

# NLP 2

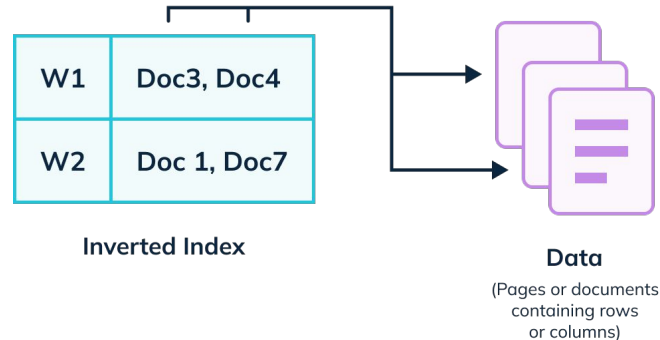
Inteligencia artificial avanzada para la ciencia  
de datos II Modulo 5 NLP 2

Inverted Index

# Inverted Index

An **inverted index** is a data structure used in information retrieval and text search systems to efficiently store and retrieve a large amount of textual data.

It is designed to improve the speed of searching and retrieval operations by mapping terms (words or tokens) to the documents or data records in which they appear.



# Software available

- **Lucene/Solr:** Apache Lucene is a high-performance, full-featured text search engine library written in Java. Apache Solr is an enterprise search platform built on top of Lucene. These tools are widely used for creating search engines and handling complex search operations.
- **Elasticsearch:** Elasticsearch is a distributed, RESTful search and analytics engine built on top of Lucene. It's often used for real-time search and log analytics. Elasticsearch is known for its scalability and ease of use.
- **Opensearch:** Elasticsearch fork supported by AWS, elasticsearch is an open source initiative that didn't receive earnings for AWS PaaS distribution of Elasticsearch, AWS forked the project and keep it open source to be distributed in the cloud.

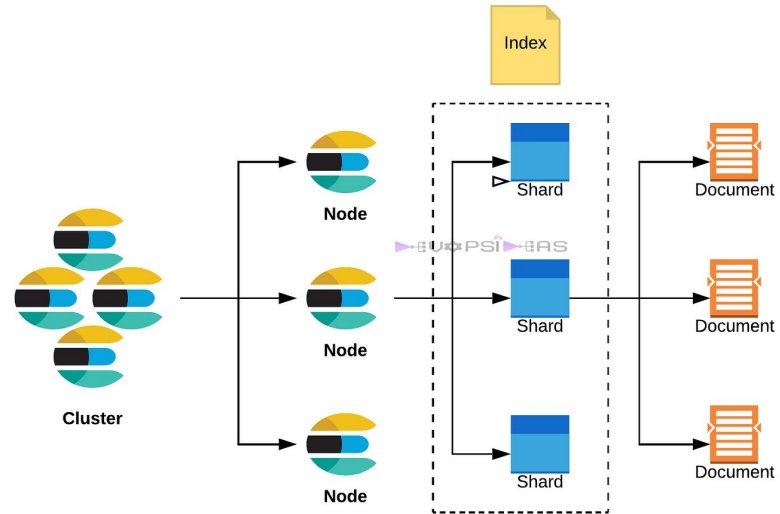


# ES Architecture

Elasticsearch is a distributed, RESTful search and analytics engine designed for horizontal scalability, real-time search, and large-scale data storage and retrieval.

Its architecture is based on a cluster structure to handle vast amounts of data and to provide high availability and fault tolerance.

Elasticsearch Component Relation



# Basics

Imagine you are back in the 80's and you want to search a topic in a book.

You will go to the index , in order to find the pages in which this topic (Token) appears.

Here are some aspects you would need to know:

1. Someone read the whole book and make a map of the word and the page where it appears.
2. This map is sorted and structured to be efficient to look for

Contents	
Foreword .....	ix
Preface to the Paperback Edition .....	xv
Acknowledgments .....	xvii
<b>How to Understand Math .....</b>	<b>1</b>
Introduction .....	3
The "Aha!" Factor .....	9
Think Like a Mathematician .....	23
<b>Playful Problem-Solving .....</b>	<b>45</b>
Math You Can Play .....	47
Math You Can Touch .....	63
Math That Makes You Think .....	87
<b>Math with Living Books .....</b>	<b>107</b>
Math You Can Read .....	109
4,000 Years of Stumpers .....	119
<b>Let's Get Practical .....</b>	<b>137</b>
Weaving It All Together .....	139
Struggling with Math .....	161
vii	

European math, 129	geometry, 56, 84, 188
Eves, Howard, 87, 105, 251	Bhaskara's dissection, 126, 241
exercise vs. problem, 87	drafting, 83, 84
<b>F</b>	non-Euclidean, 202
Fermat, Pierre de, 87, 131	resources, 191
Fibonacci, 21, 22, 112, 127, 129, 226, 227, 243	Thales, 123, 240
fight it, 194	getting stuck, 44, 145, 172, 197, 208
finger-counting, 49, 65, 68, 165	Glaisher, James, 111, 252
Finkel, Dan, 206, 251	Gleiser, Oleg, 187, 221, 252
fluency, 159, 175	Gödel, Kurt, 133, 135
forgetting, 169, 172	Golden, John, 143, 170, 190, 224, 229, 252
fractals, 18, 19, 38, 133, 168, 169	grade level, 174
fractions, 27, 50, 75, 80, 97, 99, 172	Greek math, 123
division, 93	Gross, Herb, 40, 99, 162, 173, 186, 190, 229, 230, 252
half-halves, 42	gryphons, 97
manipulatives, 65	<b>H</b>
multiplication, 31	Halmos, Paul, 194, 217, 252
of = multiply, 95	Hankel, Hermann, 109, 252
Freyer model, 163, 164	Hardy, G. H., 53, 111, 133, 252
freedom, 3, 6, 16, 66, 142, 167, 187, 198	Heath, Sir Thomas L., 240
frog, 104	high school, 185
functionally literate, 17	Hindu math, 125
Furuta, Burt, 23, 251	Hippocrates, 117, 123
<b>G</b>	history, 87, 105, 109, 119
games	home schooling, xvii, 8, 9, 162, 167
Cuisenaire rod game, 66	buddy math, 144
dollar game, 68	daily four, 168
dull vs. strategy, 60	success, 180
gomoku, 61, 62, 79	homework, 6, 10, 39, 81, 120, 143, 144, 168, 175, 195, 196
math war, 60	horse, 106
modifying, 42	Huff, Durrell, 192, 215
nickel game, 76	hundred chart, 74
Num, 42	<b>I</b>
pattern block mastermind, 141	imagination, vii, 11, 20, 28, 38, 42, 91, 92, 133, 167, 205, 206
race to 100, 78	Indian math, 125
tens fish, 60	infinity, 5, 15, 58, 87, 105, 125, 129, 131, 133, 167, 168, 169, 201, 205
twenty-four, 60	instrumental understanding, 24
What number am I?, 13	integers, 75
gaps, perceiving, 179	
Gardner, Martin, 43, 47, 111, 115, 202, 214, 215, 251	
Garlikov, Rick, 23, 251	

INDEX • 259

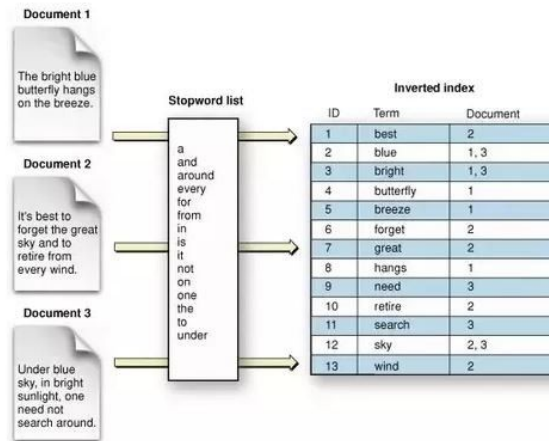
# Index time process

Reverse index software is the

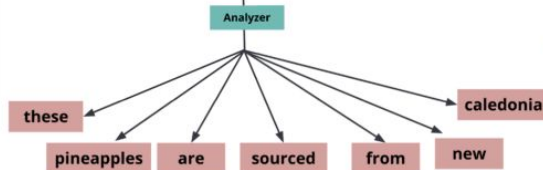
- responsible of building that index for you, so later you can search for a word and retrieve your documents.

This means software has to:

- Process the text and tokenize
- Create the reverse index structure
- Keep some statistics to score the relevance given a sequence.



```
PUT temp_index/_doc/3
{
  "description": "These pineapples are sourced from New Caledonia."
}
```



Inverted Index	
term	doc id
are	1,2,3
caledonia	3
from	1,2,3
new	1,2,3
pineapples	1,2,3
sourced	1,2,3
these	1,2,3
zealand	1,2

# Query time process

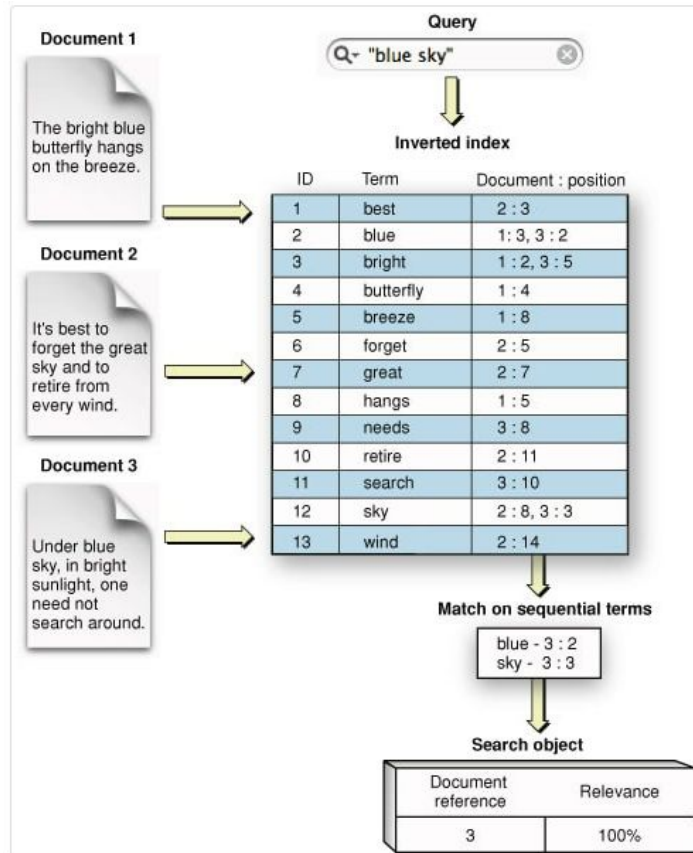
Also, this software will receive the input (sequence) and will be responsible for:

- Tokenize the input
- Compared with the index so it will generate some matches
- Sort results by using a score algorithm (elasticsearch uses TF/IDF as default)

```
Dev Tools
History Settings Help
Console Search Profiler Grok Debugger

1 GET /kibana_sample_data_logs/_search
2 {
3   "size": 0,
4   "aggs": {
5     "ip_count": {
6       "value_count": {
7         "field": "clientip"
8       }
9     }
10  }
11 }
12 }
```

```
1 {
2   "took": 0,
3   "timed_out": false,
4   "_shards": {
5     "total": 1,
6     "successful": 1,
7     "skipped": 0,
8     "failed": 0
9   },
10  "hits": {
11    "total": {
12      "value": 10000,
13      "relation": "gte"
14    },
15    "max_score": null,
16    "hits": []
17  },
18  "aggregations": {
19    "ip_count": {
20      "value": 14085
21    }
22  }
23 }
24 }
```



# Thanks

Do you have any questions?

emmanuel.paez@tec.mx  
Slack #module-5-nlp-1