ agent nhẹ**, được cài trên các máy chủ hoặc thiết bị để thu thập dữ liệu và gửi về Logstash hoặc Elasticsearch. Mỗi loại Beats phục vụ một mục đích cụ thể: *Filebeat* thu thập log, *Metricbeat* theo dõi hiệu năng, *Packetbeat* giám sát mạng, *Winlogbeat* lấy log từ Windows, còn *Heartbeat* kiểm tra trạng thái hoạt động của dịch vụ.

**Elastic Agent** là **phiên bản hợp nhất hiện đại của Beats**, cho phép quản lý tập trung nhiều loại dữ liệu và agent qua giao diện **Fleet trong Kibana**. Nó giúp đơn giản hóa triển khai, giám sát và bảo mật trong quy mô lớn, đồng thời tích hợp tính năng phát hiện mối đe dọa và phản ứng tự động.

Toàn bộ các thành phần của Elastic Stack hoạt động theo chuỗi: **Beats/Elastic Agent → Logstash → Elasticsearch → Kibana**. Chuỗi này đảm bảo dữ liệu được **thu thập – xử lý – lập chỉ mục – trực quan hóa** một cách liên tục và thống nhất, giúp người dùng có thể **phân tích, giám sát và ra quyết định nhanh chóng dựa trên dữ liệu thời gian thực**.

# **3. Giới thiệu về MongoDB**

* MongoDB là một database hướng tài liệu (document), một dạng NoSQL database. Vì thế, MongoDB sẽ tránh cấu trúc table-based của relational database để thích ứng với các tài liệu như JSON có một schema rất linh hoạt gọi là BSON. **MongoDB** sử dụng lưu trữ dữ liệu dưới dạng Document JSON nên mỗi một collection sẽ các các kích cỡ và các document khác nhau. Các dữ liệu được lưu trữ trong document kiểu JSON nên truy vấn sẽ rất nhanh.

## 3.1. Các feature của MongoDB gồm có: (Quốc Anh)

* **Các ad hoc query:** hỗ trợ search bằng field, các phép search thông thường, regular expression searches, và range queries.
* **Indexing:** bất kì field nào trong BSON document cũng có thể được index.
* **Replication:** có ý nghĩa là “nhân bản”, là có một phiên bản giống hệt phiên bản đang tồn tại, đang sử dụng. Với cơ sở dữ liệu, nhu cầu lưu trữ lớn, đòi hỏi cơ sở dữ liệu toàn vẹn, không bị mất mát trước những sự cố ngoài dự đoán là rất cao. Vì vậy, người ta nghĩ ra khái niệm “nhân bản”, tạo một phiên bản cơ sở dữ liệu giống hệt cơ sở dữ liệu đang tồn tại, và lưu trữ ở một nơi khác, đề phòng có sự cố.
* **Aggregation:** Các Aggregation operation xử lý các bản ghi dữ liệu và trả về kết quả đã được tính toán. Các phép toán tập hợp nhóm các giá trị từ nhiều Document lại với nhau, và có thể thực hiện nhiều phép toán đa dạng trên dữ liệu đã được nhóm đó để trả về một kết quả duy nhất. Trong SQL, count(\*) và GROUP BY là tương đương với Aggregation trong MongoDB.
* **Lưu trữ file:** MongoDB được dùng như một hệ thống file tận dụng những function trên và hoạt động như một cách phân phối qua sharding.

## 3.2. Ưu điểm của MongoDB là gì? (Cung)

* Dữ liệu lưu trữ phi cấu trúc, không có tính ràng buộc, toàn vẹn nên tính sẵn sàng cao, hiệu suất lớn và dễ dàng mở rộng lưu trữ.
* Dữ liệu được caching (ghi đệm) lên RAM, hạn chế truy cập vào ổ cứng nên tốc độ đọc và ghi cao.

## 3.3. Nhược điểm của MongoDB (Tuấn)

* Không ứng dụng được cho các mô hình giao dịch nào có yêu cầu độ chính xác cao do không có ràng buộc.
* Không có cơ chế transaction (giao dịch) để phục vụ các ứng dụng ngân hàng.
* Dữ liệu lấy RAM làm trọng tâm hoạt động vì vậy khi hoạt động yêu cầu một bộ nhớ RAM lớn.
* 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.

# **4. Áp dụng Elasticsearch trong hệ thống tìm kiếm nhạc**

4.1. Mô tả bài toán (Quốc Anh)

Trong các nền tảng nghe nhạc trực tuyến hiện nay, việc cung cấp khả năng **tìm kiếm bài hát, nghệ sĩ và album nhanh chóng, chính xác** là yếu tố quan trọng để nâng cao trải nghiệm người dùng. Tuy nhiên, với khối lượng dữ liệu âm nhạc khổng lồ gồm hàng triệu bản ghi, cùng nhiều biến thể về tên bài hát, cách viết có dấu – không dấu hoặc sai chính tả, các hệ thống tìm kiếm truyền thống thường phản hồi chậm và thiếu chính xác.

Bài toán đặt ra là cần **xây dựng một hệ thống tìm kiếm nhạc thông minh** có khả năng truy vấn nhanh, hỗ trợ **tìm kiếm toàn văn (full-text search)**, **gợi ý tự động (autocomplete)**, **xử lý lỗi nhập liệu (fuzzy search)** và **xếp hạng kết quả theo độ liên quan**. Việc ứng dụng **Elasticsearch** – một nền tảng tìm kiếm và phân tích dữ liệu phân tán – sẽ giúp hệ thống có thể **lưu trữ, lập chỉ mục và truy xuất dữ liệu âm nhạc** theo thời gian thực, đáp ứng yêu cầu về tốc độ và khả năng mở rộng.

## 4.2. Mục tiêu bài toán (Cung)

* Ứng dụng Elasticsearch để xây dựng **hệ thống tìm kiếm nhạc tốc độ cao và có độ chính xác cao**.
* Hỗ trợ **tìm kiếm tiếng Việt không dấu, đồng nghĩa và lỗi gõ thường gặp**.
* Cung cấp chức năng **gợi ý bài hát, nghệ sĩ hoặc album** trong quá trình người dùng nhập từ khóa.
* Cho phép **xếp hạng kết quả theo độ phổ biến, mức độ liên quan và hành vi người dùng**.
* Tạo nền tảng **mở rộng linh hoạt**, có khả năng xử lý **lượng truy vấn lớn trong thời gian thực**, phục vụ cho các ứng dụng nghe nhạc trực tuyến hiện đại.

## 4.3. Nguồn dữ liệu (Tuấn)

Nguồn: [MillionSongSubset](http://millionsongdataset.com/pages/getting-dataset/#subset)

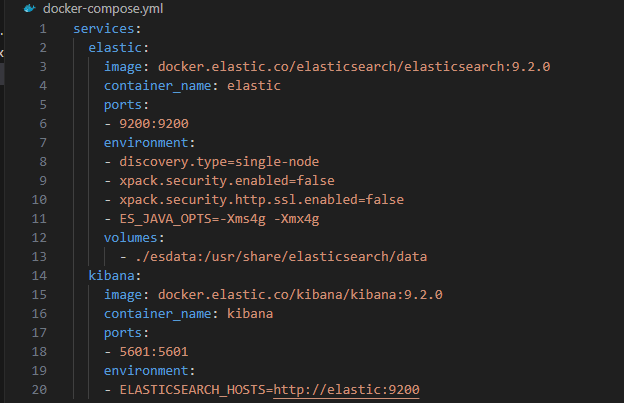
## 4.4. Các bước thực hiện

### 4.4.1. Cài đặt Docker (Quốc Anh)

Hướng dẫn cài đặt Docker từ [trang chủ](https://docs.docker.com/desktop/setup/install/windows-install/)

Cấu hình cơ bản và khởi động các dịch vụ

* Cấu hình file docker-compose.yml



* Vào command prompt và chạy docker compose up -d
* Docker:

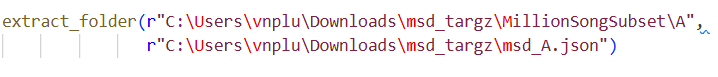


### 4.4.2. Tiền xử lý dữ liệu (Tuấn)

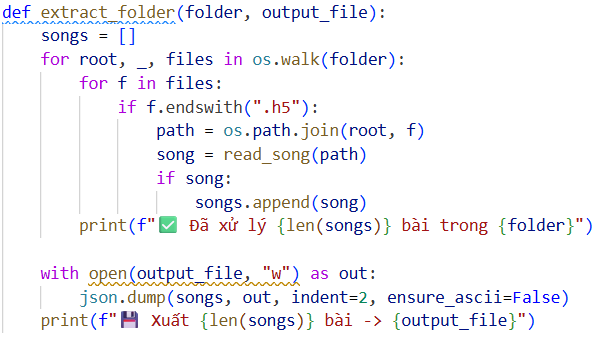
B1: Phân tích dữ liệu đầu vào

* Sau khi đọc xem dữ liệu, ta có thể trích dẫn đặc trưng : “Tittler”; “Artist”; “Duration”; “Tempo”; “Year”; “Text”.
* Cấu trúc:
  + Tittle: Tiêu đề bài hát
  + Artist: Nhạc sĩ
  + Duration: Thời lượng bài hát
  + Tempo: Nhịp điệu bài hát
  + Text: Nhận xét về bài hát
  + Year: Năm xuất bản

B2: Làm sạch dữ liệu



* Hàm làm sạch dữ liệu



* Hàm tách các đặc trưng như tittle, artist, duration, tempo, year

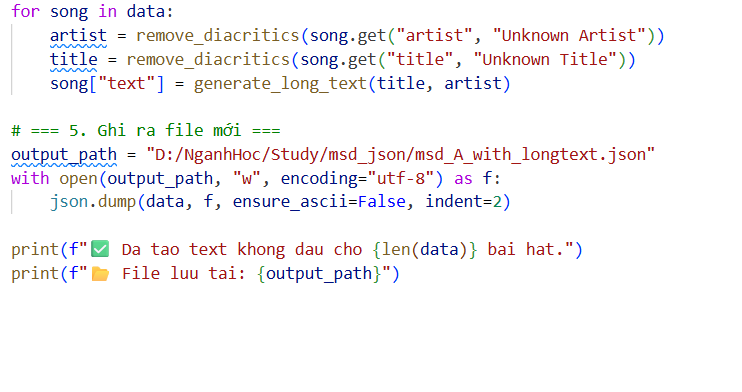
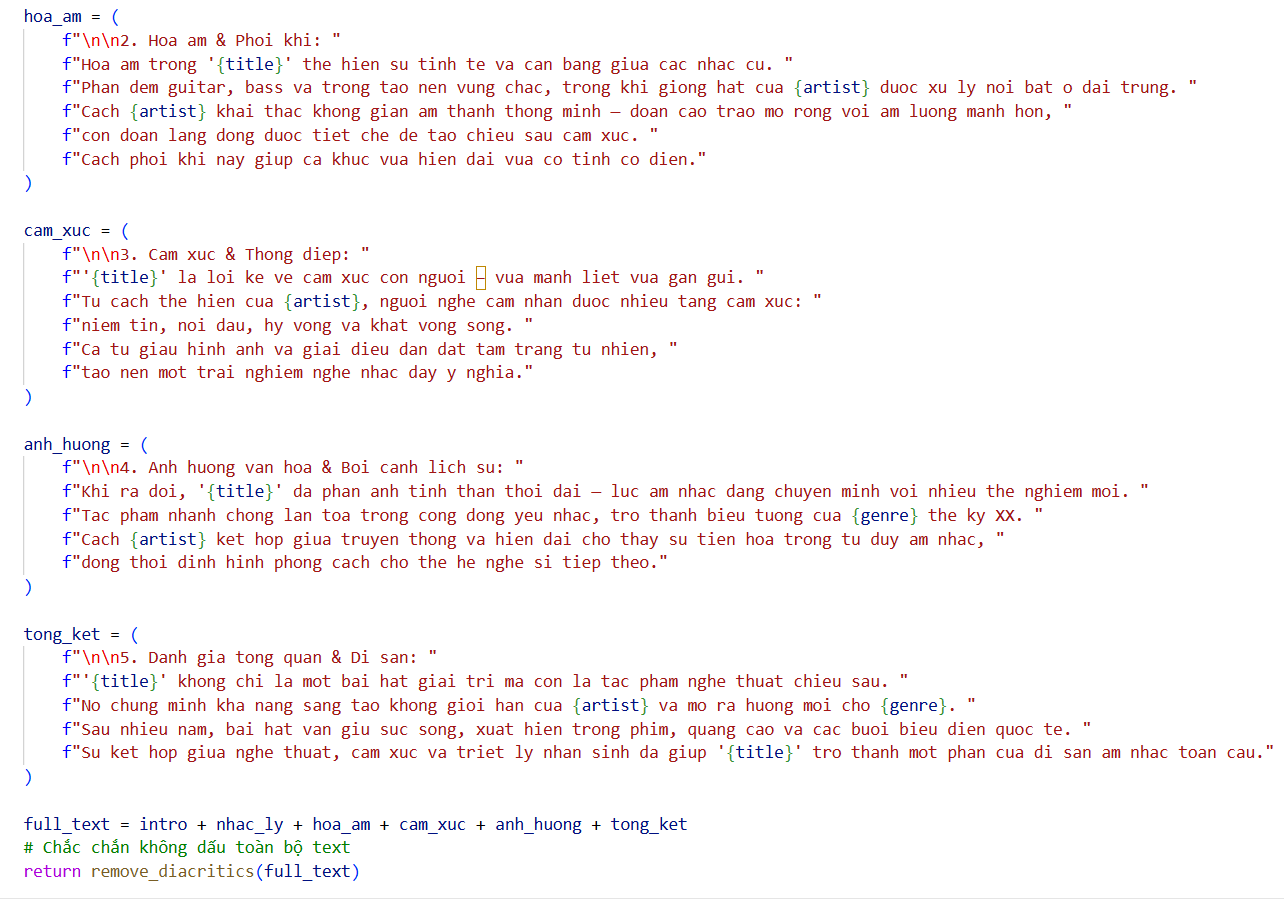


* Xuất dữ liệu từ file .h5 ra file json

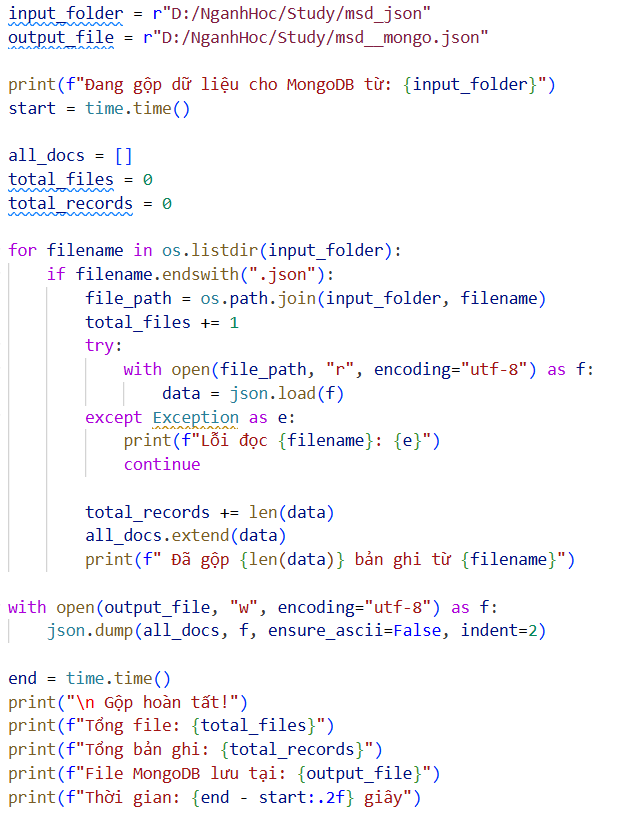


B3: Thêm trường text nhận xét

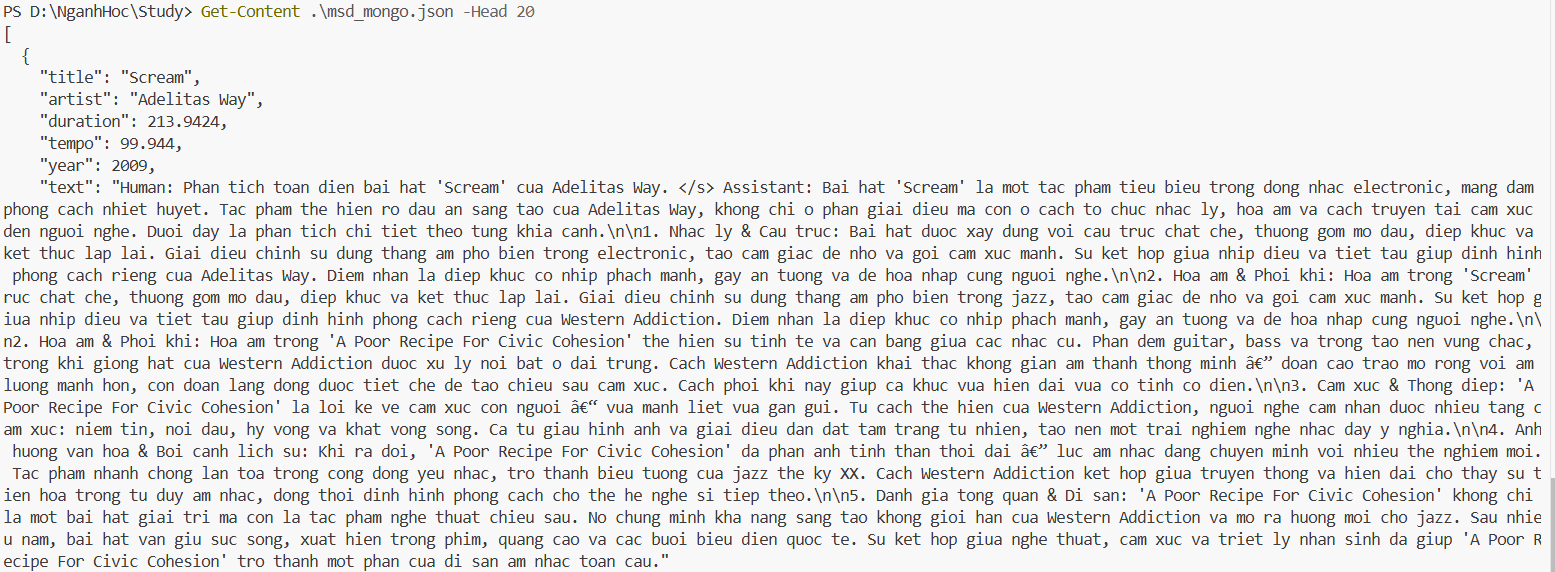




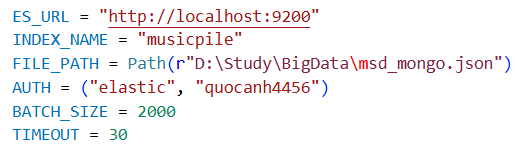
B4: Gộp các file json đã tạo được thành 1 file



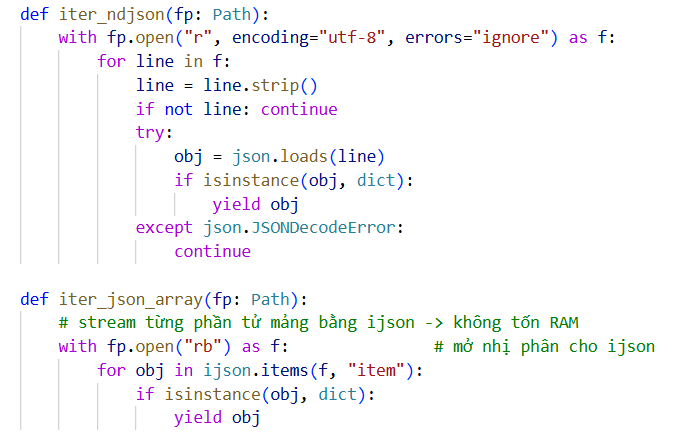
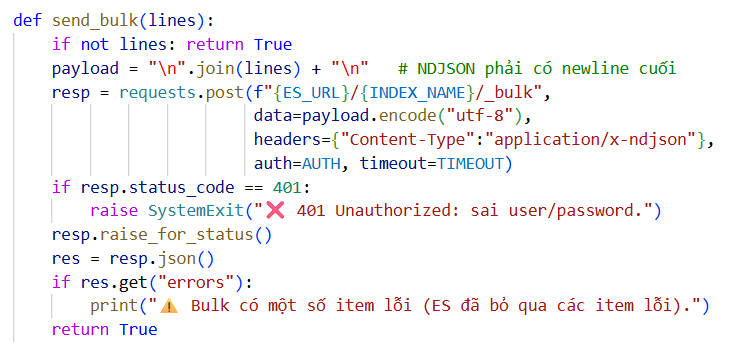
* Data sau khi gộp file



### 4.4.3. Import dữ liệu vào Elasticsearch (elasticsearch.py) (Quốc Anh)

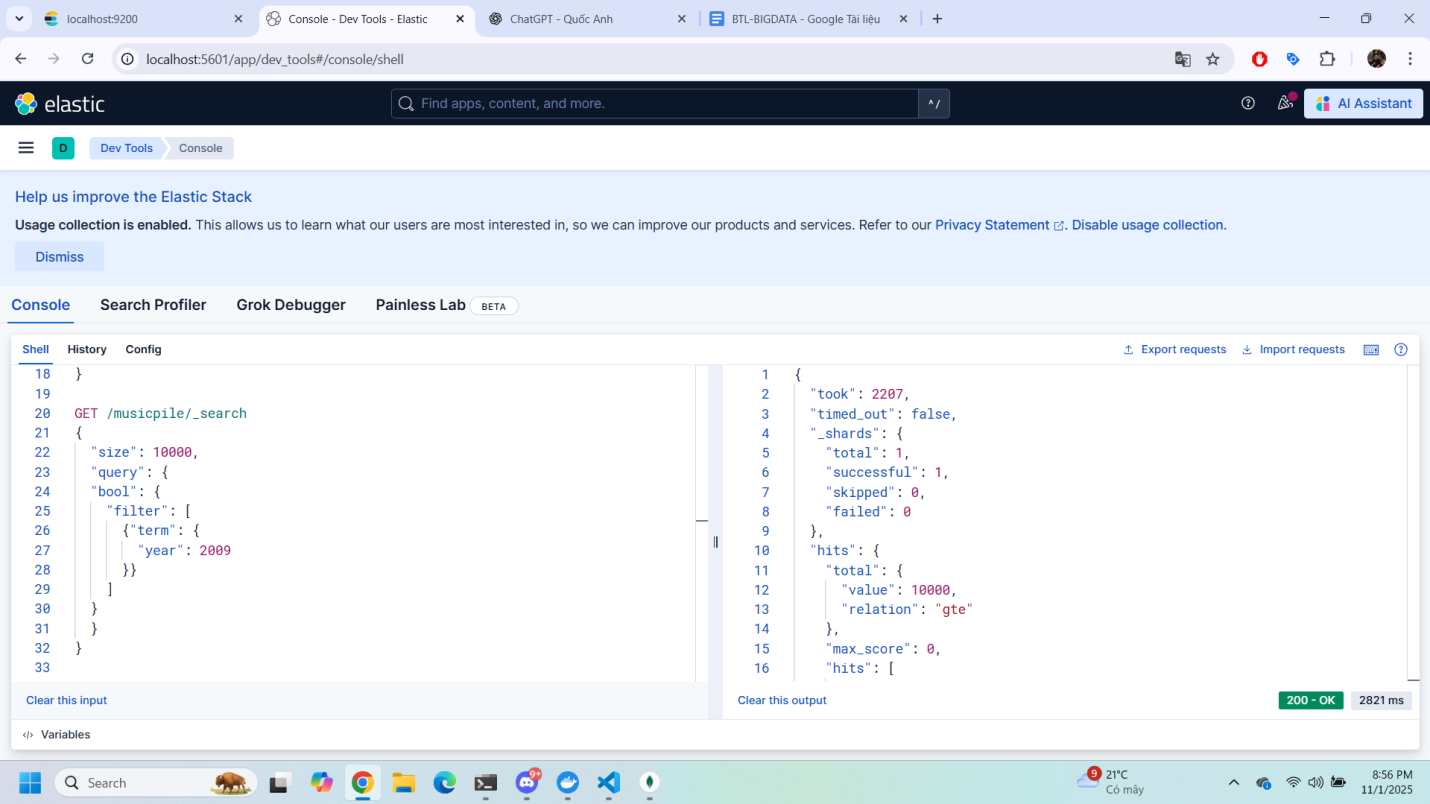
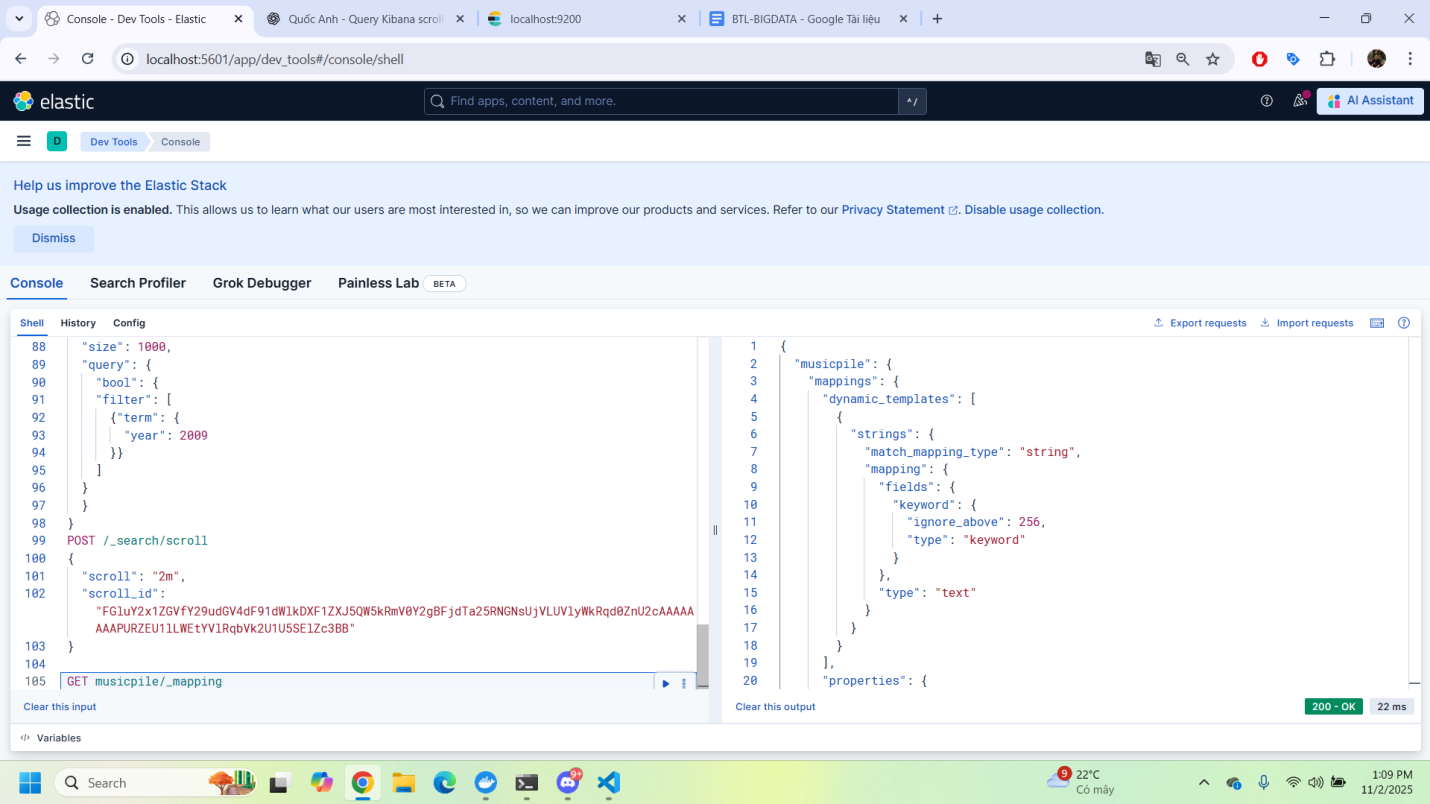
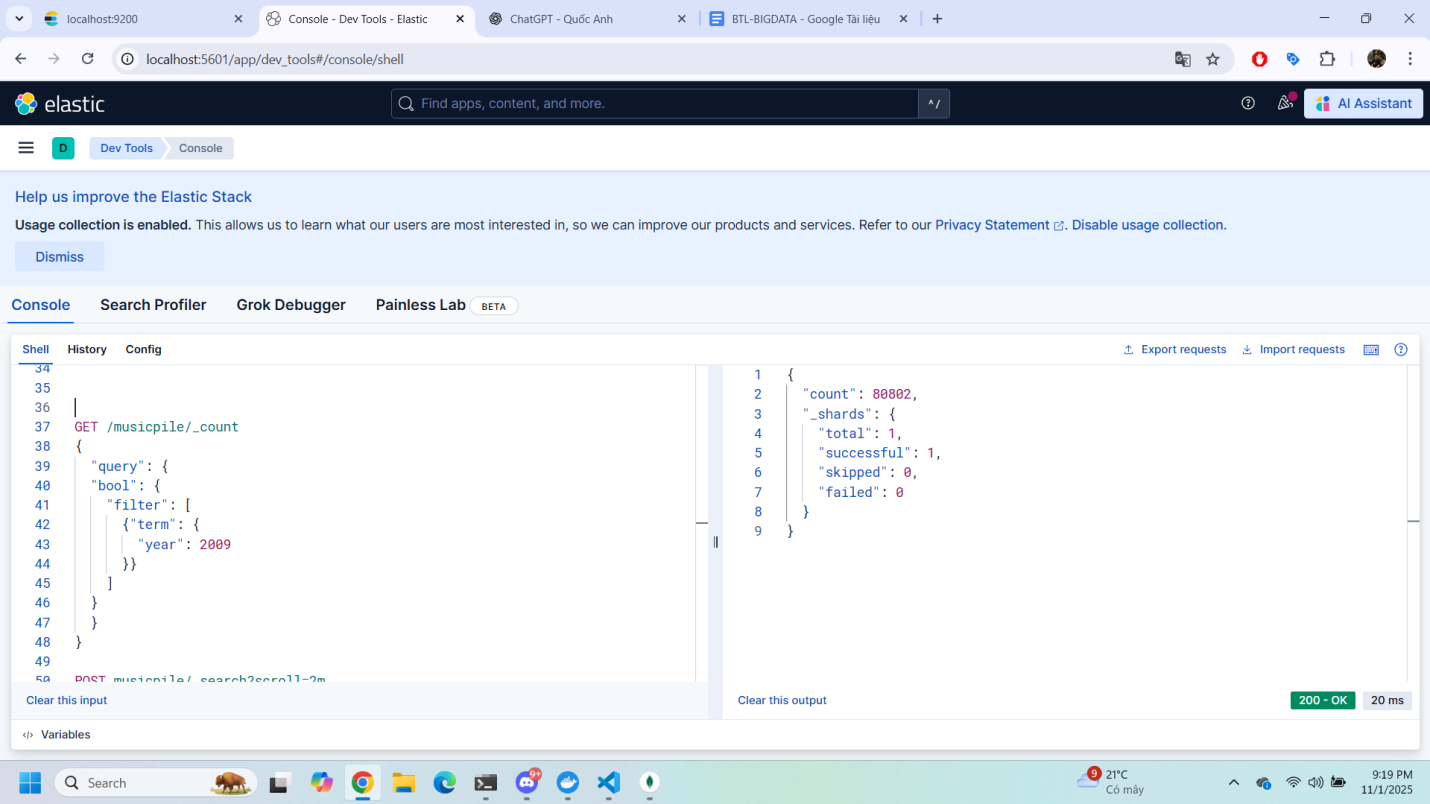
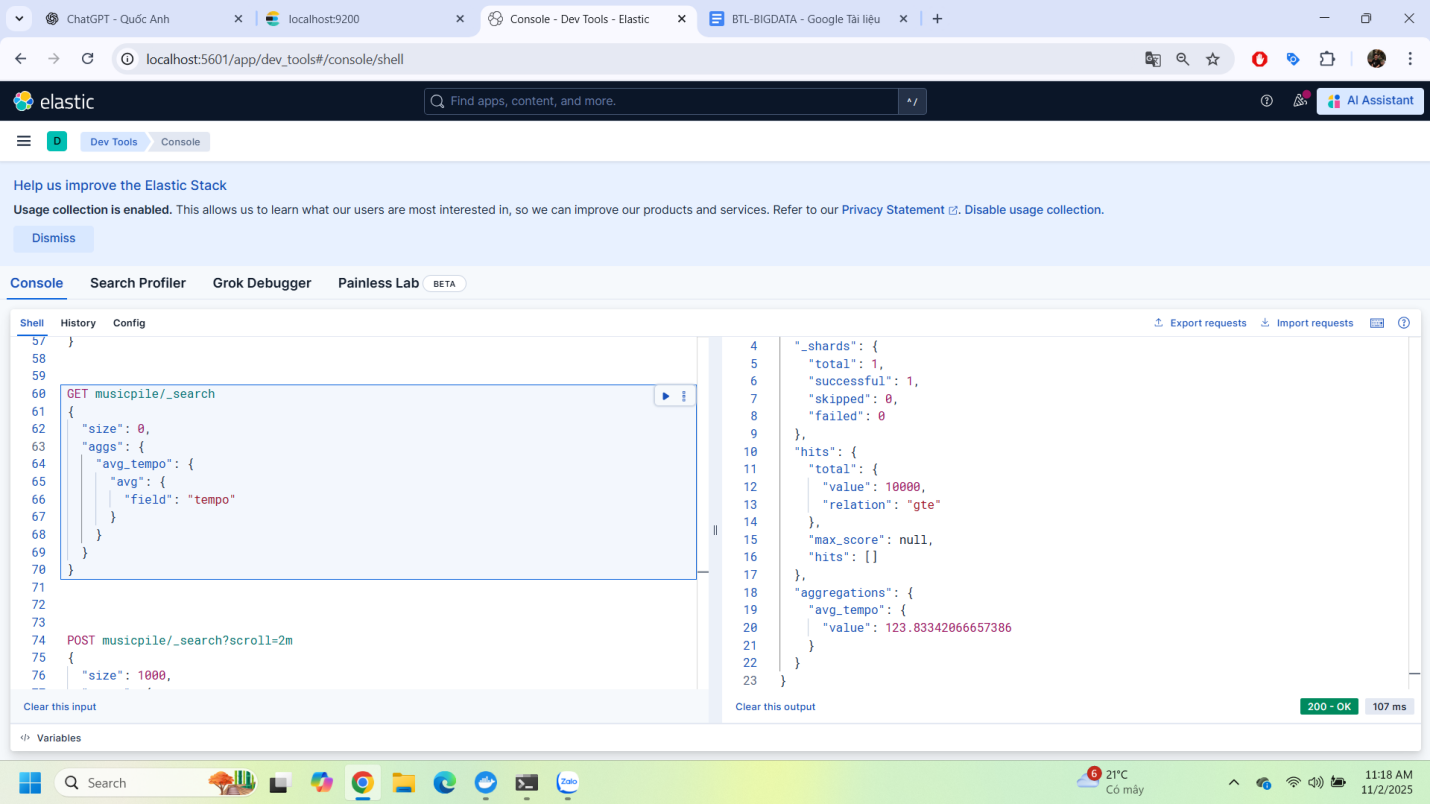
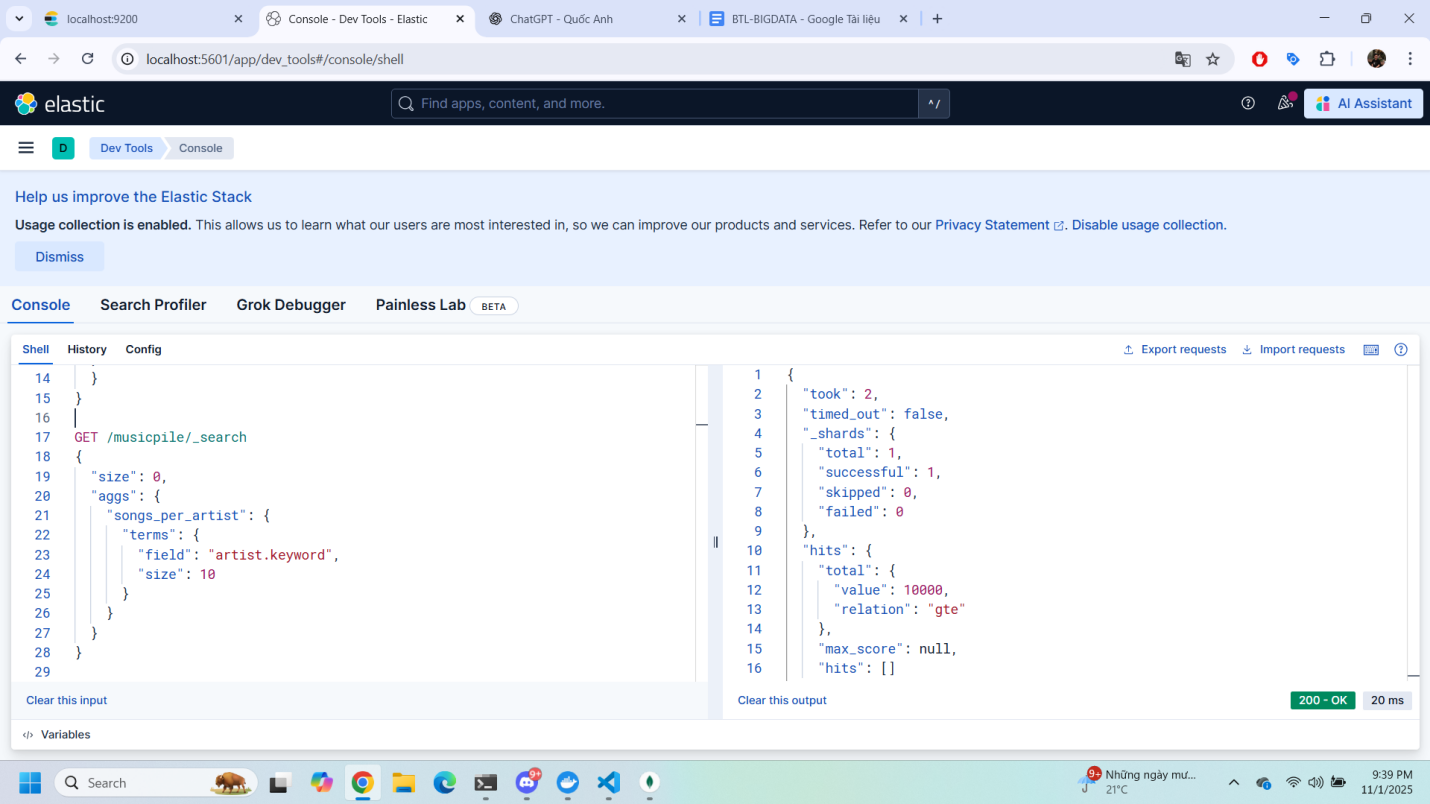
B1. Kết nối Elasticsearch

B2. Đọc file Json và thêm từng bản ghi vào Elasticsearch

* Đọc từng document từ file
* Ghi vào Elasticsearch

### 4.4.4. Thực hiện truy vấn với Elasticsearch Query để phân tích và tối ưu hóa hiệu suất của các truy vấn trước đó (Phước)

Câu truy vấn cơ bản

* Truy vấn tất cả bản ghi có “năm: 2009”
* Giải thích: Truy vấn này **dùng để lọc tất cả các document có year = 2009 trong index musicpile**, trả về tối đa 10.000 kết quả, và cho phép xem thời gian xử lý của Elasticsearch (để đo hiệu năng hoặc phân tích dữ liệu).
* Truy vấn xem cấu trúc dữ liệu
* Giải thích: Hiển thị **cấu trúc lưu trữ của index musicpile**, cho thấy Elasticsearch tự động ánh xạ các trường dạng chuỗi thành kiểu **text** (cho tìm kiếm) và **keyword** (cho lọc chính xác), giúp hệ thống tìm kiếm và phân tích dữ liệu nhanh, chính xác hơn.
* Truy vấn đếm số lượng các bản ghi có năm = 2009
* Giải thích: Truy vấn này đếm xem có bao nhiêu bản ghi trong index musicpile có năm = 2009,kết quả: **80.802 bản ghi**, chạy trong **20 mili-giây**, **thành công 100%**. Dùng để kiểm tra nhanh số lượng dữ liệu sau khi import hoặc để xác nhận filter chính xác.
* Truy vấn tính nhịp độ trung bình 
* Truy vấn tìm kiếm nghệ sĩ nào có nhiều bài hát nhất trong index
* Giải thích: Truy vấn này dùng để **thực hiện phép thống kê (aggregation)**, cụ thể là **đếm số lượng bài hát theo từng nghệ sĩ** trong index musicpile, và **hiển thị top 10 nghệ sĩ có nhiều bài hát nhất**.

Câu truy vấn nâng cao

* Truy vấn truy xuất toàn bộ dữ liệu “year = 2009”
* Giải thích: Lấy batch đầu tiên (1.000 bản ghi) và tạo scroll context sống trong 2 phút. Gửi scroll\_id vừa nhận ở trên để lấy thêm 1.000 bản ghi tiếp theo.

## 4.5. Kết quả và nhận xét

### 4.5.1. Hiệu quả của Elasticsearch trong phân tích log:

Elasticsearch cho phép triển khai hệ thống **tìm kiếm và phân tích dữ liệu âm nhạc** nhanh chóng, linh hoạt và hiệu quả trên quy mô lớn. Nhờ cơ chế lập chỉ mục (index) và lưu trữ dữ liệu dạng tài liệu JSON, người dùng có thể dễ dàng truy vấn các bài hát theo **tên, nghệ sĩ, thể loại, năm phát hành hoặc nội dung phân tích chi tiết**. Các truy vấn giúp khai thác dữ liệu theo nhiều góc độ: từ việc **tìm kiếm bài hát có từ khóa cụ thể**, **lọc nhạc theo năm**, cho đến **thống kê số lượng bài hát theo nghệ sĩ hoặc tính trung bình nhịp độ**.

Với khả năng xử lý song song và phản hồi trong thời gian thực, Elasticsearch giúp **rút ngắn đáng kể thời gian truy vấn**, kể cả khi dữ liệu lên đến hàng trăm nghìn bản ghi. Khi kết hợp với Kibana, hệ thống còn có thể **trực quan hóa kết quả tìm kiếm, biểu đồ thống kê và xu hướng âm nhạc theo từng năm**, hỗ trợ hiệu quả cho các ứng dụng **phân tích, gợi ý hoặc nghiên cứu âm nhạc thông minh**.

### 4.5.2. Nhận xét

Trong quá trình thực hiện các truy vấn trên bộ dữ liệu **musicpile**, bao gồm các trường *Title* (tiêu đề bài hát), *Artist* (nghệ sĩ), *Duration* (thời lượng), *Tempo* (nhịp điệu), *Text* (nhận xét, mô tả bài hát) và *Year* (năm xuất bản), hệ thống **Elasticsearch** cho thấy khả năng xử lý dữ liệu nhanh chóng, chính xác và ổn định. Các thao tác truy vấn được thực hiện thông qua **Kibana Dev Tools** và **mã Python**, sử dụng các câu lệnh như search, count, term, range và scroll để tìm kiếm, lọc và thống kê dữ liệu.

Kết quả cho thấy, **tốc độ truy vấn của Elasticsearch rất cao**. Các câu lệnh search và count trả về kết quả chỉ trong vài mili-giây đến vài giây, dù dữ liệu có quy mô hàng chục nghìn bản ghi. Nhờ cơ chế lập chỉ mục (index) và cấu trúc lưu trữ dạng JSON, Elasticsearch có thể xử lý đồng thời nhiều truy vấn mà không gây quá tải hệ thống. Khi áp dụng truy vấn scroll để lấy dữ liệu vượt giới hạn 10.000 bản ghi, hệ thống vẫn duy trì hiệu năng ổn định, không xảy ra lỗi timeout hoặc mất context.

**Kết quả truy vấn trả về đúng và rõ ràng**, phản ánh chính xác điều kiện lọc đã đặt ra. Việc tìm kiếm theo year, artist hoặc theo từ khóa trong trường text cho ra dữ liệu đầy đủ, logic và dễ hiểu. Dạng kết quả JSON giúp người dùng dễ dàng phân tích, trích xuất hoặc chuyển đổi sang các định dạng khác như CSV phục vụ cho trực quan hóa và báo cáo.

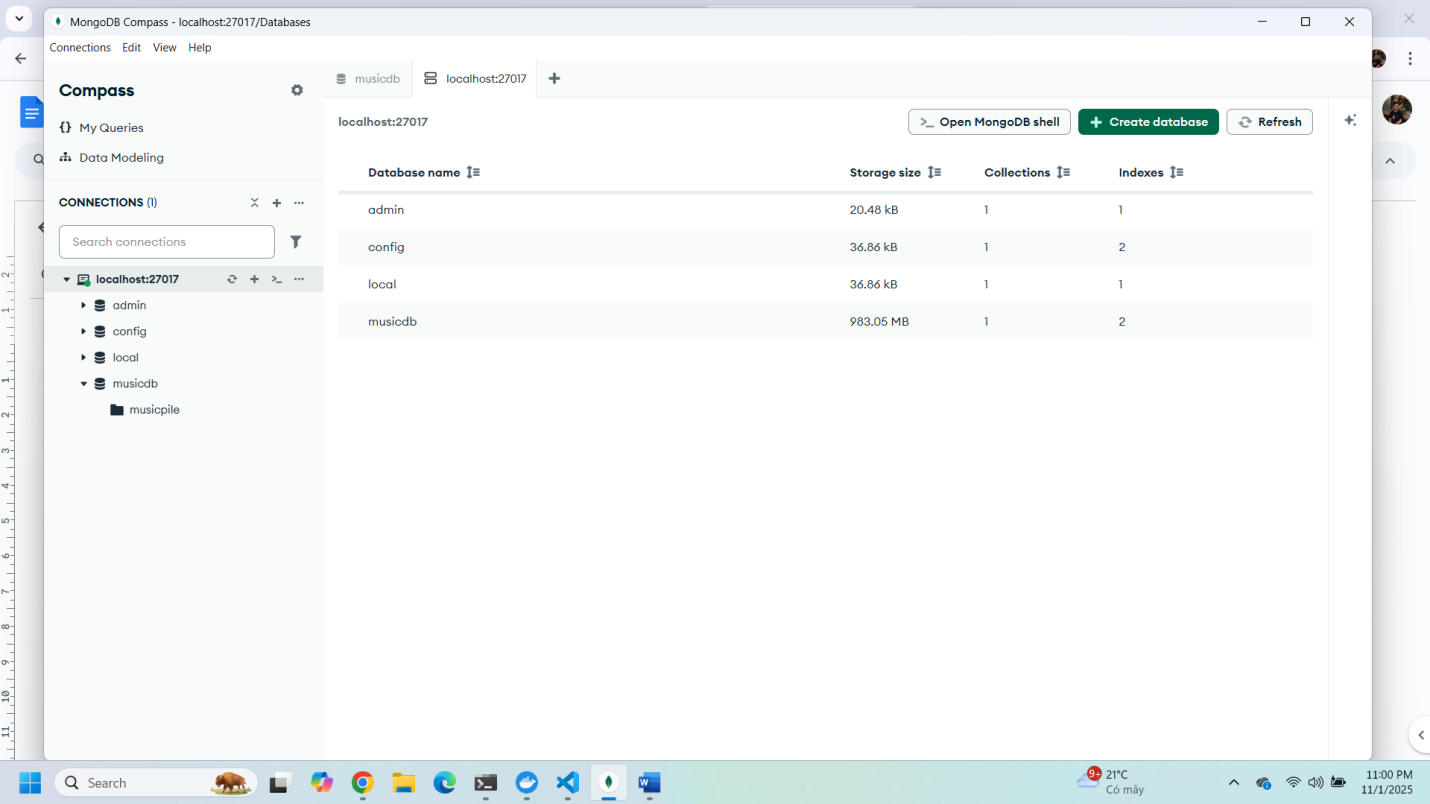
Về **hiệu năng xử lý và tính ổn định**, Elasticsearch chứng minh khả năng hoạt động tốt với khối lượng dữ liệu lớn. Cơ chế phân tán (distributed architecture) và xử lý song song (parallel processing) giúp tăng tốc độ tìm kiếm và đảm bảo độ tin cậy khi truy cập dữ liệu. Việc kết hợp cùng **Kibana** cho phép trực quan hóa kết quả truy vấn, hiển thị biểu đồ, thống kê và phân tích dữ liệu âm nhạc một cách trực quan, giúp người dùng nhận diện nhanh xu hướng hoặc đặc điểm nổi bật của từng năm và từng nghệ sĩ.

Tổng thể, hệ thống Elasticsearch mang lại hiệu quả cao trong **tìm kiếm và phân tích dữ liệu âm nhạc**. Nó giúp rút ngắn đáng kể thời gian truy vấn, đảm bảo độ chính xác của kết quả, dễ dàng mở rộng quy mô và tích hợp vào các hệ thống gợi ý, phân tích hoặc nghiên cứu âm nhạc thông minh trong tương lai.

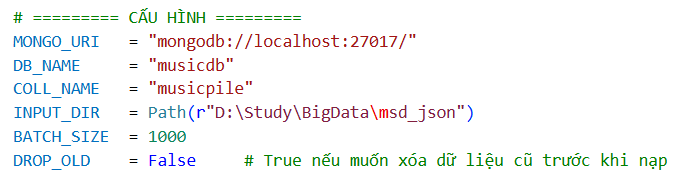
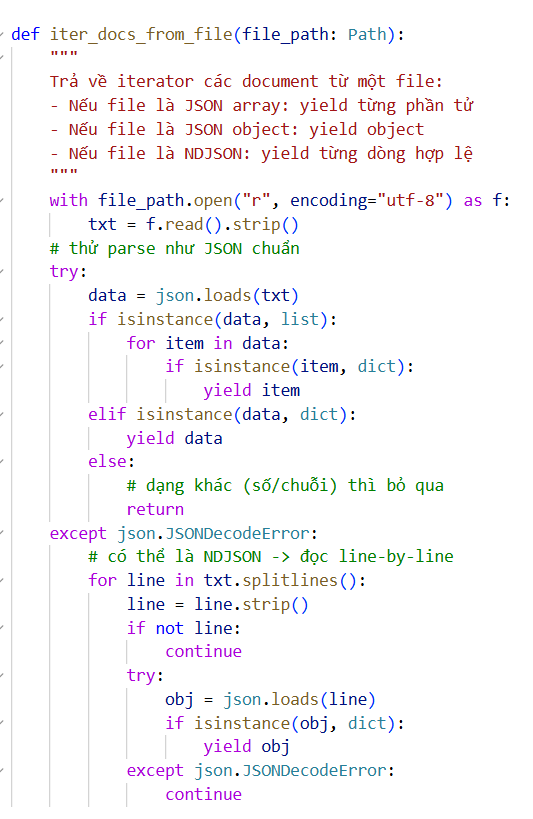
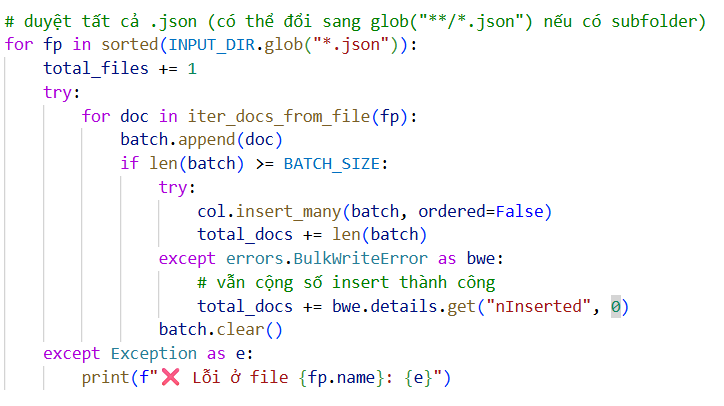
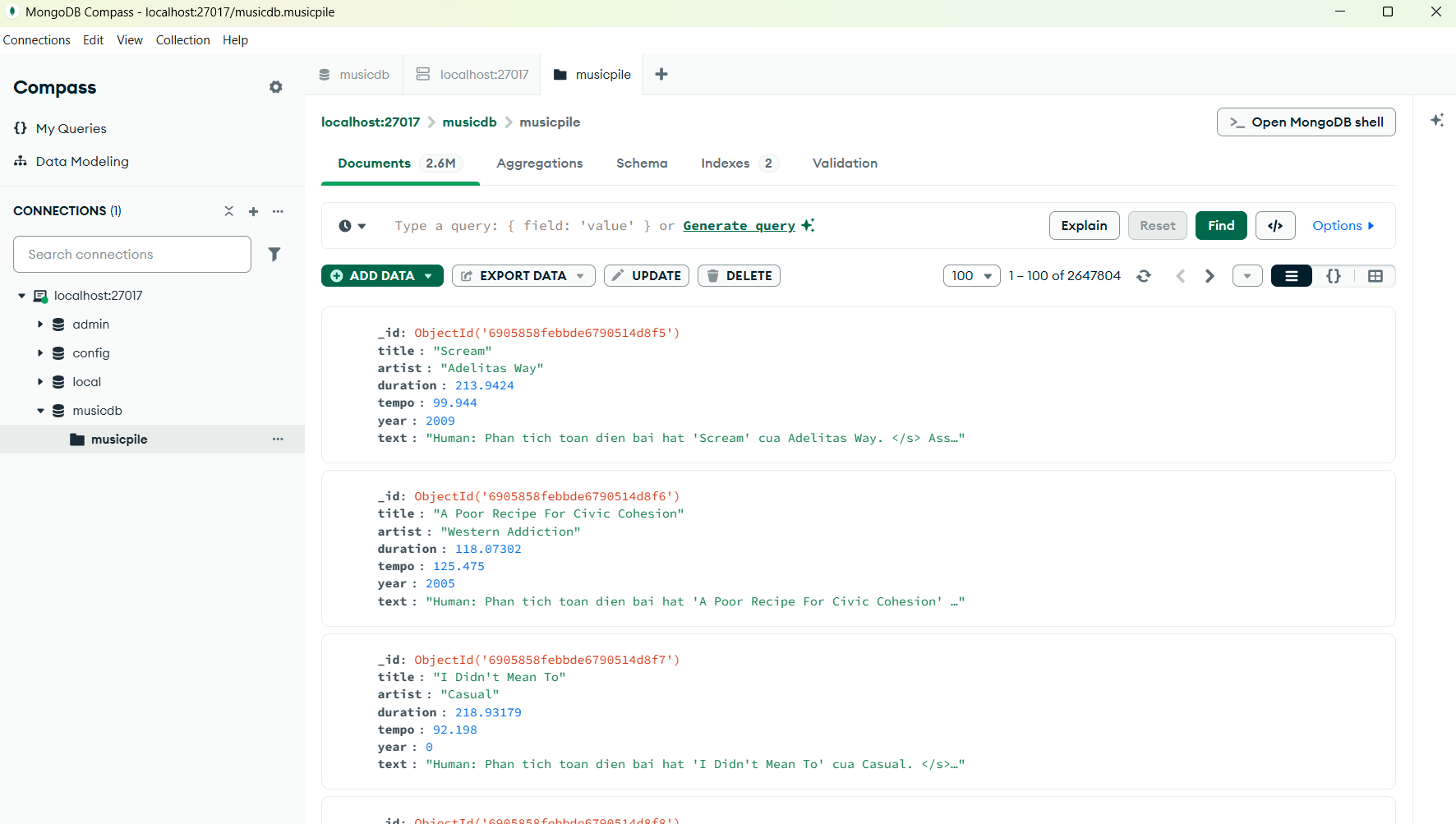
# **5. Triển khai trên MongoDB**

## 5.1. Cài đặt MongoDB

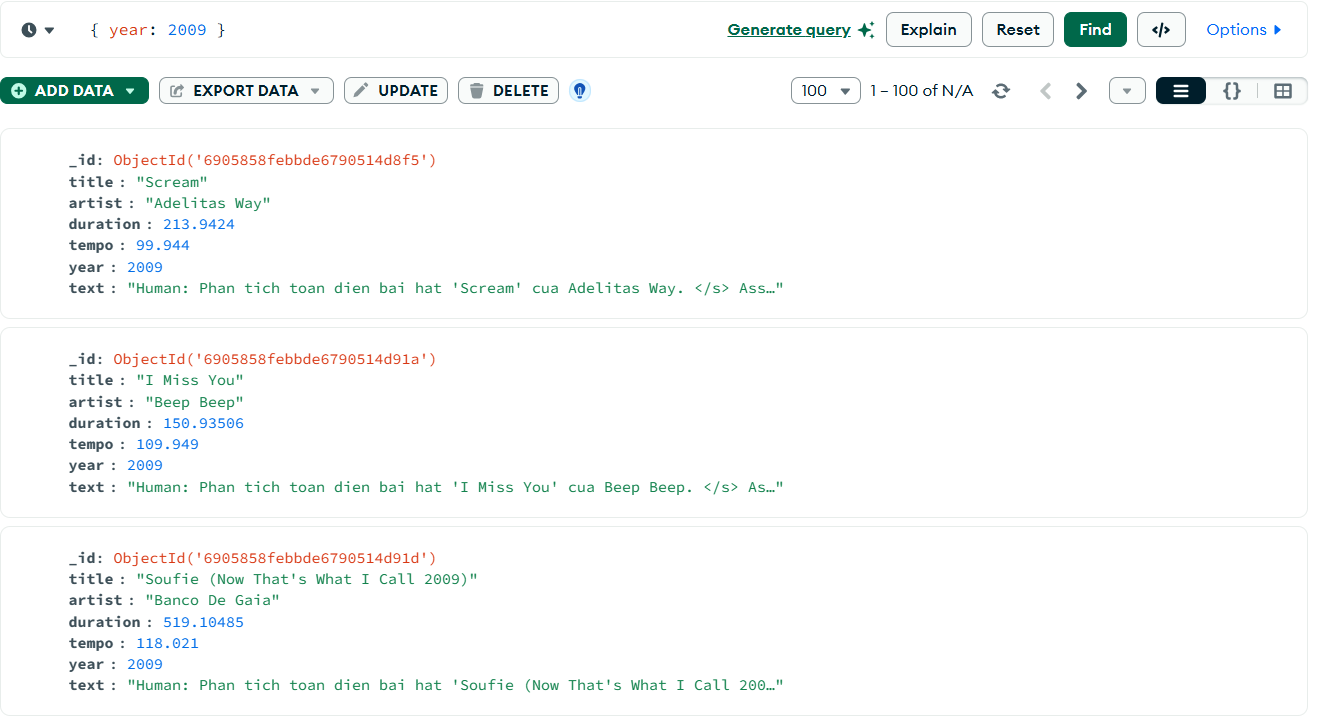
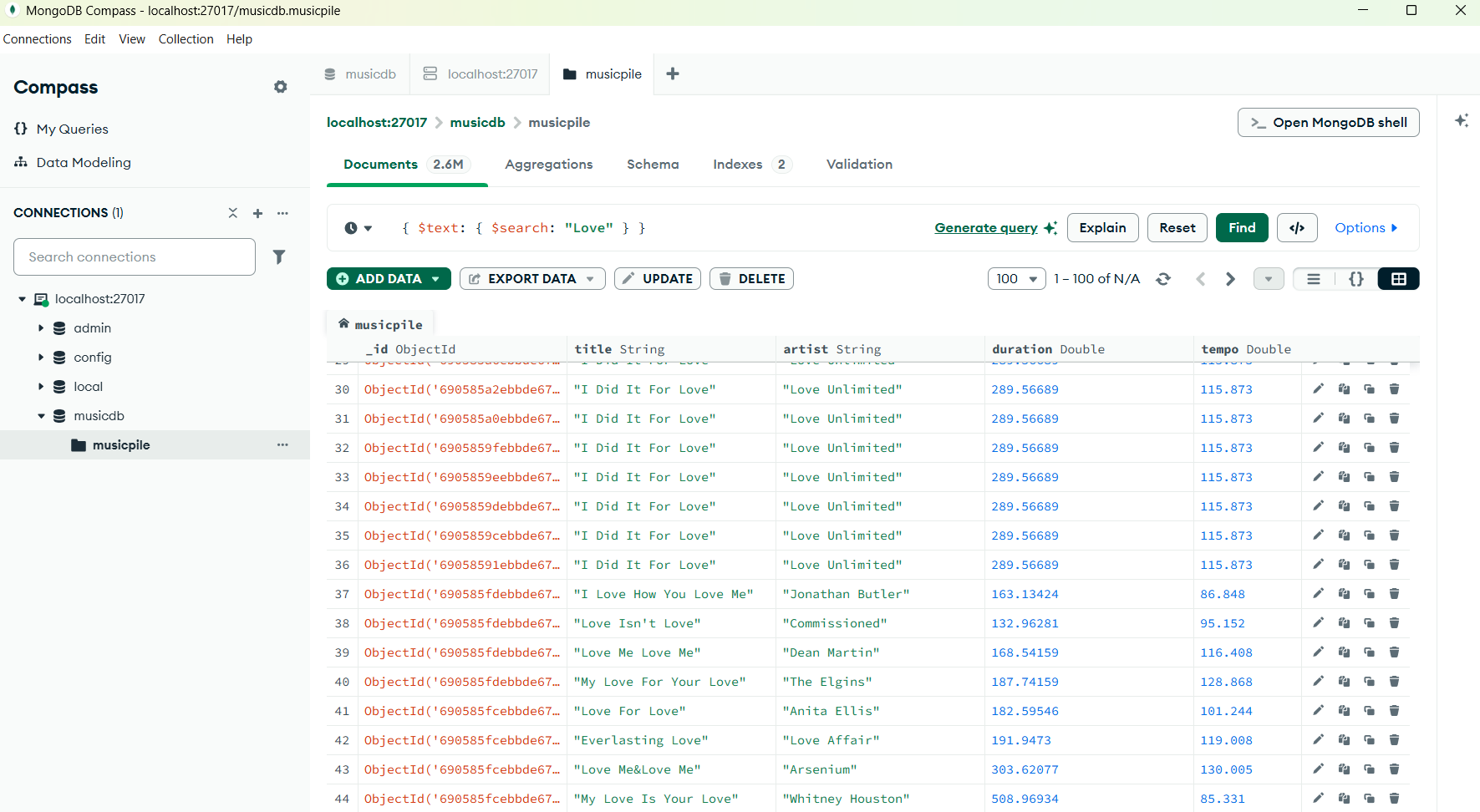
Cài đặt trên trang chính thức [MongoDB](https://www.mongodb.com/try/download/community)



## 5.2. Cấu hình kết nối và nạp dữ liệu MongoDB (Cung)

* Cấu hình 
* Đọc dữ liệu từ file Json 
* Đấy dữ liệu lên MongoDB
* Dữ liệu hiển thị trên MongoDB

## 5.3. Thực hiện truy vấn (Quốc Anh)

* Truy vấn tìm trường dữ liệu year = 2009
* Truy vấn này dùng để lọc và hiển thị các document trong collection musicpile có trường year bằng 2009. MongoDB sẽ duyệt qua toàn bộ dữ liệu và chỉ trả về các bản ghi thỏa điều kiện year = 2009.
* Truy vấn theo từ khóa
* Giải thích: Truy vấn này sử dụng Text Index trong MongoDB để tìm kiếm toàn văn (full-text search). $text cho phép tìm trong nhiều trường cùng lúc (như title, artist, lyrics, album, …) mà không cần chỉ định từng field cụ thể. $search: "Love" nghĩa là MongoDB sẽ tìm tất cả document có chứa từ “Love” ở bất kỳ trường nào được lập chỉ mục văn bản.

# **6. Thảo luận và phân tích**

## 6.1. Ưu điểm của Elasticsearch

* Elasticsearch cho phép tìm kiếm dữ liệu một cách nhanh chóng với hiệu năng cao gần như là real-time (near-realtime searching) bởi Elasticsearch được thiết kế dựa trên Apache Lucene có khả năng vượt trội trong mảng full-text research.
* Hỗ trợ tìm kiếm mờ (fuzzy search), tức là từ khóa tìm kiếm có thể bị sai lỗi chính tả hoặc không đúng cú pháp nhưng Elasticsearch vẫn có thể trả về kết quả đúng.
* Elasticsearch hỗ trợ hầu hết mọi loại dữ liệu, trừ những kiểu dữ liệu không hỗ trợ hiển thị dưới dạng văn bản (text).
* Độ trễ giữa thời gian lập chỉ mục document và tìm kiếm nó là rất ngắn (khoảng 1s), do đó, Elasticsearch rất phù hợp cho các trường hợp cần sử dụng khẩn cấp như phân tích bảo mật hoặc giám sát cơ sở hạ tầng (security analyzes và infrastructure monitoring).
* Lưu trữ dữ liệu full-text và quản lý vòng đời chỉ mục, cho phép người dùng truy xuất và phân tích, tổng hợp lượng dữ liệu rất lớn với tốc độ cao, hiệu quả hơn.
* Elasticsearch là hệ thống phân tán tự nhiên, dễ dàng mở rộng và tích hợp mạnh mẽ. Các document được lưu trữ trong Elasticsearch được phân tán trong các container khác nhau được gọi là các phân vùng (partitions), được duplicated để tích hợp với các bản sao dữ liệu trùng lặp trong trường hợp xảy ra lỗi phần cứng. Bản chất phân tán của Elasticsearch cho phép nó mở rộng hàng trăm (hàng nghìn) máy chủ và quản lý hàng petabyte dữ liệu.
* Dễ dàng phục hồi dữ liệu bằng các bản sao lưu đầy đủ được tạo bởi các khái niệm gateway trong Elasticsearch.
* Elasticsearch hoạt động như một server cloud, response format JSON và được viết bởi Java nên nó có thể hoạt động được trên nhiều nền tảng khác nhau, đồng thời tương tác, hỗ trợ nhiều ngôn ngữ lập trình như Java, JavaScript, Node.js, Go, .NET (C#), PHP, Perl, Python, Ruby. Do đó cũng là một điểm yếu khi độ bảo mật không cao.

## 6.2. Nhược điểm của Elasticsearch

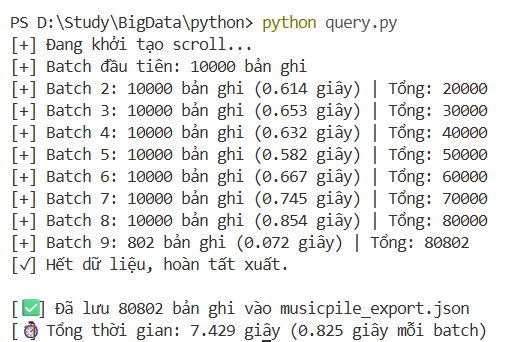
* Elasticsearch không cung cấp bất kỳ tính năng nào cho việc xác thực và phân quyền (authentication or authorization) khiến ElasticSearch kém bảo mật hơn so với các hệ quản trị cơ sở dữ liệu hiện nay.
* Elasticsearch được thiết kế cho mục đích tìm kiếm nên thường không dùng làm DB chính. Elasticsearch không mạnh trong các thao tác CRUD, nên thường sẽ dùng song song cùng một DB chính như SQL, MySQL, MongoDB, ...
* Một lý do khác không nên sử dụng Elasticsearch làm DB chính là Elasticsearch không hỗ trợ transaction (database transaction), nó sẽ không đảm bảo được toàn vẹn dữ liệu trong các hoạt động insert, update, delete dễ dẫn tới mất mát dữ liệu.
* Không phù hợp với những hệ thống thường xuyên cập nhật dữ liệu do sẽ rất tốn kém trong việc lập chỉ mục dữ liệu.
* Elasticsearch không hỗ trợ xử lý request và response bằng nhiều định dạng (chỉ dùng JSON) so với một search engine khác cũng xây dựng dựa trên Lucene như Apache Solr (hỗ trợ JSON, CSV, XML).

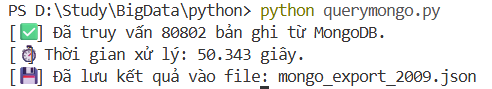
## 6.3. So sánh với MongoDB

|  |  |  |
| --- | --- | --- |
| Tiêu chí | Elasticsearch |  |
| Mục đích sử dụng | Chủ yếu dùng để tìm kiếm toàn văn (full-text) và phân tích dữ liệu theo thời gian thực, thường áp dụng trong hệ thống log hoặc giám sát dữ liệu lớn. | Được thiết kế để lưu trữ và truy vấn dữ liệu phi cấu trúc hoặc bán cấu trúc, phục vụ cho các ứng dụng web và hệ thống lưu trữ linh hoạt |
| Kiến trúc lưu trữ | Dữ liệu được tổ chức thành chỉ mục (index), chia nhỏ thành các shard để tối ưu khả năng truy xuất và mở rộng. | Sử dụng cấu trúc tài liệu JSON/BSON, các bản ghi được lưu trong collection, dễ quản lý và trực quan hơn. |
| Khả năng tìm kiếm | Hỗ trợ các truy vấn full-text mạnh mẽ, cho phép tìm kiếm với điều kiện bool, fuzzy, wildcard,… và tốc độ phản hồi rất nhanh. | Tìm kiếm văn bản ở mức cơ bản, có thể mở rộng với Atlas Search để tăng hiệu suất và độ chính xác khi truy vấn |
| Khả năng mở rộng | Tự động mở rộng quy mô nhờ cơ chế shard và khả năng phân phối dữ liệu linh hoạt giữa các node trong cluster. | Hỗ trợ mở rộng ngang (horizontal scaling) thông qua sharding, nhưng yêu cầu cấu hình thủ công và chi tiết hơn. |
| Tài nguyên phần cứng | Tiêu tốn nhiều tài nguyên, đặc biệt là RAM và CPU, do đặc thù xử lý tìm kiếm và lập chỉ mục phức tạp. | Nhẹ hơn, ít tốn tài nguyên hơn Elasticsearch, nhưng vẫn cần nâng cấp khi khối lượng dữ liệu tăng cao. |
| Độ dễ sử dụng và quản lý | Việc cài đặt và quản trị phức tạp, đòi hỏi hiểu biết chuyên sâu về cấu trúc cluster và quản lý shard. | Thân thiện hơn với lập trình viên, có giao diện trực quan và tài liệu hướng dẫn rõ ràng, dễ tiếp cận. |

## 6.4. So sánh thời gian chạy Elasticsearch và MongoDB

* Thời gian chạy Elasticsearch



* Thời gian chạy MongoDB

# **7. Kết luận**

* Elasticsearch đang trở thành một trong những công cụ được ứng dụng rộng rãi nhất trong hệ sinh thái Big Data nhờ khả năng tìm kiếm nhanh, linh hoạt và phân tích dữ liệu theo thời gian thực. Ban đầu được phát triển như một công cụ tìm kiếm toàn văn, Elasticsearch hiện nay đã vượt ra ngoài giới hạn đó để trở thành nền tảng phân tích, giám sát và xử lý luồng dữ liệu quy mô lớn, phục vụ cho các tổ chức, doanh nghiệp trong việc ra quyết định và quản trị dữ liệu hiệu quả hơn.
* Với khả năng tích hợp mạnh mẽ cùng các công nghệ xử lý dữ liệu như Apache Kafka, Apache Spark hay Hadoop, Elasticsearch cho phép xây dựng các hệ thống phân tích dữ liệu theo thời gian thực, phát hiện xu hướng, mối quan hệ và bất thường trong các nguồn dữ liệu khổng lồ. Việc kết hợp này giúp nâng cao tốc độ xử lý, giảm độ trễ trong phân tích và tạo ra khả năng phản ứng nhanh trước các thay đổi trong dữ liệu. Bên cạnh đó, sự hỗ trợ của Machine Learning và khả năng kết nối với các mô hình AI giúp Elasticsearch mở rộng phạm vi ứng dụng sang các lĩnh vực như tài chính, y tế, thương mại điện tử, sản xuất công nghiệp và Internet of Things (IoT).
* Không chỉ dừng lại ở dữ liệu văn bản, Elasticsearch còn có khả năng tìm kiếm và phân tích dữ liệu phi cấu trúc, bao gồm hình ảnh, âm thanh và video, khi được kết hợp với các công nghệ xử lý ngôn ngữ tự nhiên (NLP) và học sâu (Deep Learning). Nhờ đó, hệ thống có thể thực hiện các tìm kiếm ngữ nghĩa (semantic search), phân tích nội dung, phát hiện chủ đề và cải thiện trải nghiệm người dùng trong nhiều ứng dụng khác nhau.
* Nhìn chung, Elasticsearch đang khẳng định vị thế của mình như một nền tảng toàn diện trong kỷ nguyên dữ liệu lớn, nơi tìm kiếm, phân tích và trí tuệ nhân tạo được hợp nhất thành một hệ thống thống nhất. Với khả năng mở rộng linh hoạt, tốc độ xử lý cao và khả năng tích hợp đa dạng, Elasticsearch hứa hẹn sẽ tiếp tục đóng vai trò quan trọng trong việc khai thác giá trị dữ liệu, hỗ trợ ra quyết định chiến lược và thúc đẩy đổi mới trong nhiều lĩnh vực của đời sống số hiện nay.

# **8. Tài liệu tham khảo**

Tìm hiểu chung về elasticsearch - Vibo <https://viblo.asia/p/tim-hieu-chung-ve-elasticsearch-GrLZDkg3Kk0>

https://anonystick.com/blog-developer/elasticsearch-la-gi-mo-hinh-dac-diem-va-cai-dat-elasticsearch-2020060249015172#t-4

Tìm hiểu về mongoDB <https://topdev.vn/blog/mongodb-la-gi/>