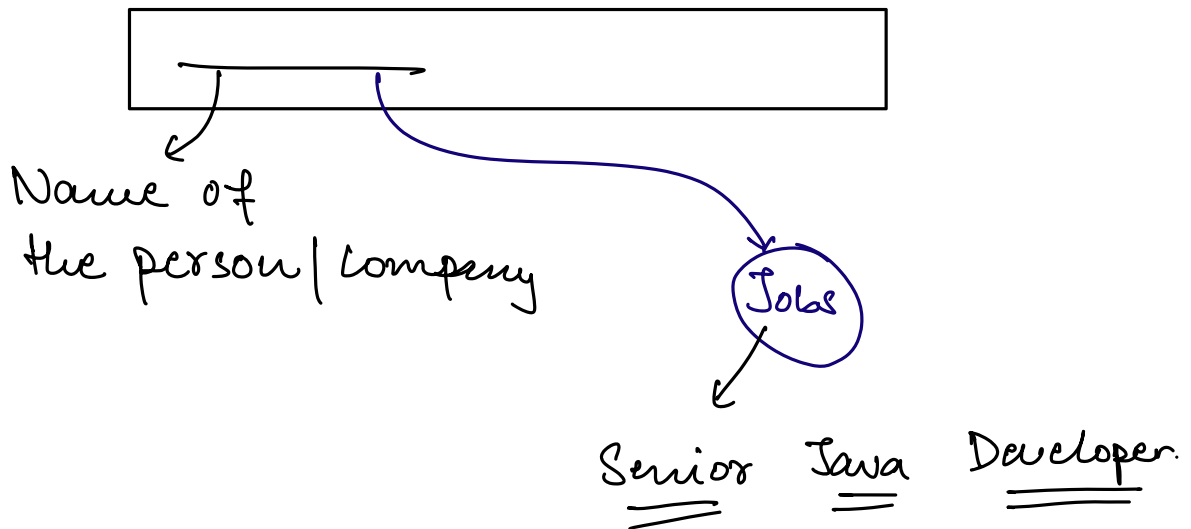
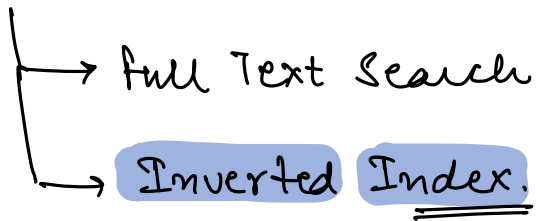
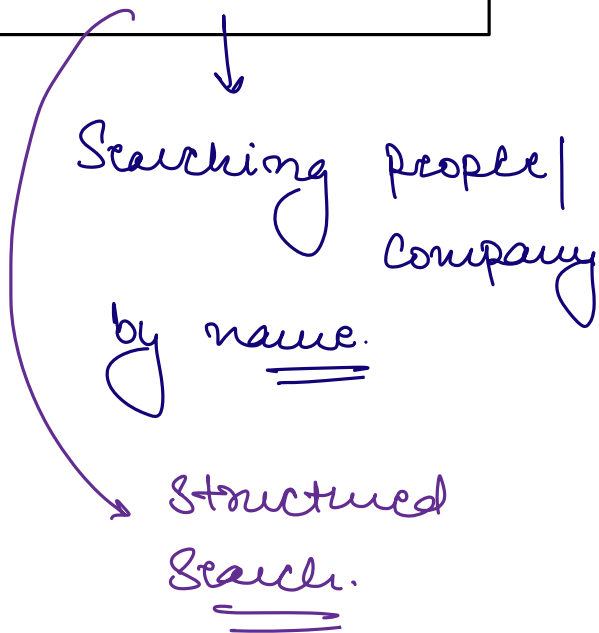


# Agenda.

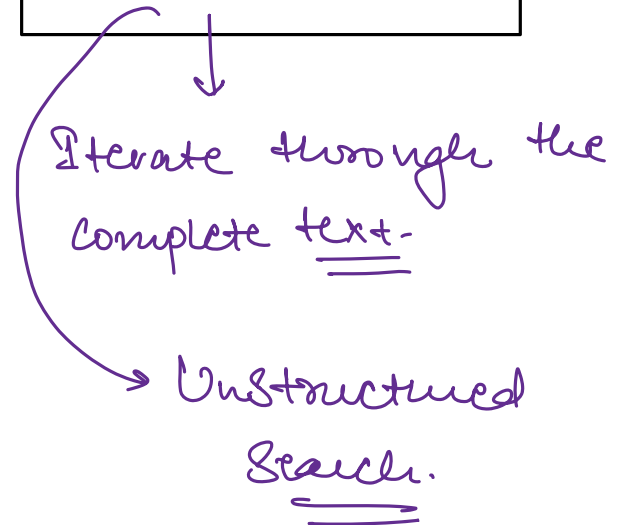
→ Elastic Search



Attribute based Search

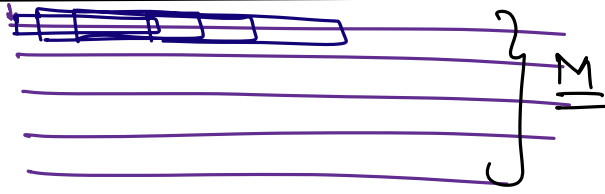


Full Text Search



⇒ SQL DB.

posts.

id	userid	title	timestamp	Content
				

⇒ Search all the posts which contains "senior java developer".

Select \* from posts  
where content like '%Senior Java Developer%';

$N \times (MK)$

No. of posts.

length of the content

Query length

K

## ⇒ Elastic Search

↳ to perform full text search efficiently.

## ⇒ Inverted Index.

↳ Mapping of the important words with page numbers where a particular word is present.

"India" → D<sub>4</sub> D<sub>10</sub> D<sub>20</sub> D<sub>25</sub> . . . .

"Cricket" → \_ \_ \_ \_ \_

"Australia" → \_ \_ \_ \_ \_

"us"

"the"

"must"

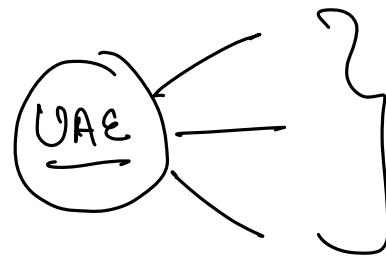
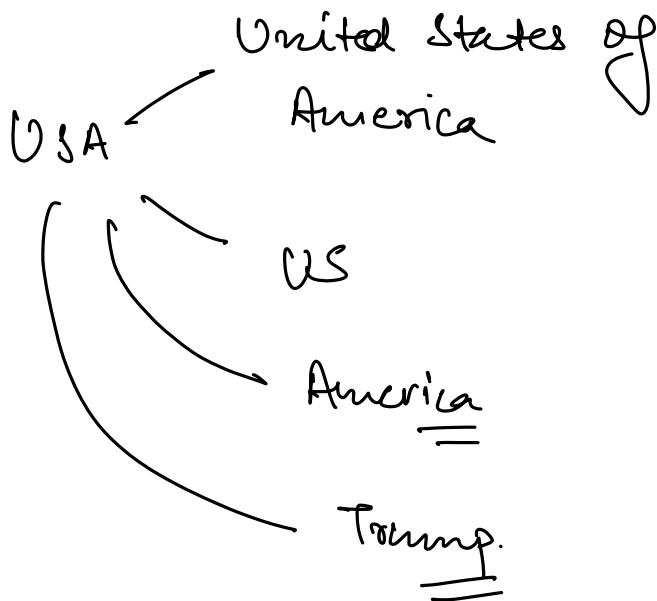
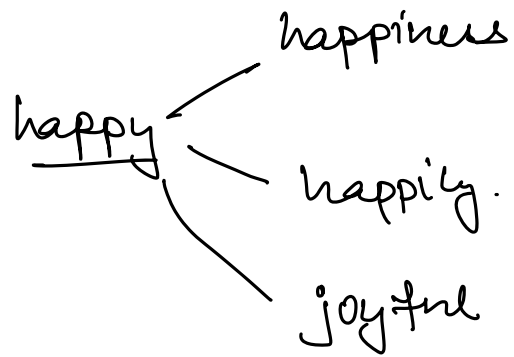
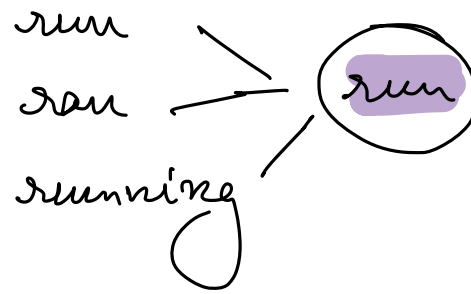
"an"

X Not so important  
words.

# NLP.

↳ Natural language Processing

wikipedia dump.



## # How to create Inverted Index ?

D<sub>1</sub>: "India defeated Australia in semi final to receive their place in CT 2025 final".

D<sub>2</sub>: "CT 2025 final is planned to be conducted in Dubai, UAE".

D<sub>3</sub>: "BJP got a major win in Delhi 2025 election".

D<sub>4</sub>: "Rohit Sharma may announce retirement after Champions Trophy 2025".

D<sub>5</sub>: "Australia out of Champions Trophy 2025".

1) Removal of Stop words.

↳ a | an | the | of | as | is | ...

2) Stemming | Root word | Base word.

3) Tokenization.

D<sub>1</sub>: "India defeated Australia in semi final  
to receive their ~~place~~ in CT 2025 final"

↓ remove stop words.

D<sub>1</sub>: "India defeated Australia semi final  
receive ~~place~~ CT 2025 final"

↓ Stemming

India defeat Australia semi final  
receive ~~place~~ CT 2025 final

↓ Tokens. → Unigrams  
→ Bigrams  
→ Trigrams

India  $\rightarrow D_1, D_7, D_{40}, D_{64}, D_{78}, D_{100} \dots$   
0.9 0.88 0.87 0.75 0.71 0.69 - -

defeat  $\rightarrow$  —

Australia  $\rightarrow D_1$

Semi  $\rightarrow$  —

Final  $\rightarrow$  —

win  $\rightarrow$  —

Champion trophy

India vs Australia

—  
—  
—

$\Rightarrow$  How to assign relevancy / importance of a Document for a particular word?

frequency.

# TF - IDF  $\rightarrow$  Inverse Document frequency.  
 $\downarrow$   
Text frequency

"Senior Java Developer"

Senior → — — — — — — — —

Java → —————

Developer → —————

Senior Java → —————

Senior Developer → —————

Java Developer → —————

Software  
Engineer.

Apache

Lucene

→ System that performs all the NLP steps on the Documents provided and creates Inverted Index to make search more efficient.

→ Works only on a single m/c.

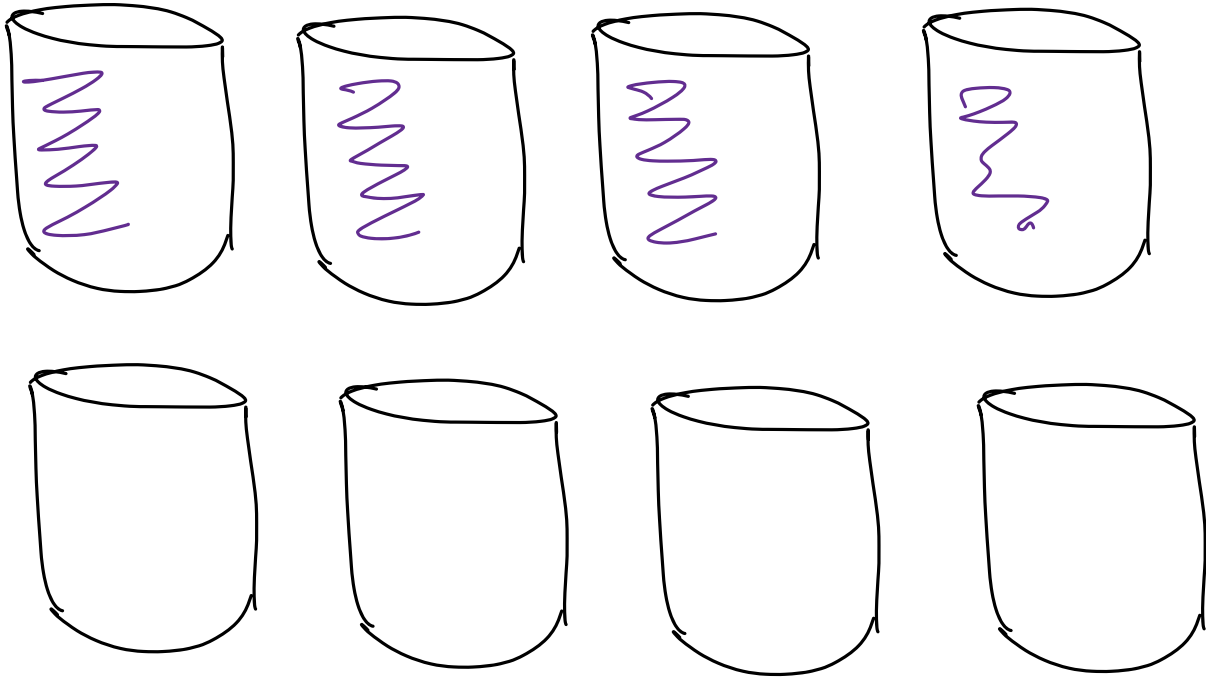
⇒ ElasticSearch.

→ Works on distributed machines.

→ Sharded Apache Lucene.



ES:



ES

→ Document DB.  
→ Sharding key ⇒

doc-id

⇒ Elastic Search

→ Highly Available

+

Eventual Consistent