**I.Giới thiệu**

Elasticsearch ra mắt năm 2010, được phát triển bởi ***Shay Banon*** bằng ngôn ngữ ***Java***, là mã nguồn mở theo giấy phép Apache 2.0 và phân tán thời gian thực. Kể từ khi phát hành thì Elasticsearch đã nhanh chóng trở thành một trong những công cụ tìm kiếm được sử dụng thông dụng nhất và được sử dụng rộng rãi trong những trường hợp liên quan đến phân tích nhật ký và tìm kiếm văn bản, thông tin bảo mật và phân tích nghiệp vụ cũng như thông tin vận hành.

 Elasticsearch là một công cụ tìm kiếm và phân tích phân tán, là một RESTful mã nguồn mở được xây dựng trên Apache Lucene. Với nhiều ***REST API*** đơn giản có khả năng phân tích và lưu trữ tất cả các dữ liệu có dạng như textextual, digital, geospatial, structured và unstructured

Thực chất hoạt động như 1 web server, có khả năng tìm kiếm nhanh chóng (near realtime) thông qua giao thức RESTful

* Elasticsearch có khả năng phân tích và thống kê dữ liệu
* Elasticsearch  có thể tích hợp nó vào hệ thống bạn là dễ dàng, bạn chỉ cần gửi request http lên là nó trả về kết quả.
* Elasticsearch là 1 hệ thống phân tán và có khả năng mở rộng tuyệt vời (horizontal scalability). Chỉ cần lắp thêm node là có thể tự động mở rộng
* Elasticsearch  được phát triển bằng Java

Elasticsearch là một trong những công cụ tìm kiếm doanh nghiệp rất phổ biến và hiện đang được sử dụng bởi nhiều tổ chức lớn như Wikipedia, The Guardian, StackOverflow, GitHub, … 

Elasticsearch là 1 server riêng biệt để “phục vụ” việc tìm kiếm dữ liệu. ES sẽ chạy một cổng (local default là 9200).

Elasticsearch có thể mở rộng lên tới hàng petabyte dữ liệu dạng có cấu trúc và không cấu trúc. Nó cũng có thể được sử dụng thay thế cho các DB lưu trữ dữ liệu document như MongoDB hay RavenDB. Sử dụng tính không chuẩn hóa để cải thiện hiệu suất tìm kiếm. Nhưng do ES không mạnh trong các thao tác CRUD(create - tạo mới, read - đọc, update - cập nhật và delete - xóa), nên thường sẽ dùng song song với 1 DB chính (SQL, MySQL, MongoDB …)

Elasticsearch không sử dụng tìm kiếm đơn thuần toàn văn bản không sử dụng index, nghĩa là tập dữ liệu càng lớn thì tìm kiếm càng lâu, trong khi ES lại “đánh index” cho các trường được chọn để tìm kiếm.

Chúng ta nên sử dụng Elasticsearch trong những trường hợp sau:

* Searching for pure text (textual search): tìm kiếm text thông thường.
* Searching text and structured data (product search by name + properties): tìm kiếm text và dữ liệu có cấu trúc.
* Data aggregation, security analytics, analysis of business data: tổng hợp dữ liệu, phân tích bảo mật, phân tích dữ liệu kinh doanh, lưu trữ số lượng dữ liệu lớn.
* Logging and log analytics: ghi lại quá trình hoạt động và phân tích nó.
* Application performance monitoring: giám sát hiệu năng ứng dụng.
* Infrastructure indicators and container monitoring.
* Geo Search: tìm kiếm theo tọa độ, phân tích và trực quan hóa dữ liệu không gian địa lý.
* JSON document storage: lưu trữ dữ liệu dạng JSON.

**II.Các khái niệm cơ bản**

### **1. Document**

Document là một JSON object với một số dữ liệu. Đây là basic information unit trong ES. Hiểu 1 cách cơ bản thì đây là đơn vị nhỏ nhất để lưu trữ dữ liệu trong Elasticsearch.

### **2. Shard**

* Shard là đối tượng của Lucene , là tập con các documents của 1 Index. Một Index có thể được chia thành nhiều shard.
* Mỗi node bao gồm nhiều Shard . Chính vì thế Shard mà là đối tượng nhỏ nhất, hoạt động ở mức thấp nhất, đóng vai trò lưu trữ dữ liệu.
* Chúng ta gần như không bao giờ làm việc trực tiếp với các Shard vì Elasticsearch đã hỗ trợ toàn bộ việc giao tiếp cũng như tự động thay đổi các Shard khi cần thiết.

Shard rất quan trọng vì:

* Cho phép phân mảnh theo chiều ngang mở rộng khối lượng bản ghi
* Cho phép phân tán và hoạt động song song trên các phân đoạn, nhờ đó tăng hiệu suất làm việc.

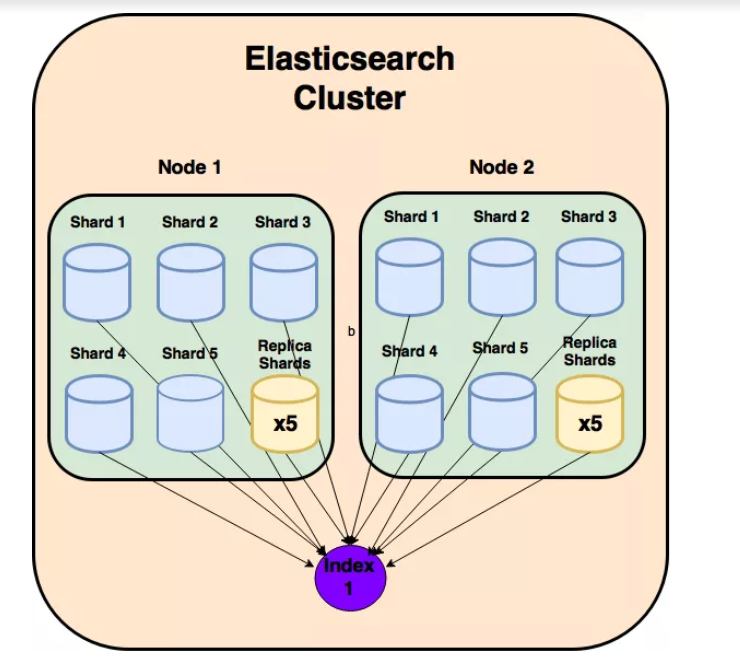
Có 2 loại Shard là : primary shard và replica shard.

#### **2.1 : Primary Shard**

* Primary Shard là sẽ lưu trữ dữ liệu và đánh index, sau đó các Primary Shard là phần nằm ngang gốc của một index và sau đó các primary shard này được sao chép thành các Replicas Shard.
* Primary Shard là sẽ 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 mỗi Primary Shard thì sẽ đi kèm 1 Replica Shard.

#### **2.2 : Replica Shard**

* Replica Shard đúng như cái tên của nó, nó là nơi lưu trữ dữ liệu nhân bản của Primary Shard khả năng sẵn sàng cao, thay thế Primary Shard khi có lỗi. Đó cũng là lý do vì so Replica Shard không được phân bố trên cùng một node với Primary Shard mà chỉ được sao chép từ nó.
* Replica Shard có vai trò đảm bảo tính toàn vẹn của dữ liệu khi Primary Shardxảy 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 ES



### **3. Index**

Nó 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.

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.

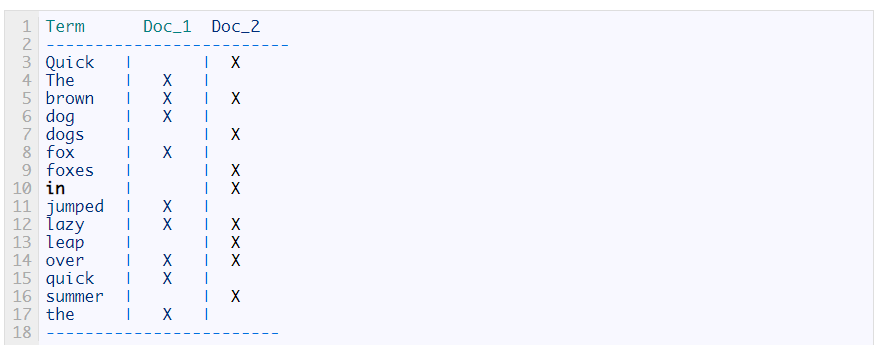
Trong Elasticsearch , sử dụng một cấu trúc được gọi là inverted index . Nó được thiết kế để cho phép tìm kiếm full-text search. Cách thức của nó khá đơn giản, các văn bản được phân tách ra thành từng từ có nghĩa sau đó sẽ được ánh xạ xem thuộc văn bản nào nhằm mục đích tạo mối liên kết giữa các từ và các document chứa từ đó. Khi tìm kiếm tùy thuộc vào dạng tìm kiếm sẽ đưa ra kết quả cụ thể.

VÍ dụ : Chúng ta có 2 văn bản cụ thể như sau :

1,The quick brown fox jumped over the lazy dog

2,Quick brown foxes leap over lazy dogs in summer

Để tạo ra một inverted index, trước hết chúng ta sẽ phân chia nội dung của từng tài liệu thành các từ riêng biệt (chúng tôi gọi là terms), tạo một danh sách được sắp xếp của tất cả terms duy nhất, sau đó liệt kê tài liệu nào mà mỗi thuật ngữ xuất hiện. Kết quả như sau:



Bây giờ, nếu chúng ta muốn tìm kiếm màu quick brown, chúng ta chỉ cần tìm trong các tài liệu trong đó mỗi thuật ngữ có xuất xuất hiện hay không. Kết quả như sau:



Như các bạn đã thấy, cả 2 đoạn văn bản đều thích hợp với từ khóa. Tuy nhiên có thể dễ dàng nhận ra rằng Doc\_1 chính xác hơn nhiều

### **4. Node**

* Là trung tâm hoạt động của Elasticsearch. Là nơi lưu trữ dữ liệu ,tham gia thực hiện đánh index của cluster cũng như thực hiện các thao tác 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.

**5. Cluster**

* 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’. Việc định danh các cluster trùng tên sẽ gây nên lỗi cho các node vì vậy khi setup cần hết sức chú ý điểm này
* Mỗi cluster có một node chính (master), được lựa chọn một cách tự động và có thể thay thế nếu sự cố xảy ra. 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 so với khi có 2 servers. Các node có thể tìm thấy nhau để hoạt động trên cùng 1 cluster qua giao thức unicast. 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.

**III.Ưu nhược điểm của ES**

### **Ưu điểm**

* 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.
* Có khả năng phân tích dữ liệu (Analysis data)
* Khả năng mở rộng theo chiều ngang tuyệt vời
* Hỗ trợ tìm kiếm mờ (fuzzy), tức là từ khóa tìm kiếm có thể bị sai lỗi chính tả hay không đúng cú pháp thì vẫn có khả năng elasticsearch trả về kết quả tốt.
* 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.
* Hỗ trợ nhiều Elasticsearc client như Java, PhP, Javascript, Ruby, .NET, Python

### **Nhược điểm**

* 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 search, do vậy với những nhiệm vụ khác ngoài search như CRUD thì elastic kém thế hơn so với những database khác như Mongodb, Mysql …. Do vậy người ta ít khi dùng elasticsearch làm database chính, mà thường kết hợp nó với một database khác.
* Trong elasticsearch không có khái niệm database transaction , tức là nó sẽ không đảm bảo được toàn vẹn dữ liệu trong các hoạt độngInsert, Update, Delete.Tức khi chúng ta thực hiện thay đổi nhiều bản ghi nếu xảy ra lỗi thì sẽ làm cho logic của mình bị sai hay dẫn tới mất mát dữ liệu. Đây cũng là 1 phần khiến elasticsearch không nên là database chính.
* Không thích hợp với những hệ thống thường xuyên cập nhật dữ liệu. Sẽ rất tốn kém cho việc đánh index 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).