**ELASTIC SEARCH WITH SPRING BOOT**

**Elasticsearch** is a distributed, RESTful search and analytics engine built on Apache Lucene. It enables fast, scalable full-text search and near real-time data analytics on large volumes of JSON documents.

### Core Concepts

* **Document**: Basic unit of data, stored as JSON. For example, a user profile or a product record.
* **Index**: Logical collection of documents, like a database in relational systems. Each index has a name and contains documents with similar characteristics.
* **Shard**: An index is split into multiple shards to distribute data and search load across nodes. Each shard is a fully functional Lucene index.
* **Replica**: Copies of shards to ensure high availability and fault tolerance. Replicas also improve search performance by load balancing.
* **Cluster & Node**: A cluster is a set of one or more nodes (servers) working together. Nodes hold shards and perform indexing and search operations.

### Data Storage and Mapping

* Data is stored as immutable segments inside shards and periodically merged for optimization.
* **Mapping** defines the schema of documents, specifying data types and how fields are indexed and stored.
* Elasticsearch supports **dynamic mapping** to automatically detect fields.

### Text Analysis

* Text fields are analyzed by breaking down content into tokens (words or terms) using **analyzers** (tokenizers, filters like stemming, stop words).
* This enables powerful full-text search capabilities.

### Searching & Querying

* Elasticsearch provides a rich JSON-based **Query DSL** supporting:
  + Full-text queries (match, fuzzy)
  + Structured queries (term, range)
  + Boolean logic combinations
  + Filtering and sorting
* Search results are scored and ranked by relevance using Lucene’s scoring algorithm.

### Aggregations and Analytics

* Aggregations enable powerful data summarization and analytics, like calculating averages, sums, or grouping by terms (similar to SQL GROUP BY).
* Supports nested and pipeline aggregations for advanced analytics.

### Features

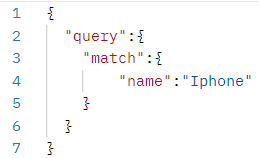
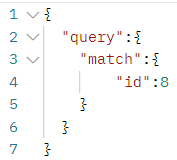
* **Near Real-Time** indexing and search.
* Highly **scalable** and **distributed**, allowing horizontal scaling by adding nodes.
* Schema-free and flexible JSON documents.
* RESTful API interface for easy integration.
* Works well with the ELK stack (Elasticsearch, Logstash, Kibana) for log analysis and visualization.

### Use Cases

* Application and website search
* Log and event data analytics
* Business intelligence dashboards
* Geo-location based search
* Security analytics and monitoring

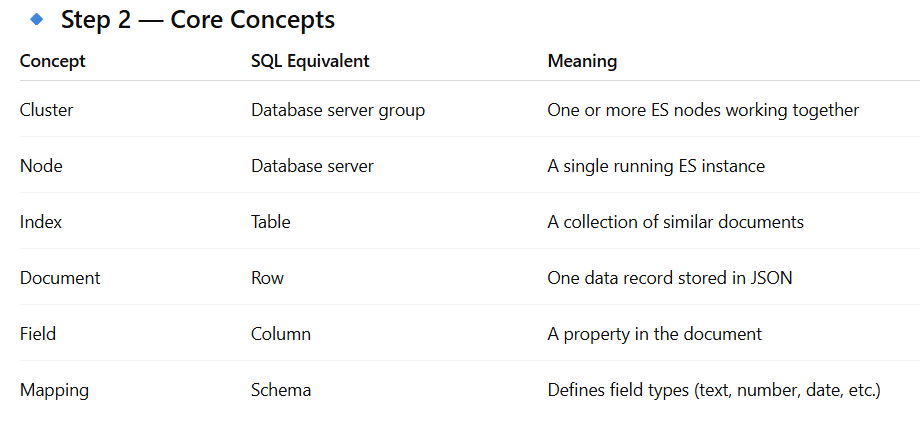
**PRACTICAL:**

* Download elastic search zip file
* In cmd run elasticsearch.bat file, it will take some time and start the server. Check for port and access it in browser. For local use, disable xpac security and rename network host to local host. Access on [localhost:9200](http://localhost:9200/)
* Refer the following project for practical: <https://github.com/irahulsm2405/ElasticSearch_SpringBoot.git>
* To see the project index, go to elastic search console/postman and go to rest api end point. eg: [localhost:9200/products](http://localhost:9200/products)
* If we want to query elastic search index/database, use following: <http://localhost:9200/products/_search>
* For filters, we can pass a filter in body and then send request to above url. Eg:



* For more filters refer to [ElasticSearchFilters.txt](https://github.com/irahulsm2405/ElasticSearch_SpringBoot/blob/main/ElasticSearchFilters.txt) on Github.

**CONCEPTS:**



**QUERY DSL (Domain Specific Language)**

 JSON**-based** syntax for defining Elasticsearch queries.

 Let’s you combine:

* **Full-text search** (match, multi\_match, query\_string)
* **Exact matches** (term, terms)
* **Filters** (range, exists, wildcard)
* **Logical conditions** (bool → must, should, must\_not)

 Output is deterministic and machine-readable (unlike free-text queries).