Basic Elasticsearch Concepts

In this document, we'll cover the basics of what you need to know about Elasticsearch in order to use it.

Indexing

Elasticsearch is able to achieve fast search responses because, instead of searching the text directly, it searches an index instead.

This is like retrieving pages in a book related to a keyword by scanning the index at the back of a book, as opposed to searching every word of every page of the book.

This type of index is called an **inverted index**, because it inverts a page-centric data structure (page->words) to a keyword-centric data structure (word->pages).

Elasticsearch uses Apache Lucene to create and manage this inverted index.

How Elasticsearch represents data

In Elasticsearch, a **Document** is the unit of search and index.

An index consists of one or more Documents, and a Document consists of one or more Fields.

> In database terminology, a Document corresponds to a table row, and a Field corresponds to a table column.

Schema

Unlike Solr, Elasticsearch is schema-free. Well, kinda.

Whilst you are not required to specify a schema before indexing documents, it is necessary to add **mapping** declarations if you require anything but the most basic fields and operations.

This is no different from specifying a schema!

The schema declares:

- what fields there are
- which field should be used as the unique/primary key
- which fields are required
- how to index and search each field

In Elasticsearch, an index may store documents of different "mapping types". You can associate multiple mapping definitions for each mapping type. A **mapping type** is a way of separating the documents in an index into logical groups.

To create a mapping, you will need the Put Mapping API, or you can add multiple mappings when you create an index.

Query DSL

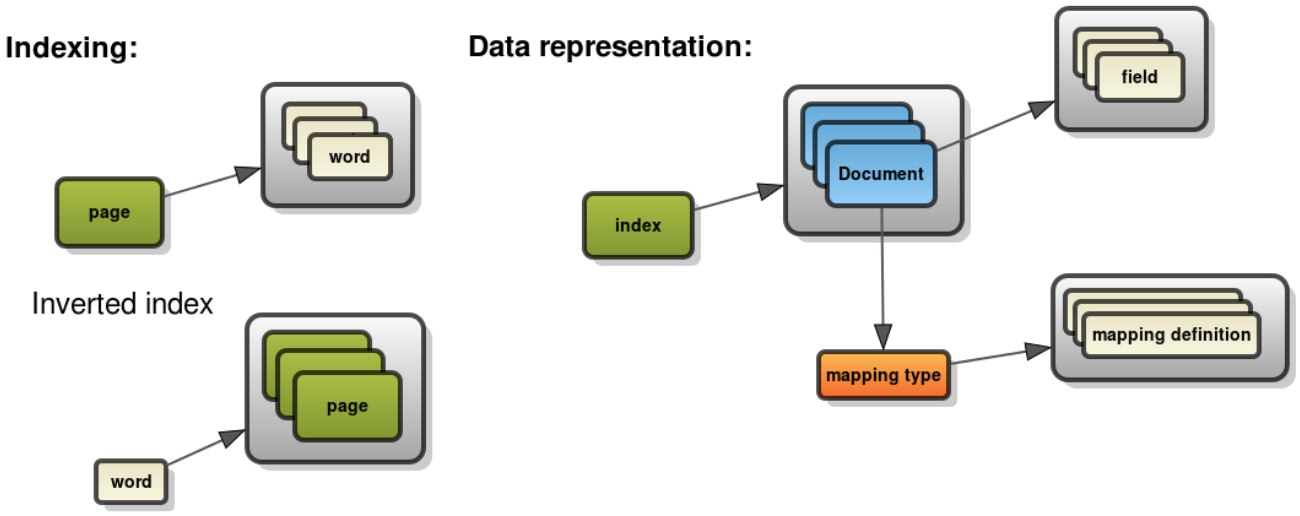
The Query DSL is Elasticsearch's way of making Lucene's query syntax accessible to users, allowing complex queries to be composed using a JSON syntax.

Like Lucene, there are basic queries such as **term** or **prefix** queries and also compound queries like the **bool** query.

The main structure of a query is roughly:

```
curl -X POST "http://localhost:9200/blog/_search?pretty=true" -d `
{"from": 0,
"size": 10,
"query" : QUERY_JSON,
FILTER_JSON,
FACET_JSON,
SORT_JSON
}'
```

Summary



main query structure

```
curl -X POST "http://localhost:9200/blog/_search?pretty=true" -d '{
                                "from": 0,
                                "size": 10,
                              "query" : QUERY_JSON,
                                  FILTER_JSON,
                                  FACET_JSON,
                                   SORT_JSON
                                      }'
```

What about Lucene, by the way?

Elasticsearch is powered by Lucene, a powerful open-source full-text search library, under the hood.

The relationship between Elasticsearch and Lucene, is like that of the relationship between a car and its engine.

For the purpose of this introduction, we haven't differentiated between the two, just as to most people, the distinction between a car and its engine is not terribly important when

learning how to drive a car.

However, to a mechanic, the distinction is a very important one. Similarly, when we dive deeper under the bonnet of Elasticsearch, we'll explore the distinctions between Lucene and ElasticSearch in detail.