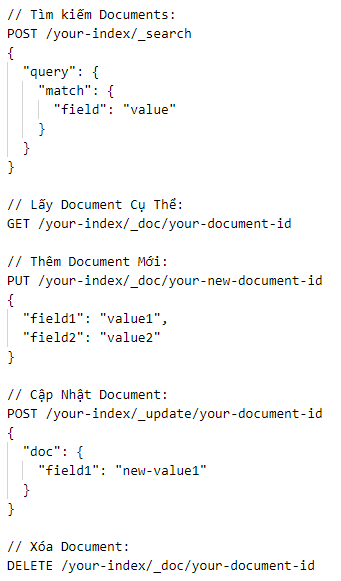
**ELASTICSEARCH**

* Elasticsearch là một Search Engine xây dựng trên nền tảng Apache Lucene, tương tự như mối quan hệ giữa PHP và Laravel.
* Nó chạy trên server độc lập và giao tiếp qua RESTful, giúp tích hợp dễ dàng với bất kỳ hệ thống hay ngôn ngữ lập trình nào thông qua HTTP requests.
* Những đặc điểm quan trọng của Elasticsearch:
  + *Phân tán và Mở rộng*: Elasticsearch là hệ thống phân tán và có khả năng mở rộng ngang tuyệt vời. Việc thêm node tự động mở rộng hệ thống.
  + *Open Source và Java-based*: Elasticsearch là dự án nguồn mở, được phát triển bằng Java và được phát hành theo giấy phép Apache.
  + *JSON-based Data*: Sử dụng cơ sở dữ liệu NoSQL để lưu trữ và truy vấn dữ liệu, tập trung vào tối ưu hóa hiệu suất tìm kiếm. Hỗ trợ tìm kiếm chính xác gần thời gian thực trên dữ liệu lớn.
  + *RESTful APIs*: Cung cấp giao diện RESTful API giúp ứng dụng và dịch vụ tương tác dễ dàng với Elasticsearch, tối ưu cho việc tích hợp vào các ứng dụng hiện có.



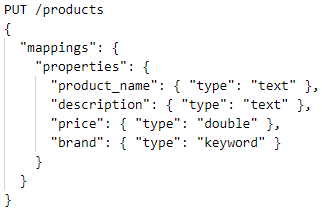
* + *Multi Data Resources*: Hỗ trợ lấy dữ liệu từ nhiều nguồn khác nhau như logs, metrics, và dữ liệu đa dạng từ các ứng dụng khác nhau.

**Cách Elasticsearch tổ chức lưu trữ dữ liệu**

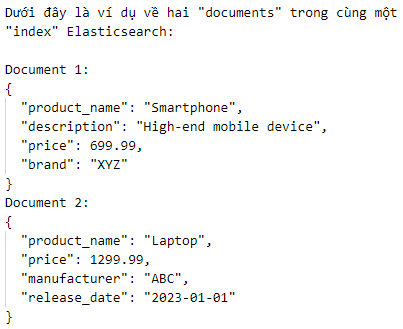
* Cấu trúc lưu trữ dữ liệu được tổ chức logic hoá có nét tương đồng với hệ cơ sở dữ liệu quan hệ RDMS với các thành phần như sau:

|  |  |
| --- | --- |
| **RDMS** | **Elasticsearch** |
| Database | Cluster |
| Table | Index/Indicies |
| Row | Document |
| Column | Field |

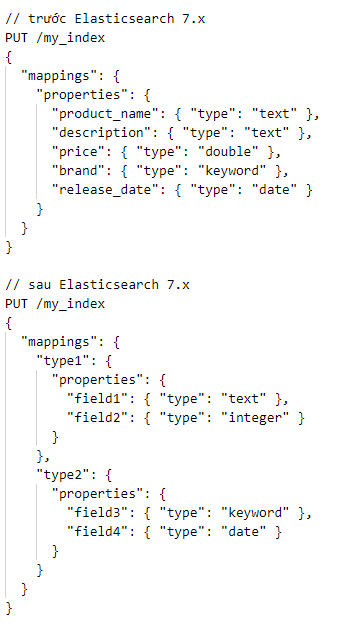
* **Indexes**
  + "index" là một tập hợp các "documents" có liên quan
  + Đặc điểm: Một cơ sở dữ liệu có thể bao gồm nhiều index. Mỗi "index" trong Elasticsearch có một tên duy nhất để phân biệt với các "index" khác
  + Mapping: Mỗi "index" có một "mapping" mô tả cấu trúc dữ liệu của các "documents" trong "index". "Mapping" này xác định các trường (fields), kiểu dữ liệu, và các thuộc tính khác của dữ liệu



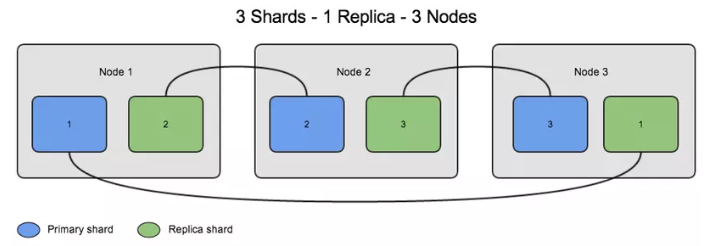
* + Tùy chọn Cấu trúc: "Index" không yêu cầu cấu trúc dữ liệu giống nhau giữa các "documents". Mỗi "document" trong "index" có thể có cấu trúc dữ liệu riêng
* **Documents**
  + Trong Elasticsearch, mô hình dữ liệu được tổ chức thành các "documents", và "documents" này được nhóm lại trong các "indices".
  + Mỗi "document" trong Elasticsearch là một đối tượng JSON, trong đó có nhiều "fields" chứa dữ liệu
  + Các "documents" trong một "index" không cần phải có cấu trúc dữ liệu giống nhau, có thể chứa các "fields" khác nhau. Điều này là một khác biệt quan trọng so với cơ sở dữ liệu quan hệ, nơi mà mỗi bảng có cấu trúc cố định:



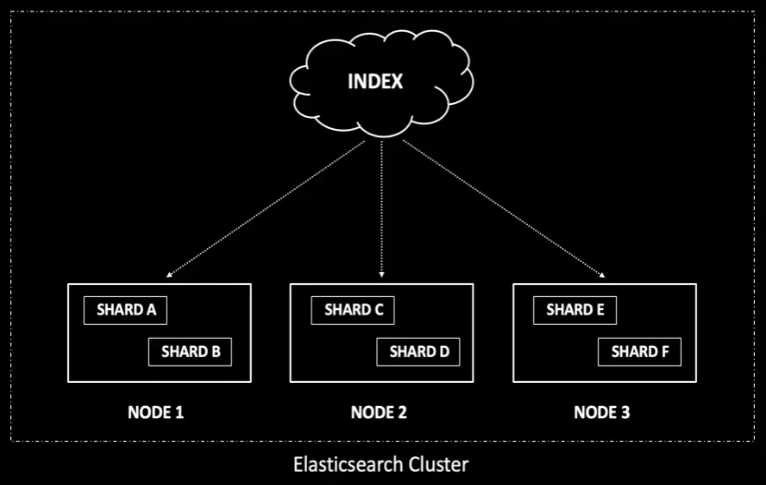
* **Fields**
  + Mỗi "field" trong Elasticsearch đại diện cho một thuộc tính cụ thể của mỗi "document" và chúng được đặt tên và định kiểu trong "mapping" của "index"
* **Types**
  + Được sử dụng làm danh mục của index, cho phép lưu trữ các loại dữ liệu khác nhau trong cùng một index
  + Trong phiên bản Elasticsearch 7.x trở đi, khái niệm "type" đã thay đổi. Trước đó, mỗi "index" có thể chứa nhiều "types", nhưng từ Elasticsearch 7.x trở đi, chỉ có một "type" cho mỗi "index". Tính năng "type" được loại bỏ trong các phiên bản mới và sẽ không được hỗ trợ trong tương lai.



* + Elasticsearch sử dụng inverted index để đánh chỉ mục cho các document
* **Shard**
  + Shard là tập con các Document của 1 Index . Một Index có thể được chia thành nhiều shard.
  + Chúng ta gần như không bao giờ làm việc trực tiếp với các Shard vì Elasticsearch đã support toàn bộ việc giao tiếp cũng như tự động thay đổi các Shard khi cần thiết.
  + Có 2 loại Shard là : Primary Shard và Replica Shard.
    - *Primary Shard*
      * Primary Shard được lưu trữ dữ liệu và đánh Index, sau đó 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 1 Replica Shard.
    - *Replica Shard*
      * Replica Shard 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. Đó cũng là lý do vì sao Replica Shard không được phân bố trên cùng một node với Primary Shard vì nếu cả 2 nằm trên cùng một node và node đó gặp sự cố, thì không có lợi ích gì từ việc có các bản sao (replica) nữa.



* + - * Tăng cường tốc độ tìm kiếm bởi có thể setup lượng Replica Shard nhiều hơn mặc định của Elasticsearch hoặc thực hiện thao tác tìm kiếm song song trong các bản sao này.
* **Cluster**
  + "Cluster" là tập hợp các nodes hoạt động cùng với nhau, chia sẽ cùng thuộc tính cluster.name. Chính vì thế Cluster sẽ được xác định bằng 1 '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.
  + Một cluster có thể gồm 1 hoặc nhiều nodes. Các nodes có thể hoạt động trên cùng 1 server . Tuy nhiên trong thực tế , một cluster sẽ gồm nhiều nodes hoạt động trên các server khác nhau để đảm bảo nếu 1 server gặp sự cố thì server khác (node khác) có thể hoạt động đầy đủ chức năng.
  + Các node có thể tìm thấy nhau để hoạt động trên cùng 1 cluster qua giao thức unicast



**Node**

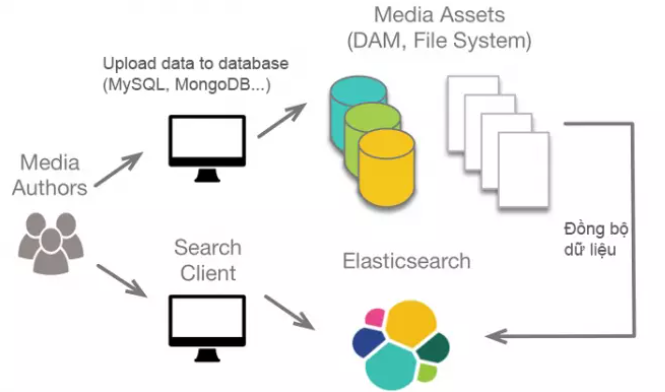
* + Node được coi là bộ phận “đầu não” của Elasticsearch. Đây là nơi lưu trữ dữ liệu trực tiếp, tham gia thực hiện đánh Index của Cluster và thực hiện các thao tác tìm kiếm dữ liệu.
  + Mỗi node được định danh bằng 1 unique name

**Ưu nhược điểm của ES**

* 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.
* Có khả năng mở rộng dựa theo chiều ngang rất tuyệt vời.
* Hỗ trợ Structured Query DSL (Domain-Specific Language ), cung cấp việc đặc tả những câu truy vấn phức tạp một cách cụ thể và rõ ràng bằng JSON. <https://viblo.asia/p/query-dsl-trong-elasticsearch-Eb85oJq2l2G>
* Hỗ trợ nhiều Elasticsearc client như Java (Elasticsearch High Level REST Client), PhP, Javascript (Elasticsearch.js), Ruby, .NET, Python

**Nhược điểm**

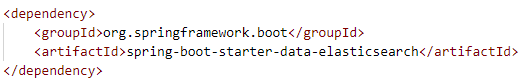
* Elasticsearch được thiết kế cho mục đích tìm kiếm nên thường không dùng làm DB chính. Elasticsearch không mạnh trong các thao tác CRUD, nên thường sẽ dùng song song cùng một DB chính như SQL, MySQL, MongoDB, ...



* 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).

**Demo**

* Depedency:



* Link: https://viblo.asia/p/tao-mot-project-voi-spring-boot-va-elasticsearch-841-su-dung-thu-vien-spring-data-elasticsearch-huong-dan-cai-dat-va-su-khac-nhau-giua-2-phien-ban-8xx-voi-7xx-elasticsearch-Rk74aRXvJeO