Search Engines

Have you ever been looking for something but didn't know where to find it? There comes the Search engine. Search engine is a software system that is designed to search for information. We search for some information and the results will be returned.

Which are the top few search engines in the world?

1. Google - No need for further introductions. The search engine giant holds the first place in search
2. Bing - Bing is Microsoft's attempt to challenge Google in the area of search
3. Yahoo - Yahoo is still the most popular email provider and according to reports holds the fourth place in search
4. Ask.com
5. AOL.com

What is the process of a search engine?

The repository of web pages is referred to as the 'Index', and it is this data store which is organized and used to provide the search results you see on the search engine. Indexing is the process of organizing the masses of data and pages so they can be searched quickly for relevant results to your search query.

How can we implement a search engine for our organization?

Elasticsearch is a powerful tool for implementing search engine, both on premise and on cloud. Elasticsearch is a distributed, RESTful search and analytics engine capable of solving a growing number of use cases. As the heart of the Elastic Stack, it centrally stores your data so you can discover the expected and uncover the unexpected.

Elasticsearch is a highly scalable open-source full-text search and analytics engine. It allows you to store, search, and analyze big volumes of data quickly and in near real time. It is generally used as the underlying engine/technology that powers applications that have complex search features and requirements.

Elasticsearch lets you perform and combine many types of searches — structured, unstructured, geo, metric — any way you want.

Here are a few sample use-cases that Elasticsearch could be used for:

You run an online web store where you allow your customers to search for products that you sell. In this case, you can use Elasticsearch to store your entire product catalog and inventory and provide search and autocomplete suggestions for them.

You want to collect log or transaction data and you want to analyze and mine this data to look for trends, statistics, summarizations, or anomalies. In this case, you can use Logstash (part of the Elasticsearch/Logstash/Kibana stack) to collect, aggregate, and parse your data, and then have Logstash feed this data into Elasticsearch. Once the data is in Elasticsearch, you can run searches and aggregations to mine any information that is of interest to you.

You run a price alerting platform which allows price-savvy customers to specify a rule like "I am interested in buying a specific electronic gadget and I want to be notified if the price of gadget falls below $X from any vendor within the next month". In this case you can scrape vendor prices, push them into Elasticsearch and use its reverse-search (Percolator) capability to match price movements against customer queries and eventually push the alerts out to the customer once matches are found.

You have analytics/business-intelligence needs and want to quickly investigate, analyze, visualize, and ask ad-hoc questions on a lot of data (think millions or billions of records). In this case, you can use Elasticsearch to store your data and then use Kibana (part of the Elasticsearch/Logstash/Kibana stack) to build custom dashboards that can visualize aspects of your data that are important to you. Additionally, you can use the Elasticsearch aggregations functionality to perform complex business intelligence queries against your data.

Elasticsearch is a near real time search platform. What this means is there is a slight latency (normally one second) from the time you index a document until the time it becomes searchable.

Basic concepts in Elasticsearch

Cluster: A cluster is a collection of one or more nodes (servers) that together holds your entire data and provides federated indexing and search capabilities across all nodes. A cluster is identified by a unique name which by default is "elasticsearch". This name is important because a node can only be part of a cluster if the node is set up to join the cluster by its name.

Make sure that you don't reuse the same cluster names in different environments, otherwise you might end up with nodes joining the wrong cluster

Node: A node is a single server that is part of your cluster, stores your data, and participates in the cluster's indexing and search capabilities. Just like a cluster, a node is identified by a name which by default is a random Universally Unique IDentifier (UUID) that is assigned to the node at startup. A node can be configured to join a specific cluster by the cluster name. In a single cluster, you can have as many nodes as you want.

Index: An index is a collection of documents that have somewhat similar characteristics. For example, you can have an index for customer data, another index for a product catalog, and yet another index for order data. Index is equivalent to database in the RDBMS world.

Type: A type used to be a logical category/partition of your index to allow you to store different types of documents in the same index. Type is equivalent to table in the RDBMS world.

Document: A document is a basic unit of information that can be indexed. Document is equivalent to record in RDBMS world.

Shards & Replicas: An index can potentially store a large amount of data that can exceed the hardware limits of a single node. For example, a single index of a billion documents taking up 1TB of disk space may not fit on the disk of a single node or may be too slow to serve search requests from a single node alone.

To solve this problem, Elasticsearch provides the ability to subdivide your index into multiple pieces called shards. When you create an index, you can simply define the number of shards that you want. Each shard is in itself a fully-functional and independent "index" that can be hosted on any node in the cluster.

In a network/cloud environment where failures can be expected anytime, it is very useful and highly recommended to have a failover mechanism in case a shard/node somehow goes offline or disappears for whatever reason. To this end, Elasticsearch allows you to make one or more copies of your index's shards into what are called replica shards, or replicas for short.