Index me baby

Par Nerea Enrique (aka CurriedN)

DRUPALCAMP ROAZHON

Rennes - 28 au 30 mars 2024

your high-digital place open

AXESS Digital & Business Experts

SMILE

Talan

Index me baby

Par Nerea Enrique (aka CurriedN)

DRUPALCAMP ROAZHON

Rennes - 28 au 30 mars 2024

Hola!

ekino.

Who?

✌️ **Nerea Enrique**

@nep94

@curryed

Where?

🐘 **PHP Engineer at ekino.**

4 years

What?

@CurriedN

6 years

Content

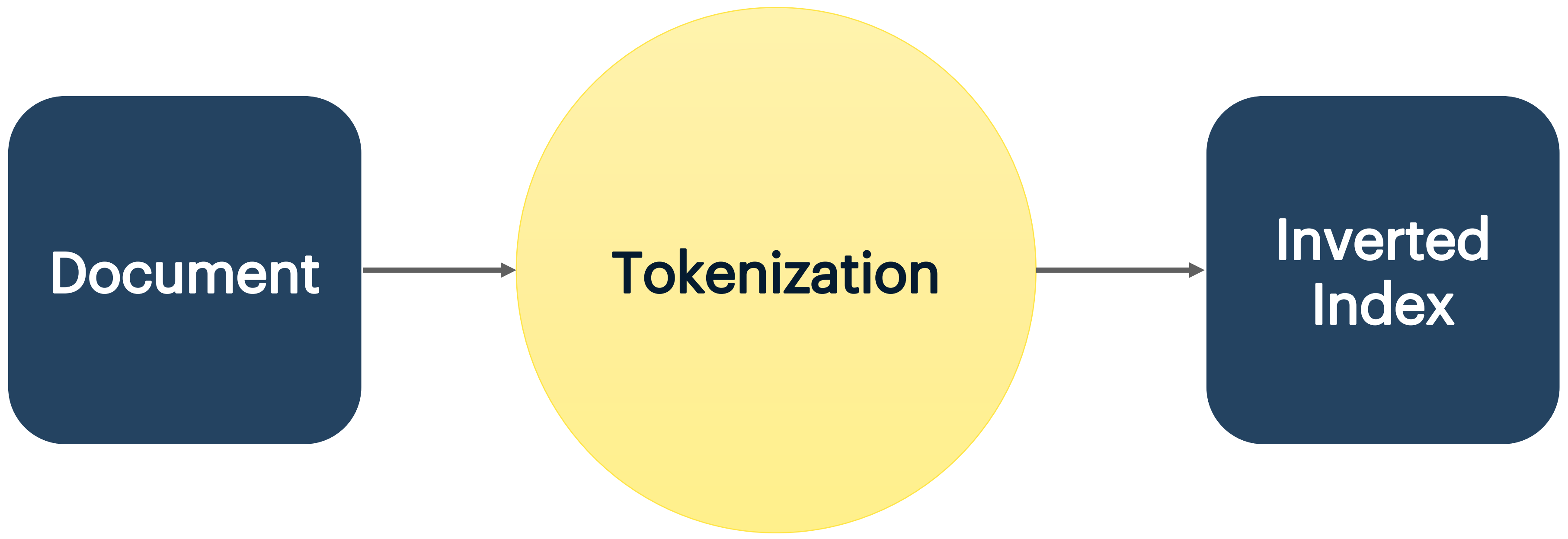
ekino.

→ Search engines

Full text search

Document

Full text search



ekino.

Tokenization

- Lexical analysis, lexing or tokenization
    - Process of converting a sequence of characters into a sequence of tokens
- Tokens
    - Strings/symbols with an assigned and thus identified meaning
- Break sentences into individual tokens
- Make it easier to understand and analyze it
- Easier to change tokens than sentences

Source: https://en.wikipedia.org/wiki/Lexical_analysis#Token
https://beram-presentation.gitlab.io/php-static-analysis-101/#10

Tokenization

The quick brown fox jumps over the lazy dog

Tokenization

The quick brown fox jumps over the lazy dog

↓

The, quick, brown, fox, jumps, over, the, lazy, dog

Source: https://www.elastic.co/guide/en/elasticsearch/reference/current/analysis-overview.html

Inverted Index

- Map a token to documents
- Instead of a document to words

Source: https://en.wikipedia.org/wiki/Inverted_index

Inverted index

D1 = "The quick brown fox."
D2 = "The lazy dog sleeps."
D3 = "A quick nap is refreshing."

D1 = The, quick, brown, fox
D2 = The, lazy, dog, sleeps
D3 = A, quick, nap, is, refreshing

Source: https://en.wikipedia.org/wiki/Inverted_index

Inverted index

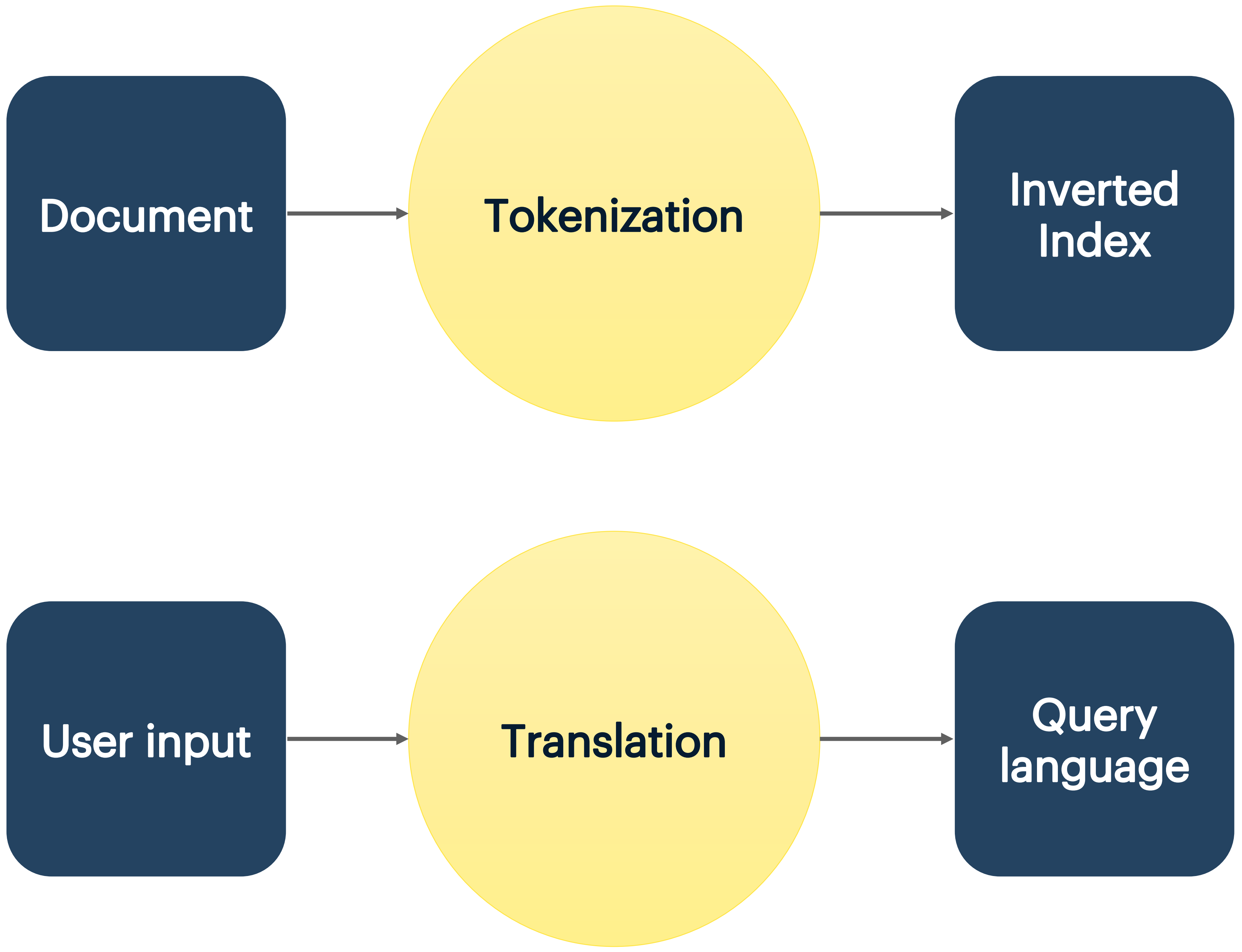
D1 = The, quick, brown, fox
D2 = The, lazy, dog, sleeps
D3 = A, quick, nap, is, refreshing

$\Rightarrow$

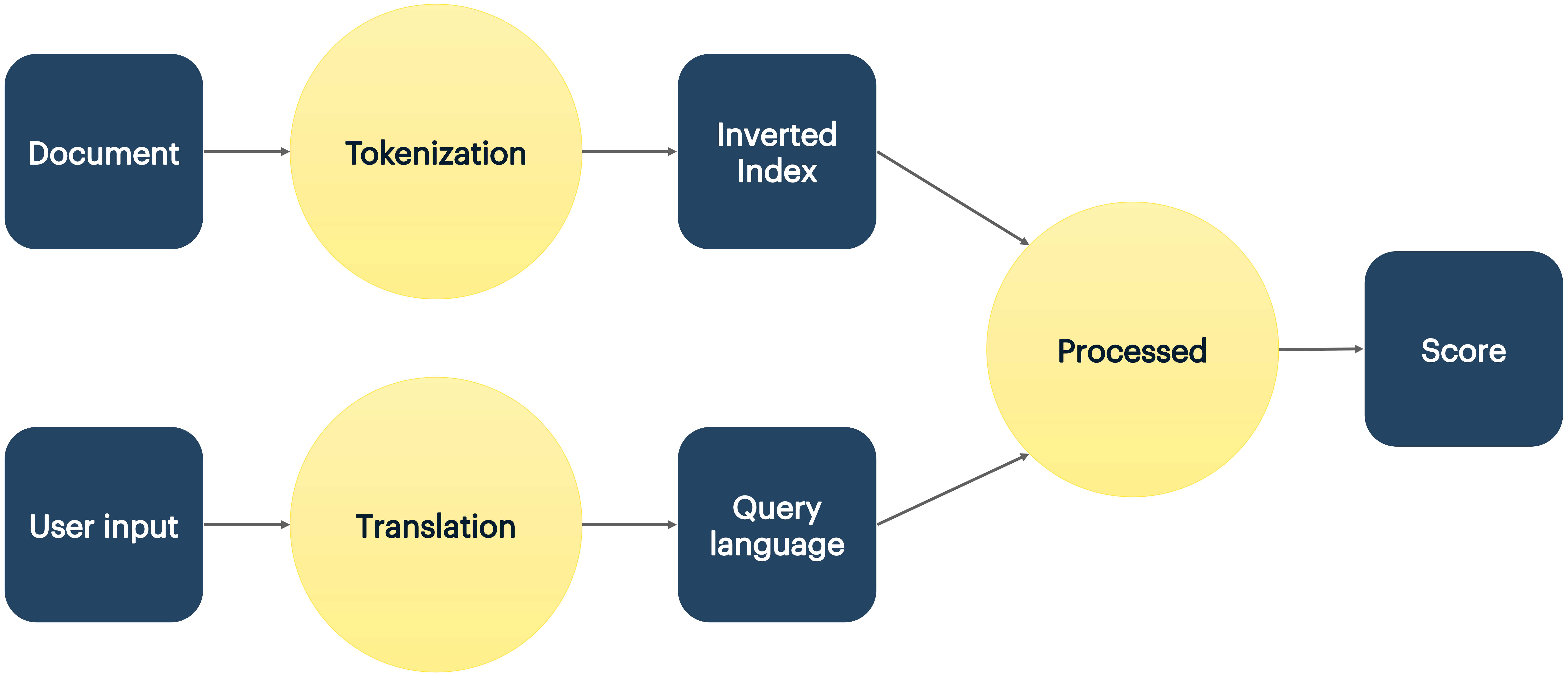
brown: [1]
dog: [2]
fox: [1]
is: [3]
lazy: [2]
nap: [3]
quick: [1, 3]
refreshing: [3]
sleeps: [2]
the: [1, 2]

Source: https://en.wikipedia.org/wiki/Inverted_index

Full text search



Full text search



Welcome to my library

The building not the library you might think of 😎

ekino.

Library

Library

Title: Harry Potter and the Deathly Hallows
Genre: Fantasy

Title: Harry Potter and the Prisoner of Azkaban
Genre: Fantasy

Title: The Lord of the Rings
Genre: Fantasy

Title: The Adventures of Huckleberry Finn
Genre: Adventure fiction

Title: Harry Potter and the Half-Blood Prince
Genre: Fantasy

Title: The Silmarillion
Genre: Fantasy

Title: The Hobbit
Genre: Fantasy

Title: Harry Potter and the Philosopher's Stone
Genre: Fantasy

Title: The Grapes of Wrath
Genre: Historical fiction

Title: The Catcher in the Rye
Genre: Realistic fiction

Title: Harry Potter and the Goblet of Fire
Genre: Fantasty

Title: Harry Potter and the Order of the Phoenix
Genre: Fantasy

Title: Harry Potter and the Chamber of Secrets
Genre: Fantasy

Title: The Great Gatsby
Genre: Fiction, tragedy

ekino.

Library

ekino.

Library

Fantasy

Title: Harry Potter and the Deathly Hallows
Genre: Fantasy

Title: Harry Potter and the Prisoner of Azkaban
Genre: Fantasy

Title: The Lord of the Rings
Genre: Fantasy

Title: The Adventures of Huckleberry Finn
Genre: Adventure fiction

Title: Harry Potter and the Half-Blood Prince
Genre: Fantasy

Title: The Silmarillion
Genre: Fantasy

Title: The Hobbit
Genre: Fantasy

Title: Harry Potter and the Philosopher's Stone
Genre: Fantasy

Title: The Grapes of Wrath
Genre: Historical fiction

Title: The Catcher in the Rye
Genre: Realistic fiction

Title: Harry Potter and the Goblet of Fire
Genre: Fantasy

Title: Harry Potter and the Order of the Phoenix
Genre: Fantasy

Title: Harry Potter and the Chamber of Secrets
Genre: Fantasy

Title: The Great Gatsby
Genre: Fiction, tragedy

Library

ekino.

Library

Fantasy & Harry & Potter

Title: Harry Potter and the Deathly Hallows
Genre: Fantasy

Title: Harry Potter and the Prisoner of Azkaban
Genre: Fantasy

Title: The Lord of the Rings
Genre: Fantasy

Title: The Adventures of Huckleberry Finn
Genre: Adventure fiction

Title: Harry Potter and the Half-Blood Prince
Genre: Fantasy

Title: The Silmarillion
Genre: Fantasy

Title: Harry Potter and the Philosopher's Stone
Genre: Fantasy

Title: The Hobbit
Genre: Fantasy

Title: The Grapes of Wrath
Genre: Historical fiction

Title: Harry Potter and the Goblet of Fire
Genre: Fantasty

Title: The Catcher in the Rye
Genre: Realistic fiction

Title: Harry Potter and the Order of the Phoenix
Genre: Fantasy

Title: Harry Potter and the Chamber of Secrets
Genre: Fantasy

Title: The Great Gatsby
Genre: Fiction, tragedy

Library

Library

Fantasy &
Harry &
Potter &
Phoenix

Title: Harry Potter and the Prisoner of Azkaban
Genre: Fantasy

Title: Harry Potter and the Deathly Hallows
Genre: Fantasy

Title: The Lord of the Rings
Genre: Fantasy

Title: The Adventures of Huckleberry Finn
Genre: Adventure fiction

Title: Harry Potter and the Half-Blood Prince
Genre: Fantasy

Title: The Silmarillion
Genre: Fantasy

Title: Harry Potter and the Philosopher's Stone
Genre: Fantasy

Title: The Hobbit
Genre: Fantasy

Title: The Grapes of Wrath
Genre: Historical fiction

Title: Harry Potter and the Goblet of Fire
Genre: Fantasty

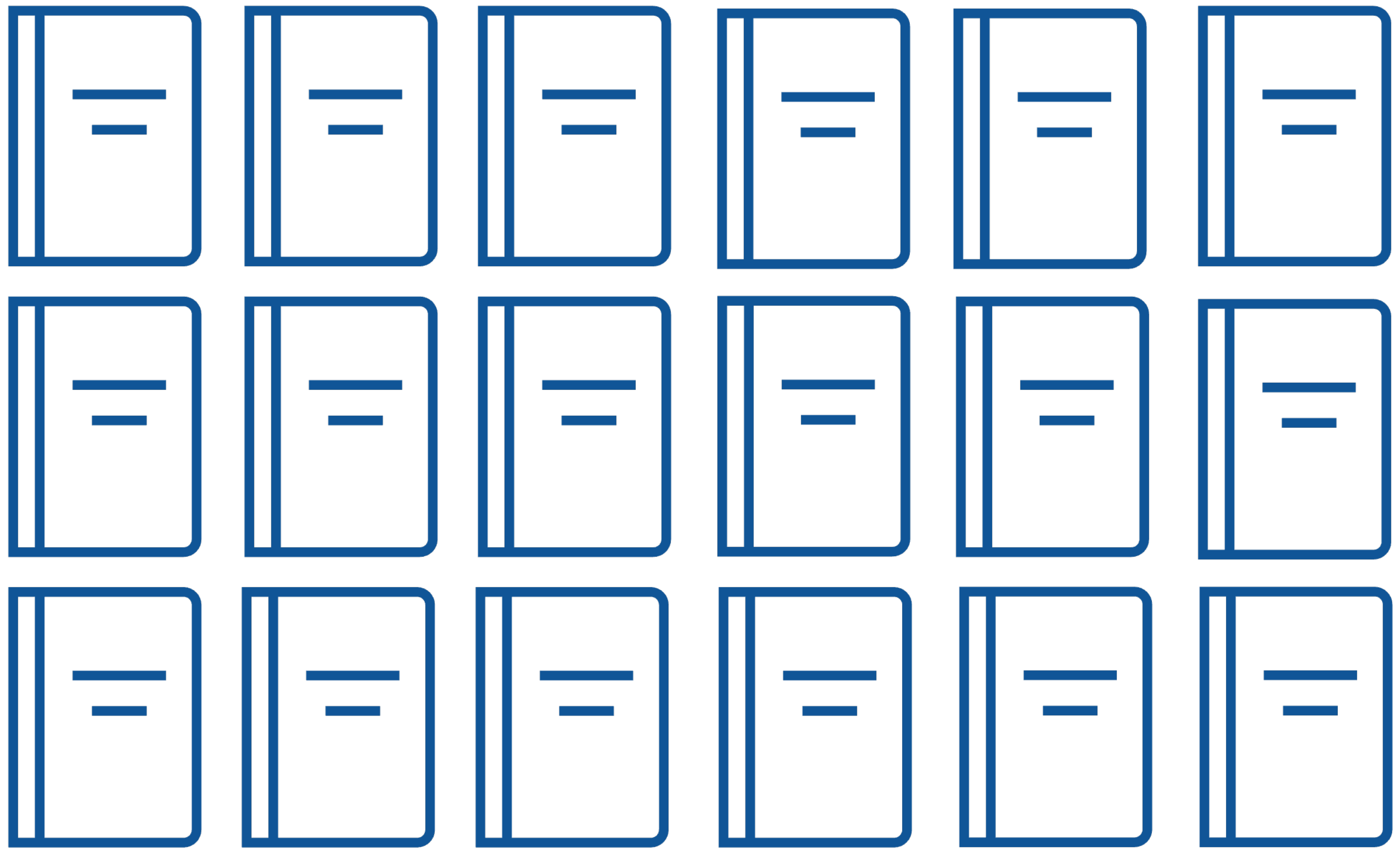
Title: The Catcher in the Rye
Genre: Realistic fiction

Title: Harry Potter and the Order of the Phoenix
Genre: Fantasy

Title: Harry Potter and the Chamber of Secrets
Genre: Fantasy

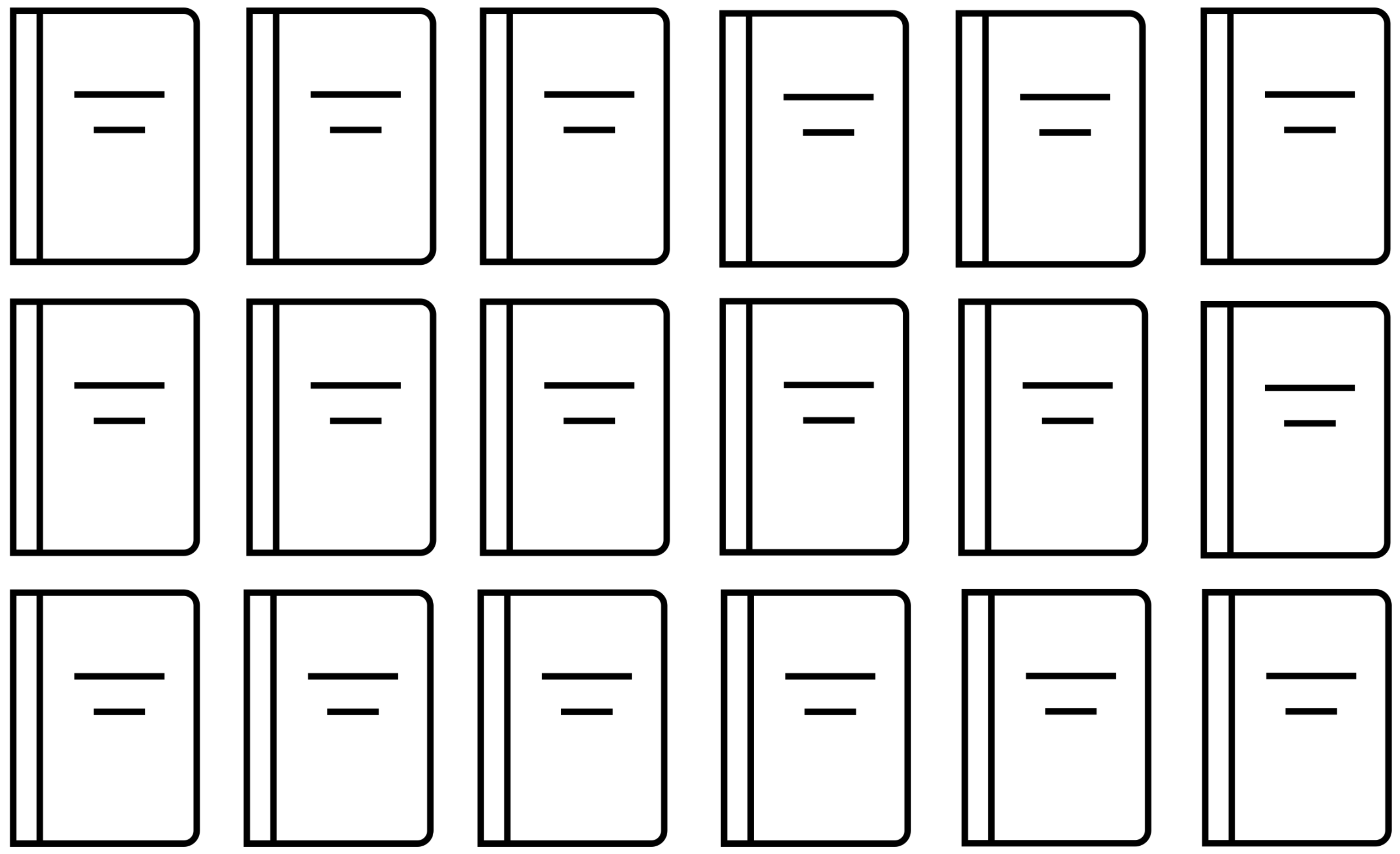
Title: The Great Gatsby
Genre: Fiction, tragedy

Library



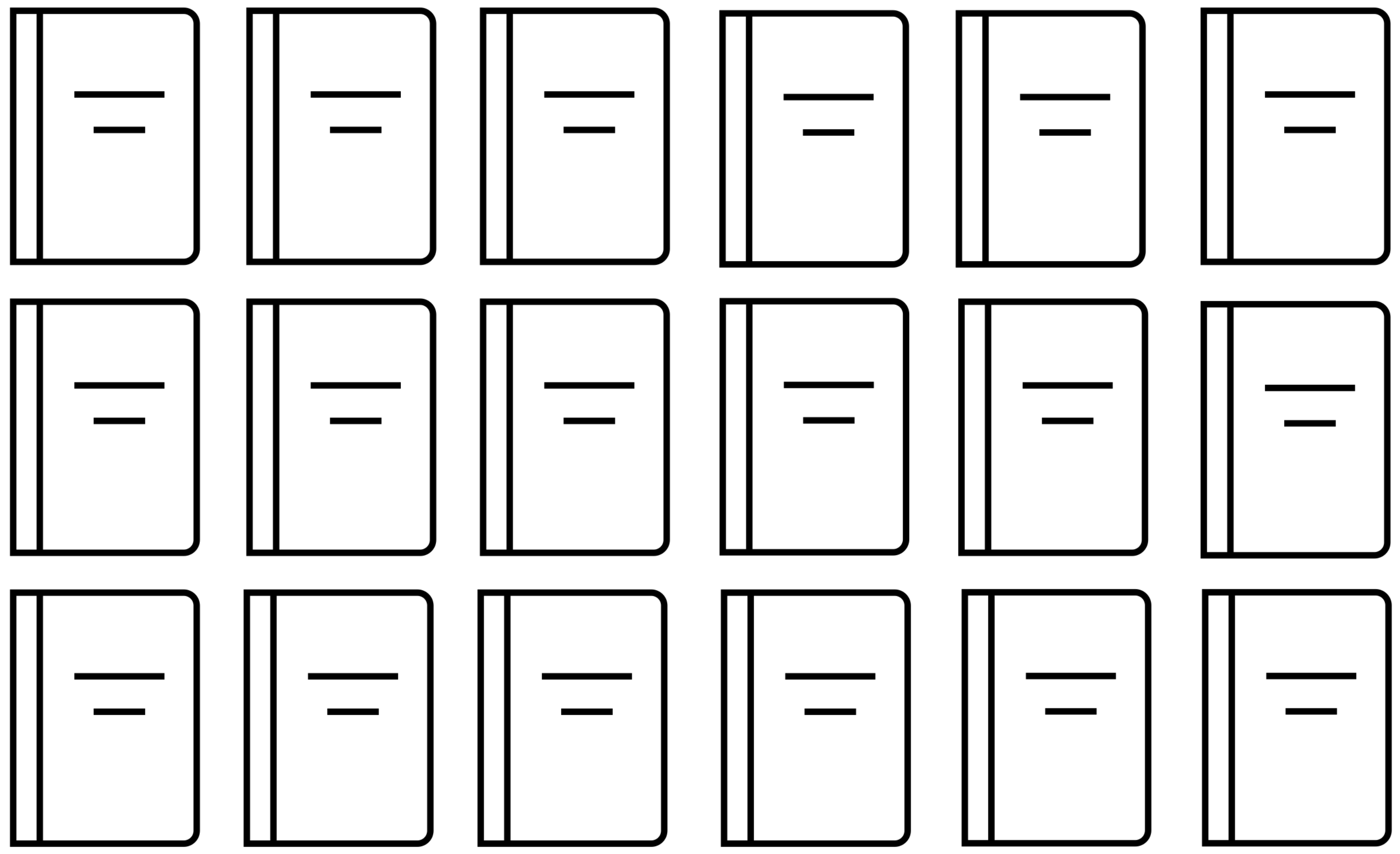
Documents

Library



Inversed index

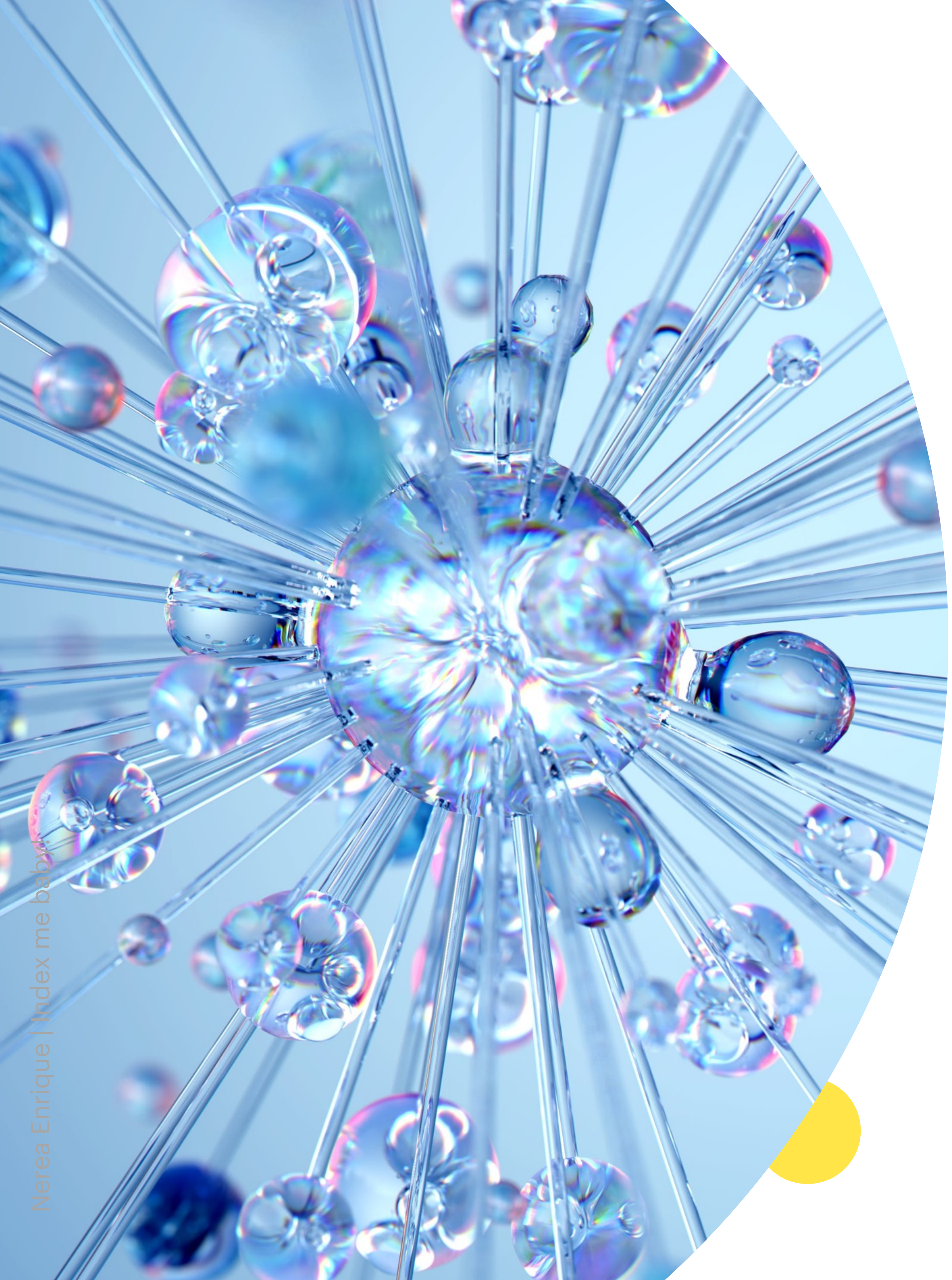
Library



User

ekino.

→ Elastic math 🤓



Elastic math

Some notions

- Inverse Document Frequency IDF
- Term Frequency TF
- Coordination factor
- Score

Inversed Document Frequency (IDF)

- Quantifies the importance of a term for a document
- Calculated as

$$idf(D, tm) = \log \frac{totalDocuments}{documentsContaining(tm)}$$

Source: https://en.wikipedia.org/wiki/Tf%E2%80%93idf

ekino.

Elastic math

Inversed Document Frequency (IDF)

Harry, Potter, tragedy

- More frequently a term appears => less important it is => lower IDF it gets

Title: The Grapes of Wrath
Genre: Historical fiction

Title: Harry Potter and the Goblet of Fire
Genre: Fantasty

Title: The Catcher in the Rye
Genre: Realistic fiction

Title: Harry Potter and the Order of the Phoenix
Genre: Fantasy

Title: Well done if you can read this
Genre: Unfiction

Title: Harry Potter and the Chamber of Secrets
Genre: Fantasy

Title: The Great Gatsby
Genre: Fiction, tragedy

Source: https://en.wikipedia.org/wiki/Tf%E2%80%93idf

Term frequency (TF)

- Measures the frequency of a term within a document
- Calculated as

$$tf(D, tm) = \frac{timesTermAppearsInDocument(D, tm)}{numberOfTermsWithinTheDocument(D)}$$

Source: https://en.wikipedia.org/wiki/Tf%E2%80%93idf

Term frequency (TF)

D1 = “The cat chased the mouse. The mouse scaped.”

D2 = ”There is a mouse in the house.”

Source: https://en.wikipedia.org/wiki/Tf%E2%80%93idf

Term frequency (TF)

D1 = “The cat chased the mouse. The mouse scaped.”

D2 = ”There is a mouse in the house.”

↓

TF(D1, “mouse”) = 2/8 = 0,25

TF(D2, “mouse”) = 1/7 = 0,14

Source: https://en.wikipedia.org/wiki/Tf%E2%80%93idf

Coordination factor

- Number of terms from the query matched in the document

$$cf = \frac{q(D)}{q}$$

Coordination factor

D1 = “The cat chased the mouse. The mouse scaped.”

D2 = ”There is a mouse in the house.”

CF(D1, “mouse”) = 1/1 = 1

CF(D2, “mouse”) = 1/1 = 1

CF(D1, “mouse, cat”) = 2/2 = 1

CF(D2, “mouse, cat”) = 1/2 = 0,5

Score

- Computed value
- Based on IDF and TF
- Ranks documents

Score

Fantasy, Harry Potter, Phoenix

- Higher the score => Higher the relevance

Title: The Grapes of Wrath
Genre: Historical fiction

Title: Harry Potter and the Goblet of Fire
Genre: Fantasty

Title: The Catcher in the Rye
Genre: Realistic fiction

Title: Well done if you can read this
Genre: Unfiction

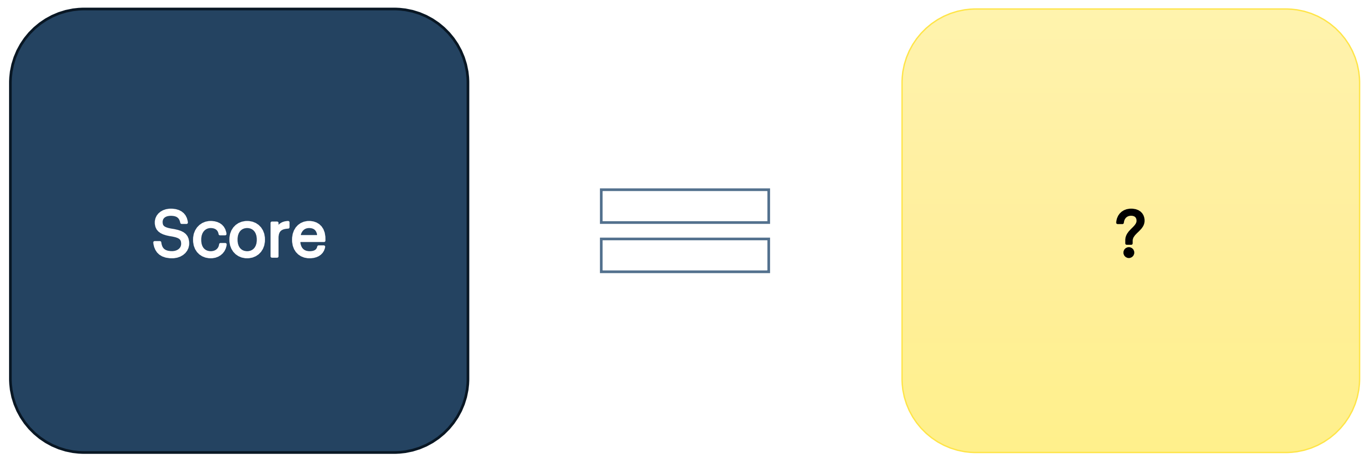
Title: Harry Potter and the Order of the Phoenix
Genre: Fantasy

Title: Harry Potter and the Chamber of Secrets
Genre: Fantasy

Title: The Great Gatsby
Genre: Fiction, tragedy

Elastic math

Score



Elastic math

Disclaimer

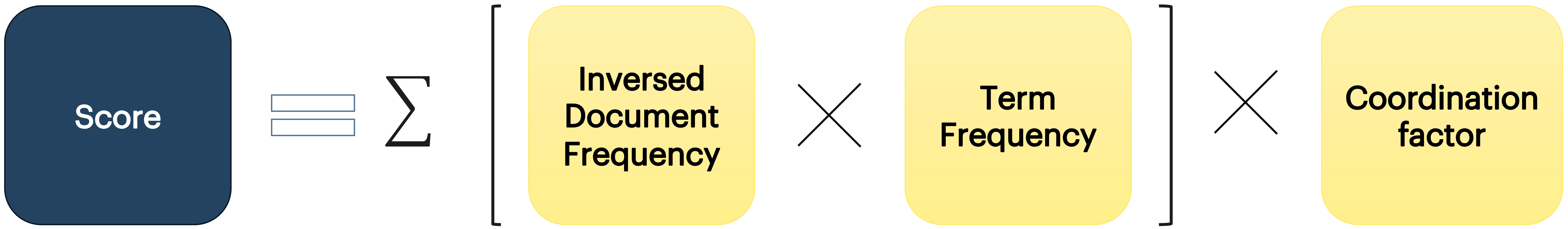
- Not the actual algorithm (simplified and changed for the presentation)
- Maths

BM25

$$S(D,Q) = \sum_{i=1}^{n} IDF(q_i) \,.\, \frac{f(q_i,D)\,.\,(k_1+1)}{f(q_i,D) + k_1\,.\,(1 - b + b.\frac{|D|}{avgdl})}$$

Source: https://www.elastic.co/fr/blog/how-to-improve-elasticsearch-search-relevance-with-boolean-queries
https://www.elastic.co/fr/blog/practical-bm25-part-2-the-bm25-algorithm-and-its-variables

Score



Source: https://www.youtube.com/watch?v=UhONe6GSfGQ

Example

Genre : Fantasy, Title : Hunger Games

Score = [IDF(fanta) . TF(fanta)] . CF

Genre : Fantasy, Title : Narnia

Score = [IDF(fanta) . TF(fanta)] . CF

Genre : Fantasy, Title : Fantastic Beasts and Where to Find Them

Score = [IDF(fanta) . TF(fanta)] . CF

Genre : Horror, Title : IT

Score = [IDF(fanta) . TF(fanta)] . CF

Reminder:

Score = [IDF(fanta) . TF(fanta)] . CF
TF = timesTermAppearsInDocument / numberOfTermsWithinTheDocument
IDF = log(totalDocuments / documentsContaining)
CF = Q(D) / Q

Example – TF

Genre : Fantasy, Title : Hunger Games

Score = [IDF(fanta) . 1] . CF

Genre : Fantasy, Title : Narnia

Score = [IDF(fanta) . 1] . CF

Genre : Fantasy, Title : Fantastic Beasts and Where to Find Them

Score = [IDF(fanta) . 2] . CF

Genre : Horror, Title : IT

Score = [IDF(fanta) . 0] . CF

Reminder:

Score = [IDF(fanta) . TF(fanta)] . CF
TF = timesTermAppearsInDocument / numberOfTermsWithinTheDocument
IDF = log(totalDocuments / documentsContaining)
CF = Q(D) / Q

Example - IDF

Genre : Fantasy, Title : Hunger Games

Score = [log(4/3) . 1] . CF

Genre : Fantasy, Title : Narnia

Score = [log(4/3) . 1] . CF

Genre : Fantasy, Title : Fantastic Beasts and Where to Find Them

Score = [log(4/3) . 2] . CF

Genre : Horror, Title : IT

Score = [log(4/3) . 0] . CF

Reminder:

Score = [IDF(fanta) . TF(fanta)] . CF
TF = timesTermAppearsInDocument / numberOfTermsWithinTheDocument
IDF = log(totalDocuments / documentsContaining)
CF = Q(D) / Q

Example

Genre : Fantasy, Title : Hunger Games

Score = [0,12 . 1] . CF

Genre : Fantasy, Title : Narnia

Score = [0,12 . 1] . CF

Genre : Fantasy, Title : Fantastic Beasts and Where to Find Them

Score = [0,12 . 2] . CF

Genre : Horror, Title : IT

Score = [0,12 . 0] . CF

Reminder:

Score = [IDF(fanta) . TF(fanta)] . CF
TF = timesTermAppearsInDocument / numberOfTermsWithinTheDocument
IDF = log(totalDocuments / documentsContaining)
CF = Q(D) / Q

Example - CF

Genre : Fantasy, Title : Hunger Games

Score = [0,12 . 1] . 1

Genre : Fantasy, Title : Narnia

Score = [0,12 . 1] . 1

Genre : Fantasy, Title : Fantastic Beasts and Where to Find Them

Score = [0,12 . 2] . 1

Genre : Horror, Title : IT

Score = [0,12 . 0] . 0

Reminder:

Score = [IDF(fanta) . TF(fanta)] . CF
TF = timesTermAppearsInDocument / numberOfTermsWithinTheDocument
IDF = log(totalDocuments / documentsContaining)
CF = Q(D) / Q

Example

Genre : Fantasy, Title : Hunger Games

Score = [0,12 . 1] . 1 = 0,12

Genre : Fantasy, Title : Narnia

Score = [0,12 . 1] . 1 = 0,12

Genre : Fantasy, Title : Fantastic Beasts and Where to Find Them

Score = [0,12 . 2] . 1 = 0,24

Genre : Horror, Title : IT

Score = [0,12 . 0] . 0 = 0

Reminder:

Score = [IDF(fanta) . TF(fanta)] . CF
TF = timesTermAppearsInDocument / numberOfTermsWithinTheDocument
IDF = log(totalDocuments / documentsContaining)
CF = Q(D) / Q

Example

Genre : Fantasy, Title : Hunger Games

Score = [(0,12 . 1) + (IDF(narnia) . TF(narnia))] . 0,5 =

Genre : Fantasy, Title : Narnia

Score = [(0,12 . 1) + (IDF(narnia) . TF(narnia))] . 1 =

Genre : Fantasy, Title : Fantastic Beasts and Where to Find Them

Score = [(0,12 . 2) + (IDF(narnia) . TF(narnia))] . 0,5 =

Genre : Horror, Title : IT

Score = [(0,12 . 0) + (IDF(narnia) . TF(narnia))] . 0 =

Reminder:

Score = [(IDF(fanta) . TF(fanta)) + (IDF(narnia) . TF(narnia))]. CF
TF = timesTermAppearsInDocument / numberOfTermsWithinTheDocument
IDF = log(totalDocuments / documentsContaining)
CF = Q(D) / Q

Example

Genre : Fantasy, Title : Hunger Games

Score = [(0,12 . 1) + (log(4/1) . 0)] . 0,5 =

Genre : Fantasy, Title : Narnia

Score = [(0,12 . 1) + (log(4/1) . 1)] . 1 =

Genre : Fantasy, Title : Fantastic Beasts and Where to Find Them

Score = [(0,12 . 2) + (log(4/1) . 0)] . 0,5 =

Genre : Horror, Title : IT

Score = [(0,12 . 0) + (log(4/1) . 0)] . 0 =

Reminder:

Score = [(IDF(fanta) . TF(fanta)) + (IDF(narnia) . TF(narnia))] . CF
TF = timesTermAppearsInDocument / numberOfTermsWithinTheDocument
IDF = log(totalDocuments / documentsContaining)
CF = Q(D) / Q

Example

Genre : Fantasy, Title : Hunger Games

Score = [(0,12 . 1) + (0,6 . 0)] . 0,5 =

Genre : Fantasy, Title : Narnia

Score = [(0,12 . 1) + (0,6 . 1)] . 1 =

Genre : Fantasy, Title : Fantastic Beasts and Where to Find Them

Score = [(0,12 . 2) + (0,6 . 0)] . 0,5 =

Genre : Horror, Title : IT

Score = [(0,12 . 0) + (0,6 . 0)] . 0 =

Reminder:

Score = [(IDF(fanta) . TF(fanta)) + (IDF(narnia) . TF(narnia))] . CF
TF = timesTermAppearsInDocument / numberOfTermsWithinTheDocument
IDF = log(totalDocuments / documentsContaining)
CF = Q(D) / Q

Example

Genre : Fantasy, Title : Hunger Games

Score = [(0,12 . 1) + (0,6 . 0)] . 0,5 = 0,06

Genre : Fantasy, Title : Narnia

Score = [(0,12 . 1) + (0,6 . 1)] . 1 = 0,72

Genre : Fantasy, Title : Fantastic Beasts and Where to Find Them

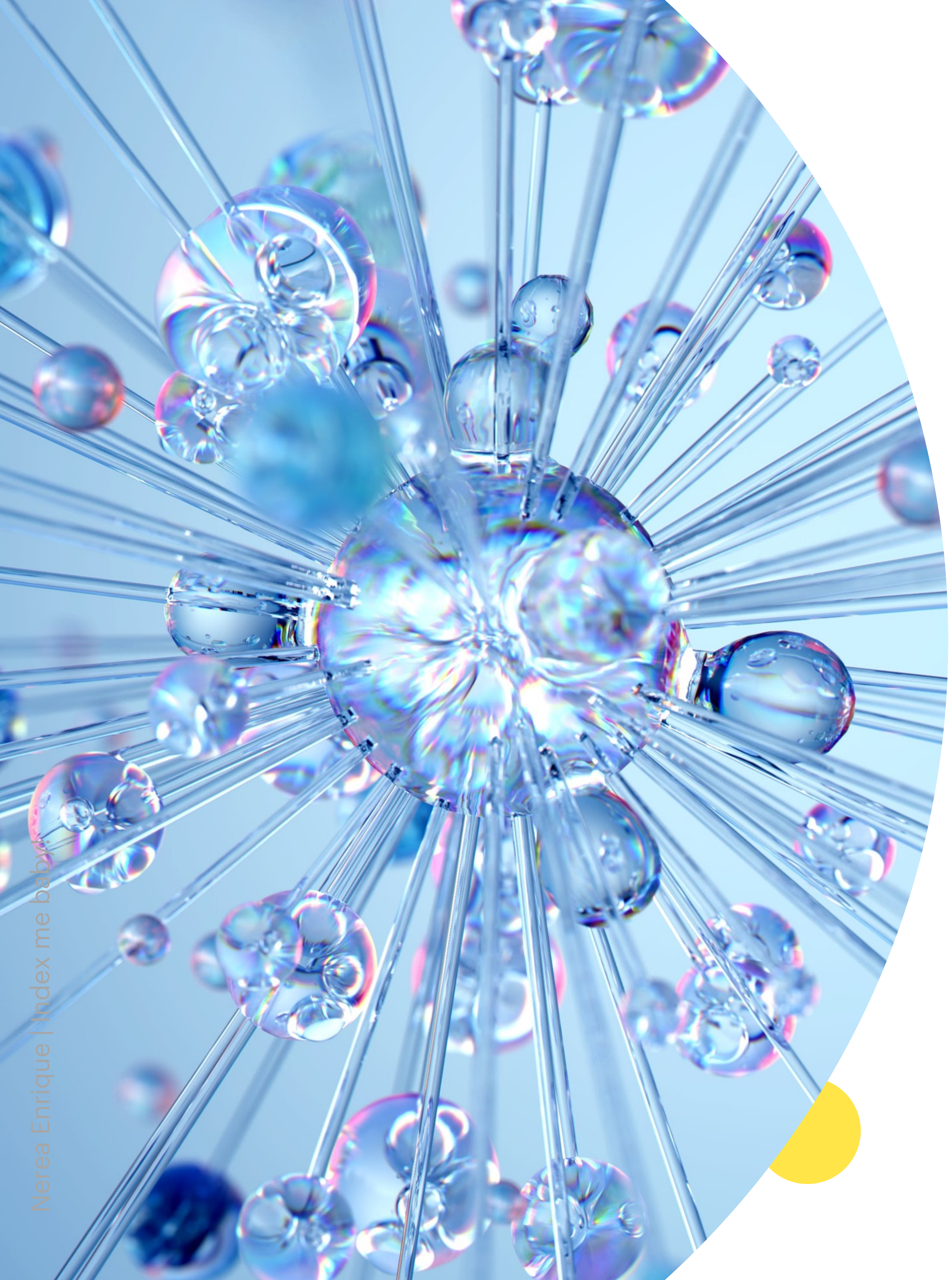
Score = [(0,12 . 2) + (0,6 . 0)] . 0,5 = 0,12

Genre : Horror, Title : IT

Score = [(0,12 . 0) + (0,6 . 0)] . 0 = 0

Reminder:

Score = [(IDF(fanta) . TF(fanta)) + (IDF(narnia) . TF(narnia))] . CF
TF = timesTermAppearsInDocument / numberOfTermsWithinTheDocument
IDF = log(totalDocuments / documentsContaining)
CF = Q(D) / Q



Elastic math

Some notions

- Score
- Inverse Document Frequency IDF
- Term Frequency TF
- Coordination factor

That’s not all folks

- Boosts
- Function score
- User input
- ...

Let’s analyze everything 🔍

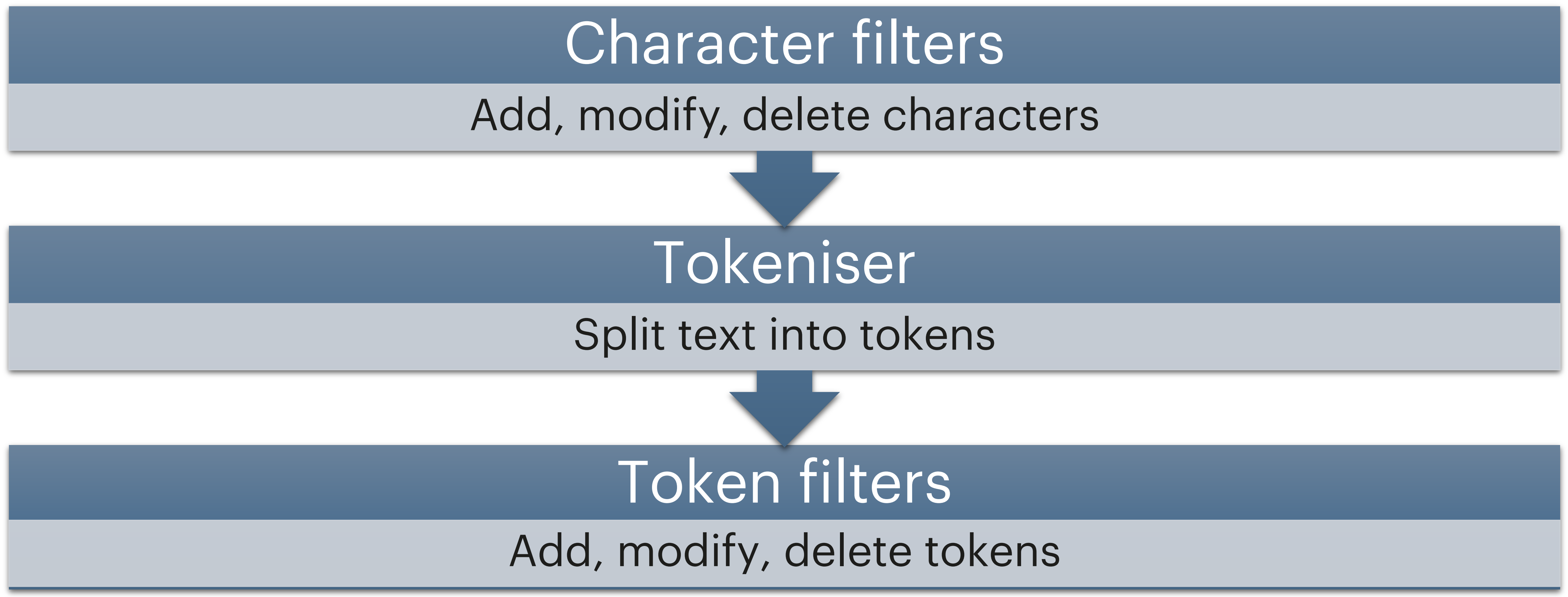
People talk with lots of words, how simplify a long, full phrase on some keywords that will be used for a search?

Let’s analyse everything

What can they do?

Let’s analyze everything 🔍

What can they do?



Example

Title: The Lord of the Rings

Title: The Great Gatsby

Title: The Adventures of Huckleberry Finn

Source: https://www.elastic.co/guide/en/elasticsearch/reference/current/analysis-charfilters.html

ekino.

Let’s analyze everything 🔍

Tokenisation

Title: The Lord of the Rings

Tokens = The, Lord, of, the, Rings

Title: The Great Gatsby

Tokens = The, Great, Gatsby

Title: The Adventures of Huckleberry Finn

Tokens = The, Adventures, of, Huckleberry, Finn

ekino.

Let's analyze everything 🔍

Token filter: lowercase

Title: The Lord of the Rings

Tokens = the, lord, of, the, rings

Title: The Great Gatsby

Tokens = the, great, gatsby

Title: The Adventures of Huckleberry Finn

Tokens = the, adventures, of, huckleberry, finn

Token filter: stop

Title : The Lord of the Rings

Tokens = lord, rings

Title : The Great Gatsby

Tokens = great, gatsby

Title : The Adventures of Huckleberry Finn

Tokens = adventures, huckleberry, finn

Let's analyze everything 🔍

Disclaimer

- Each index contains the same 15 books
- Each index is configured distinctly

Let's analyze everything 🔍

Query

```
{
  "query": {
    "match": {
      "description": "epic"
    }
  }
}
```

Let’s analyze everything 🔍

Tokens – ngram

Total: 7

Title: The Lord of the Rings
Description: The Lord of the Rings is an epic high fantasy novel by the English author and scholar J. R. R. Tolkien.

Score = 1.2574334

Title: The Silmarillion
Description: The Silmarillion is a collection of mythopoeic works by English writer J. R. R. Tolkien, edited and published posthumously by his son, Christopher Tolkien, in 1977, with assistance from Guy Gavriel Kay.

Score = 1.2202173

Title: The Hobbit
Description: The Hobbit, or There and Back Again is a children's fantasy novel by English author J. R. R. Tolkien.

Score = 0.70242006

…

The ngram tokenizer first breaks text down into words whenever it encounters one of a list of specified characters, then it emits N-grams of each word of the specified length.
N-grams are like a sliding window that moves across the word - a continuous sequence of characters of the specified length. They are useful for querying languages that don’t use spaces or that have long compound words, like German.

https://www.elastic.co/guide/en/elasticsearch/reference/current/analysis-ngram-tokenizer.html

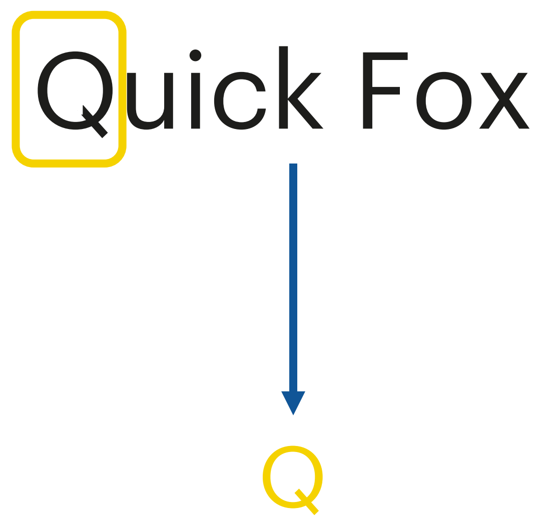
ekino.

Ngram

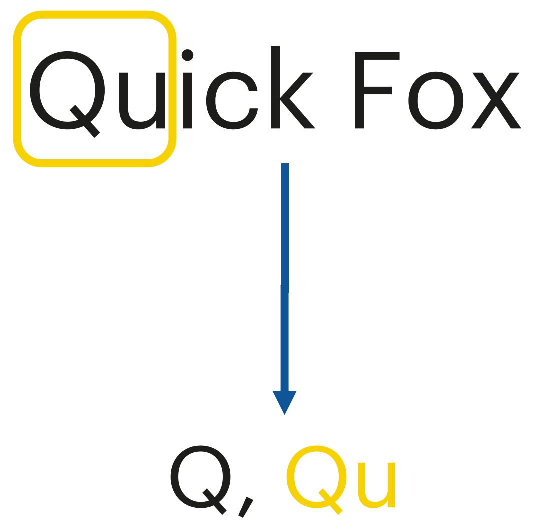
Quick Fox

ekino.

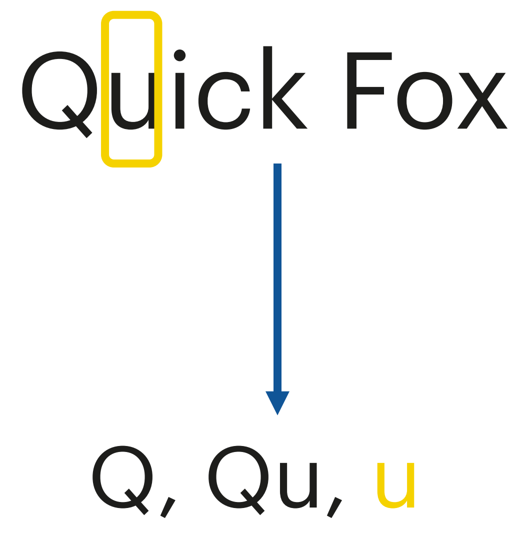
Ngram



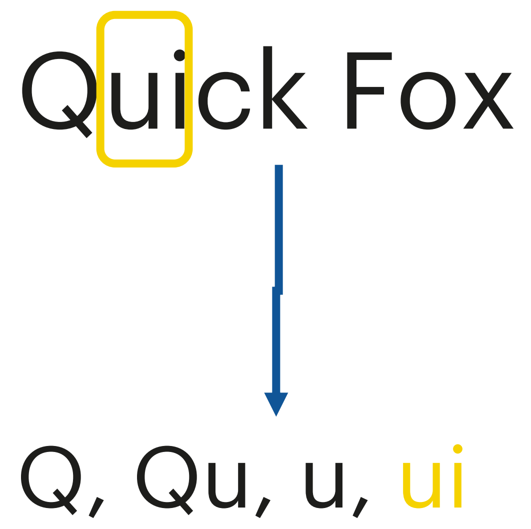
Ngram



Ngram



Ngram



Ngram

Quick Fox

$\downarrow$

Q, Qu, u, ui, i, ic, c, ck, k, "k ", " ", " F", F, Fo, o, ox, x

Ngram

$$\begin{tikzcd}\text{epic} \arrow[d] \\ \text{e, ep, p, pi, i, ic, c}\end{tikzcd}$$

ekino.

Let's analyze everything 🔍

Tokens – edge-gram

Total: 1

Title: The Lord of the Rings
Description: The Lord of the Rings is an epic high fantasy novel by the English author and scholar J. R. R. Tolkien.

Score = 6.6332088

The edge_ngram tokenizer first breaks text down into words whenever it encounters one of a list of specified characters, then it emits N-grams of each word where the start of the N-gram is anchored to the beginning of the word.

Edge N-Grams are useful for search-as-you-type queries.

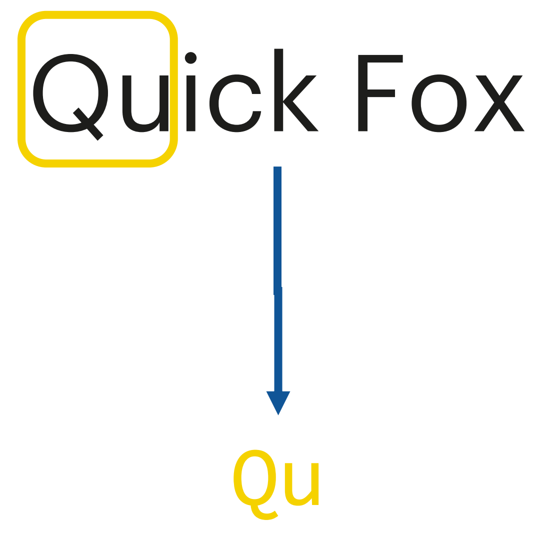
https://www.elastic.co/guide/en/elasticsearch/reference/current/analysis-edgengram-tokenizer.html

ekino.

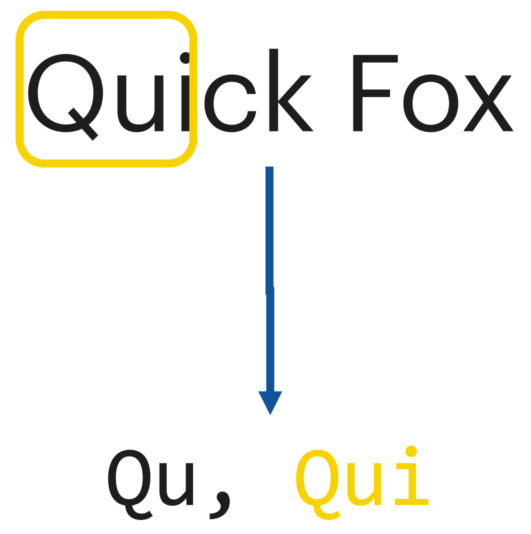
Edge-gram

Quick Fox

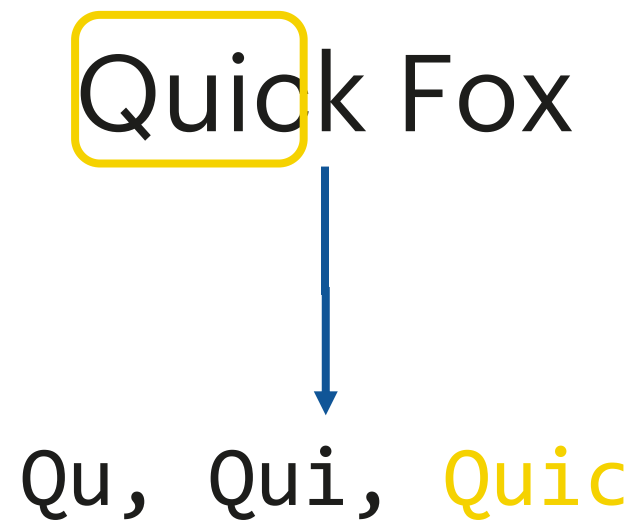
Edge-ngram



Edge-ngram



Edge-ngram



ekino.

Edge-ngram

Quick Fox

$\downarrow$

```
Qu, Qui, Quic, Quick, Fo, Fox, Foxe, Foxes
```

Edge-ngram

$$\begin{tikzcd}\text{epic} \arrow[d] \\ \text{ep, epi, epic}\end{tikzcd}$$

Let's analyze everything 🔍

Filters – lowercase

Total: 1

Title: The Lord of the Rings
Description: The Lord of the Rings is an epic high fantasy novel by the English author and scholar J. R. R. Tolkien.

Score = 2.0468292

ekino.

Let's analyze everything 🔍

Filters – stemmer

Total: 1

Title: The Lord of the Rings
Description: The Lord of the Rings is an epic high fantasy novel by the English author and scholar J. R. R. Tolkien.

Score = 2.048967

Filters – stop

Total: 1

Title: The Lord of the Rings
Description: The Lord of the Rings is an epic high fantasy novel by the English author and scholar J. R. R. Tolkien.

Score = 2.0478997

Let’s analyze everything 🔍

All filters with ngram

Total: 7

Title: The Grapes of Wrath
Description: The Grapes of Wrath is an American realist novel written by John Steinbeck and published in 1939. The book won the National Book Award and Pulitzer Prize for fiction, and it was cited prominently when Steinbeck was awarded the Nobel Prize in 1962.

Score = 0.75382936

Title: The Silmarillion
Description: The Silmarillion is a collection of mythopoeic works by English writer J. R. R. Tolkien, edited and published posthumously by his son, Christopher Tolkien, in 1977, with assistance from Guy Gavriel Kay.

Score = 0.7357905

Title: The Catcher in the Rye
Description: The Catcher in the Rye is a novel by J. D. Salinger, partially published in serial form in 1945–1946 and as a novel in 1951. It was originally intended for adults but is read by adolescents for its themes of angst, alienation, and as a critique on superficiality in society.

Score = 0.70242006

Title: The Lord of the Rings
Description: The Lord of the Rings is an epic high fantasy novel by the English author and scholar J. R. R. Tolkien.

Score = 0.63124937

...

Let's analyze everything 🔍

Books

Setting	Default settings	Ngram	Edge-ngram	Case sensitive	Stemmer	Stop	All
Score	2.048967	1.2574334	6.6332088	2.0468292	2.048967	2.0478997	0.63124937
Total	1	7	1	1	1	1	7

Books - FR

```
{
  "settings": {
    "analysis": {
      "filter": {
        "french_elision": {
          "type": "elision",
          "articles_case": true,
          "articles": [
            "l", "m", "t", "qu", "n", "s",
            "j", "d", "c", "jusqu",
            "quoiqu", "lorsqu", "puisqu"
          ]
        },
        "french_stop": {
          "type": "stop",
          "stopwords":  "_french_"
        },
        "french_keywords": {
          "type": "keyword_marker",
          "keywords": [ "Example" ]
        },
        "french_stemmer": {
          "type": "stemmer",
          "language": "light_french"
        }
      },
      "analyzer": {
        "rebuilt_french": {
          "tokenizer": "standard",
          "filter": [
            "french_elision",
            "lowercase",
            "french_stop",
            "french_keywords",
            "french_stemmer"
          ]
        }
      }
    }
  }
}
```

Source : https://www.elastic.co/guide/en/elasticsearch/reference/current/analysis-lang-analyzer.html#french-analyzer

ekino.

Let’s analyze everything 🔍

Books - EN

```
{
  "settings": {
    "analysis": {
      "filter": {
        "english_stop": {
          "type": "stop",
          "stopwords": "_english_"
        },
        "english_keywords": {
          "type": "keyword_marker",
          "keywords": ["example"]
        },
        "english_stemmer": {
          "type": "stemmer",
          "language": "english"
        },
        "english_possessive_stemmer": {
          "type": "stemmer",
          "language": "possessive_english"
        }
      },
      "analyzer": {
        "rebuilt_english": {
          "tokenizer": "standard",
          "filter": [
            "english_possessive_stemmer",
            "lowercase",
            "english_stop",
            "english_keywords",
            "english_stemmer"
          ]
        }
      }
    }
  }
}
```

Source : https://www.elastic.co/guide/en/elasticsearch/reference/current/analysis-lang-analyzer.html#english-analyzer

What about Drupal?

With ElasticSearch

Context

- Elastic search
- `search_api` module
- `elasticsearch_connector` module

What should we do?

- Before the index is prepared
- Access to the index
- Change its settings

Subscribe?

- `BuildIndexParamsEvent`
- `BuildSearchParamsEvent`
- `PrepareIndexEvent`
- `PrepareIndexMappingEvent`
- `PrepareMappingEvent`
- `PrepareSearchQueryEventEvent`

Subscribe!

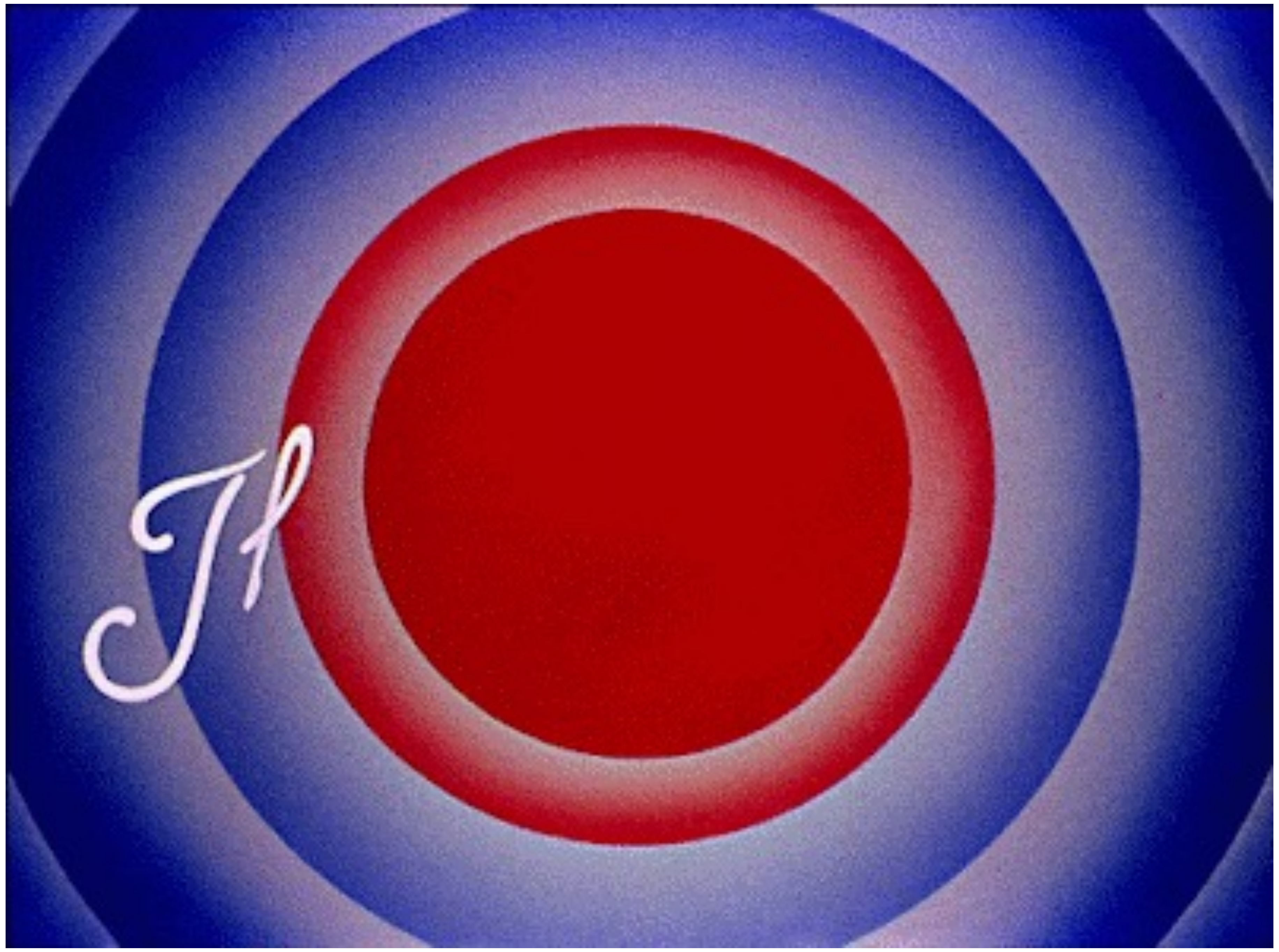
```
class ChangeIndexEventSubscriber implements EventSubscriberInterface {

  public static function getSubscribedEvents() {
    return [
      PrepareIndexEvent::PREPARE_INDEX => 'prepareIndex',
    ];
  }
…
}
```

Set it up!

```
class ChangeIndexEventSubscriber implements EventSubscriberInterface {
    public function prepareIndex(PrepareIndexEvent $event) {
        $indexConfig = $event->getIndexConfig();
        $indexConfig['body']['settings']['analysis']['analyzer'] = [
            'whitespace_lowercase' => [
                'tokenizer' => 'whitespace',
                'filter' => [ 'lowercase' ],
            ],
        ];
        ...
        $event->setIndexConfig($indexConfig);
    }
...
}
```

And...

And...



ekino.

Conclusion

- Anything could change the score
- Refine as you go

Merci

pour votre écoute !

Github

ekino.

Sources 1/2

https://en.wikipedia.org/wiki/Vector_space_model

https://lucene.apache.org/core/3_5_0/api/core/org/apache/lucene/search/Similarity.html

https://www.youtube.com/watch?v=UhONe6GSfGQ&list=PL9zDdgiGjkIcN0fBpm7NX3ZC5Fh7e0OMj

https://www.elastic.co/guide/en/elasticsearch/reference/current/analyzer-anatomy.html

https://www.youtube.com/watch?v=ajNfOPeWiAY

https://www.elastic.co/fr/blog/how-to-improve-elasticsearch-search-relevance-with-boolean-queries

https://en.wikipedia.org/wiki/Inverted_index

Sources 2/2

https://www.elastic.co/fr/blog/how-to-improve-elasticsearch-search-relevance-with-boolean-queries

https://www.elastic.co/fr/blog/practical-bm25-part-2-the-bm25-algorithm-and-its-variables

https://medium.com/elasticsearch/introduction-to-analysis-and-analyzers-in-elasticsearch-4cf24d49ddab