

|  |
| --- |
| Introduction to Elasticsearch |
|  |
| February 2  A joint paper by JP Tokyo & Co and MRIIRS  Authored by: Stuti Mehra Reviewed by: Rajesh Nath, Ms. Charu Virmani, Ms. Bindiya Ahuja |

Contents

[Introduction 3](#_Toc527993527)

[What is Elasticsearch? 3](#_Toc527993528)

How to communicate with Elasticsearch?...................................................................3

Java API………………………………………………………………………………………………………………

Node Client……………………………………………………………………………………………

Transport Client……………………………………………………………………………………

RESTful API……………………………………………………………………………………………………..

Basic Terms………………………………………………………………………………………………………………3

Cluster…………………………………………………………………………………………………………………3

Indexing………………………………………………………………………………………………………………4

Type…………………………………………………………

# Introduction

|  |
| --- |
| What is Elasticsearch?? Elasticsearch is an open source distributed, RESTful search and analytics engine which is capable of solving huge number of use cases. It is highly scalable open-source full-text search and analytics engine, which allows you to store, search and analyze large volumes of data very quicky and in near real time. It is a real-time distributed search and analytics engine. It is used for four kinds or type of search namely, full-text search, structured search, analytics and combination of these three searches. For instance:   * Wikipedia uses Elasticsearch to give full-text search with highlighted snippets and *search-as-you-type* and *did-you-mean* suggestions. * Stack Overfllow combines full-text search with geolocation queries and uses *more-like-this* to find related questions and answers. * Github uses Elasticsearch to query 130 billion lines of code.   Elasticsearch is also written in JAVA and uses Lucene internally for its indexing and searching. It uses different operations like indexing, searching and modifying the data.  **How to communicate with Elasticsearch?**  Elasticsearch uses two types of API, namely JAVA API and RESTful API. It depends on which kind of language you are using, if you are using JAVA language then you can use JAVA API and if you want to choose any other language then you can use RESTful API.  JAVA API :-  If you are using JAVA language then, Elasticsearch have two uilt-in clientsthat we can use in our code:  *Node Client:*  This is a type of node which doesn’t holds the data itself but, it knows that what kind of data is on which node in the cluster, and can forward requests directly to correct node.  *Transport Client:*  This is a light weight transport client and can be used to send the requests to remote cluster. It don’t join the cluster itself but just simply forwards the requests to a node in the cluster.  Both Java clients talk to the cluster over 9300, using native Elasticsearch transport protocol. Nodes in the cluster also communicate with each other using port 9300, if this port is not open the, your nodes would not be able to form a cluster.  RESTful API  All other languages can communicate with Elasticsearch over 9200 using a RESTful API, accessible worth our web client. We can talk with Elasticsearch using a curl command.  A request to Elasticsearch consists of the same parts as any HTTP request:  curl -X<VERB> '<PROTOCOL>://<HOST>/<PATH>?<QUERY\_STRING>' -d '<BODY>'  VERB: It is the appropriate HTTP method or verb like GET, POST, PUT, HEAD or DELETE.  PROTOCOL: It means the protocol which we are using it could be http or https.  HOST: it is te hostname of any node in your elasticsearch cluster, or localhost for a node in a local machine.  PORT: The port which is running the Elasticsearch HTTP service, whics is by default 9200.  QUERY\_STRING: It contains any optional or additional query string parameters.  **Basic terms:**   1. ***Cluster:***   A node is a running instance of Elasticsearch, while the cluster consists of one or more nodes with the same cluster.name which are working together to share their data and workload. It is a group of one or more nodes instances that are connected together. There are different kinds of nodes in an elasticsearch cluster which ca nodes, master nodes n be assigned different roles and responsibilities. They are namely data nodes, master nodes, client nodes, ingest nodes.   1. ***Indexing:***   Indexing is the act of storing data in the elasticsearch. A cluster contains multiple indexes in it. Indexing helps us to search data quickly and efficiently. If indexing concept would not be there then we have to search the entire catalog for getting the search result which we want. It helps to index the data in the catalog, so that data is organized and easily we are able to get the result.   1. ***Type:***   In applications we use objects to represent things like a user, comment or an email. Each object belongs to a class that defines the properties or data which is linked or associated with an object. Objects in the user class may have a name, gender, age and an email address. |
| “Indexing speeds up the search process and information retrieval process.” |
| **MongoDB**  MongoDB is a document database wth scalability and flexibility. It stores data in flexible, JSON-like documents meaning fields may vary from document to document and data structure can be change anytime. It documents model maps to the objects in our application code which makes data easy to work with. Ad hoc queries, indexing and real time aggregation provide us powerful ways to access and analyze our data. It is free and open source which is easily accessible to all the people.  Node.js  Node.js is an open-source, cross-platform JavaScript run-time environment that executes JavaScript code outside of a browser. JavaScript is used primarily for client-side scripting, in which scripts written in JavaScript are embedded in a webpage's HTML and run client-side by a JavaScript engine in the user's web browser.  **NPM**  NPM (short for Node.js package manager) is a package manager for the JavaScript programming language. It is the default package manager for the JavaScript runtime environment Node.js. It consists of a command line client, also called npm, and an online database of public and paid-for private packages, called the npm registry.  **JSON**  Elasticsearch uses JavaScrpt Object Notation, or JSON, as the serialization format for documents. JSON serialization is supported by most programming language and has become the standard format used by the NOSQL movement. It is simple concise and easy to read.  **Why Elasticsearch??**  Elasticsearch is a NOSQL, distributed full text database which means that ths database is document based instead of using tables or schema, we use documents, many documents.  Elasticsearch has it’s own Query Domain Specific Language, where you specify the query in JSON format. You can also nest other queries based on your need. Real projects require search on different fields by applying some conditions, different weights, recent documents, values of some predefined fields and so on. All such complexity can be expressed through a single query. The query DSL is powerful and designed to handle the real world query complexity through a single query. Elasticsearch APIs are directly related to Lucene and it is using the same as Lucene operations name. Query DSL also using the Lucene TermQuery to execute it.  **Index a document:**  PUT /myindex/employee/1  {  "first\_name" : "John",  "last\_name" : "Smith",  "age" : 25,  "about" : "Indexing practice",  "interests": [ "sports", "music" ]  } |

Retrieving a document:

{

"\_index" : "myindex",

"\_type" : "employee",

"\_id" : "1",

"\_version" : 1,

"found" : true,

"\_source" : {

"first\_name" : "Tom",

"last\_name" : "Troy",

"age" : 25,

"about" : "I love to reading novels",

"interests": [ "sports", "music" ]

}

}

It is displaying a simple search

GET {

"took": 6,

"timed\_out": false,

"\_shards": { ... },

"hits": {

"total": 3,

"max\_score": 1,

"hits": [

{

"\_index": "megacorp",

"\_type": "employee",

"\_id": "3",

"\_score": 1,

"\_source": {

"first\_name": "Douglas",

"last\_name": "Fir",

"age": 35,

"about": "I like to build cabinets",

"interests": [ "forestry" ]

}

},

{

"\_index": "megacorp",

"\_type": "employee",

"\_id": "1",

"\_score": 1,

"\_source": {

"first\_name": "John",

"last\_name": "Smith",

"age": 25,

"about": "I love to go rock climbing",

"interests": [ "sports", "music" ]

}

},

{

"\_index": "megacorp",

"\_type": "employee",

"\_id": "2",

"\_score": 1,

"\_source": {

"first\_name": "Jane",

"last\_name": "Smith",

"age": 32,

"about": "I like to collect rock albums",

"interests": [ "music" ]

}

}

]

}

}

About:

References:

1. Clinton Gormley, Zachary Tong, *Elasticsearch: The Definitive Guide,* O’Reilly Media.
2. Clinton Gormley, *“Elasticsearch- The Definitive Guide”*, https://www.elastic.co/blog/elasticsearch-definitive-guide
3. https://www.elastic.co/guide/en/elasticsearch/reference/current/modules-cluster.html
4. Morten Ingebrigtsen, https://www.elastic.co/blog/found-indexing-for-beginners-part1
5. Zachary Tong, *“What is an Elasticsearch Index?”* https://www.elastic.co/blog/what-is-an-elasticsearch-index
6. D. Chen, Y. Chen, B.N. Bronlow, P.P. Kanjamala, C.A. G. Arredono, B.L. RadSpinner, M.A. Ravelling*“Real-Time or Near Real-Time Persisting Daily Healthcare Data Into HDFS and ElasticSearch Index Inside a Big Data Platform” IEEE Transactions on Industrial Informatics.*