67-300 SEARCH ENGINES

BIG PICTURE

LECTURER: JOAO PALOTTI (JPALOTTI@ANDREW.CMU.EDU)

15TH MARCH 2017

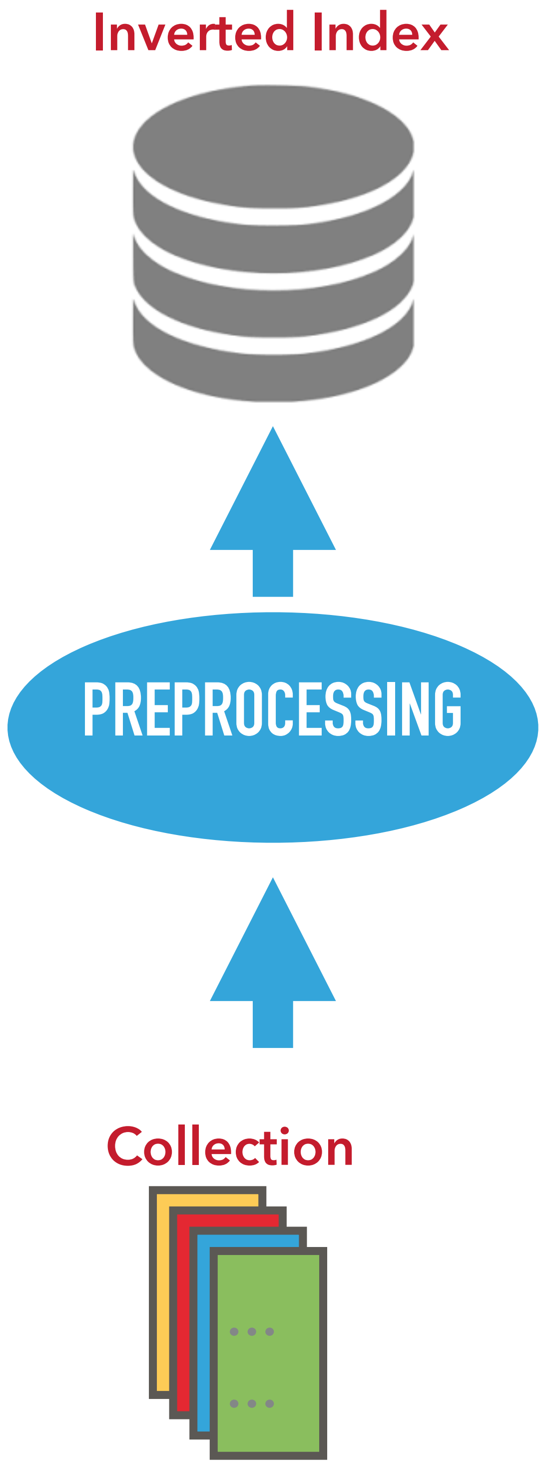
LECTURE GOALS

- See the big picture of the search process
- Document representation
- Indexing (Inverted Index)
- Boolean Queries
- Spellcheck (bonus part)

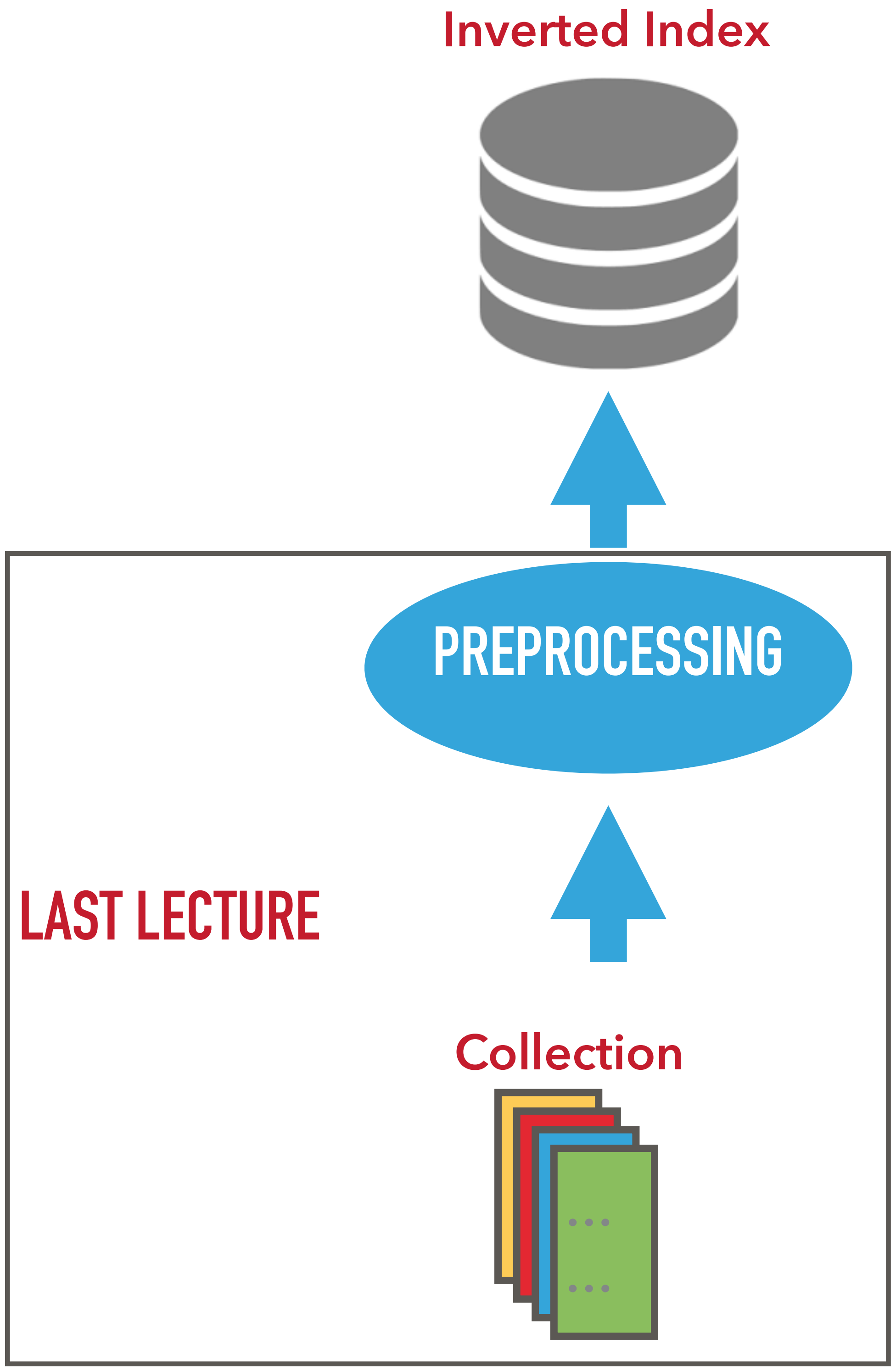

KEEP
CALM
AND
SEE THE
BIG PICTURE

BIG PICTURE

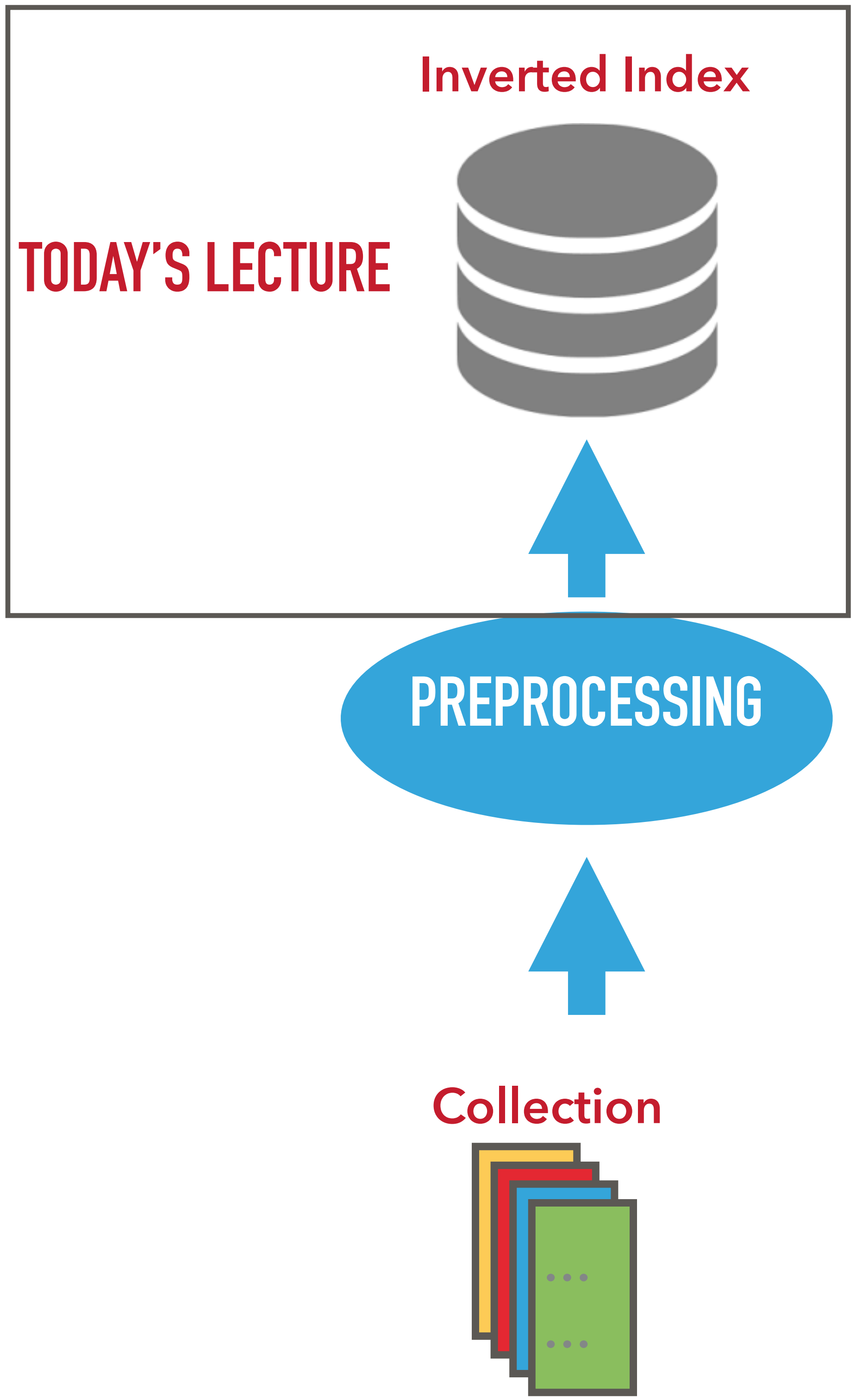
BIG PICTURE



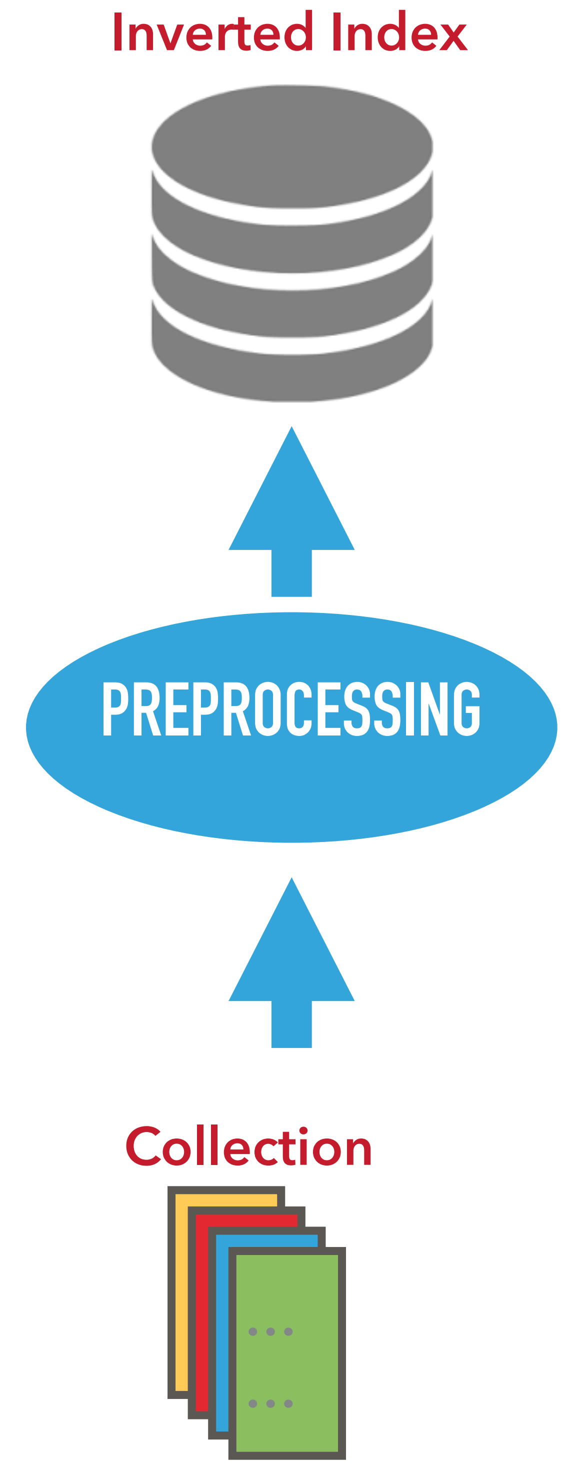
BIG PICTURE



BIG PICTURE

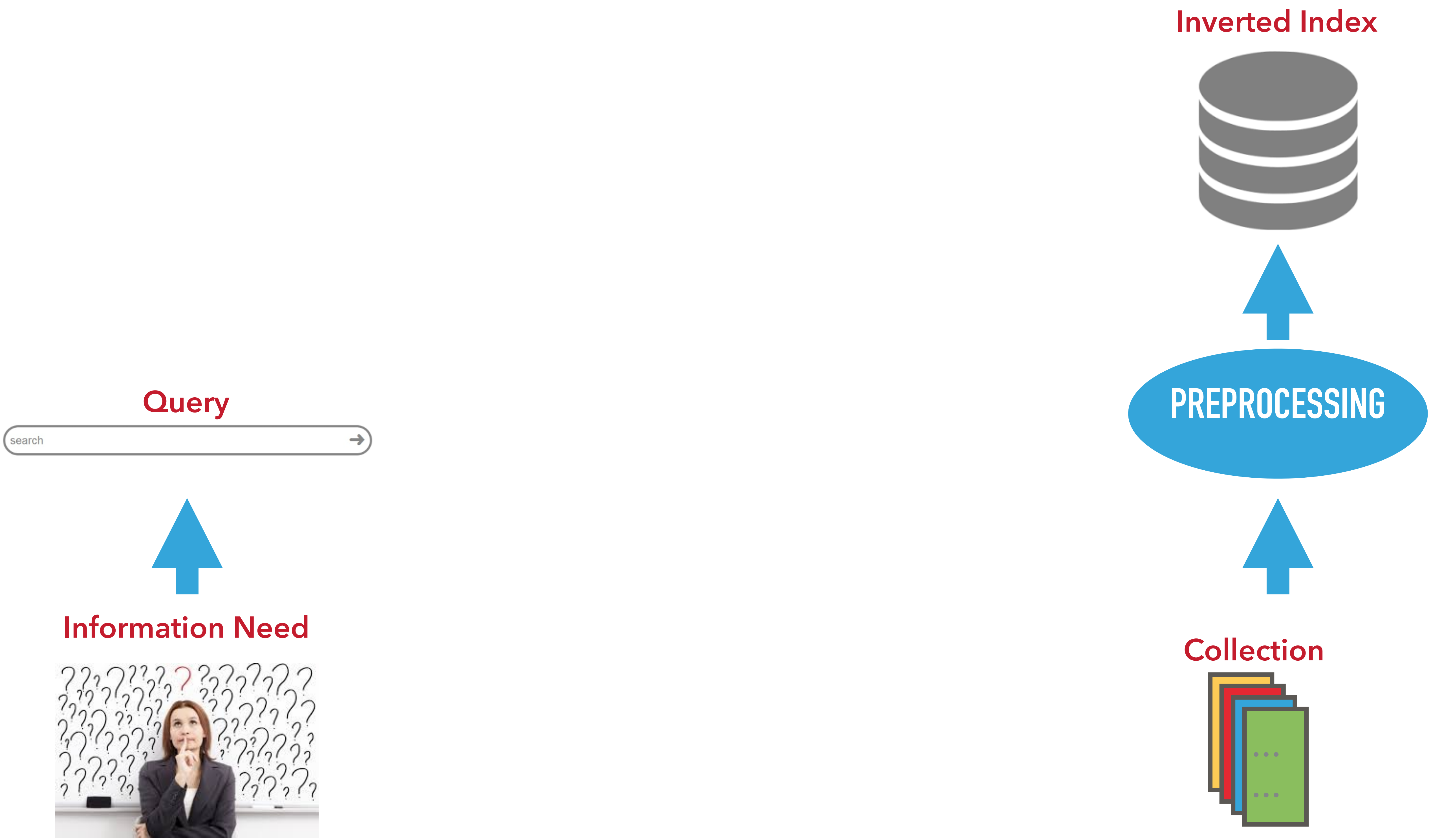


BIG PICTURE

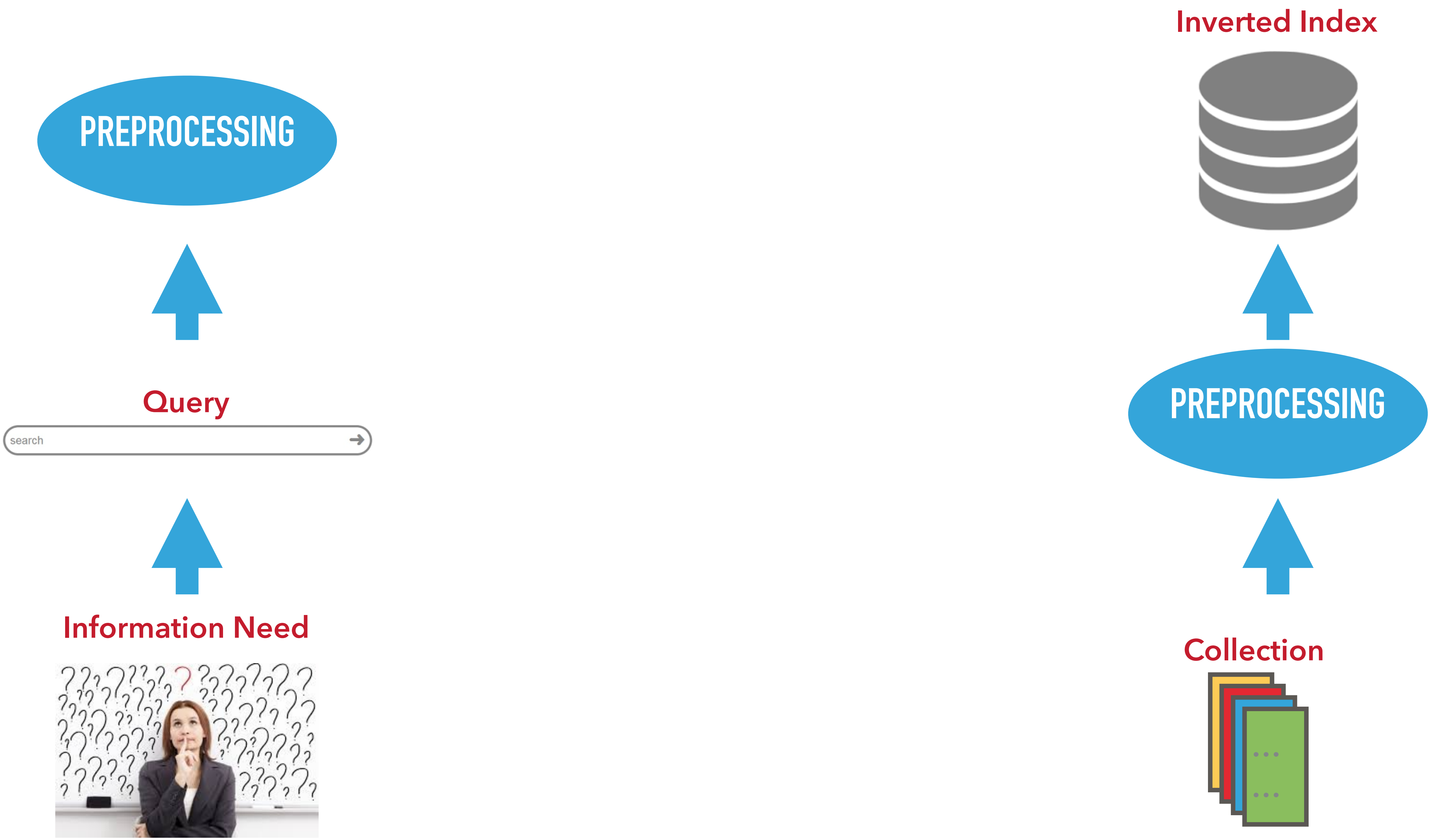


Information Need

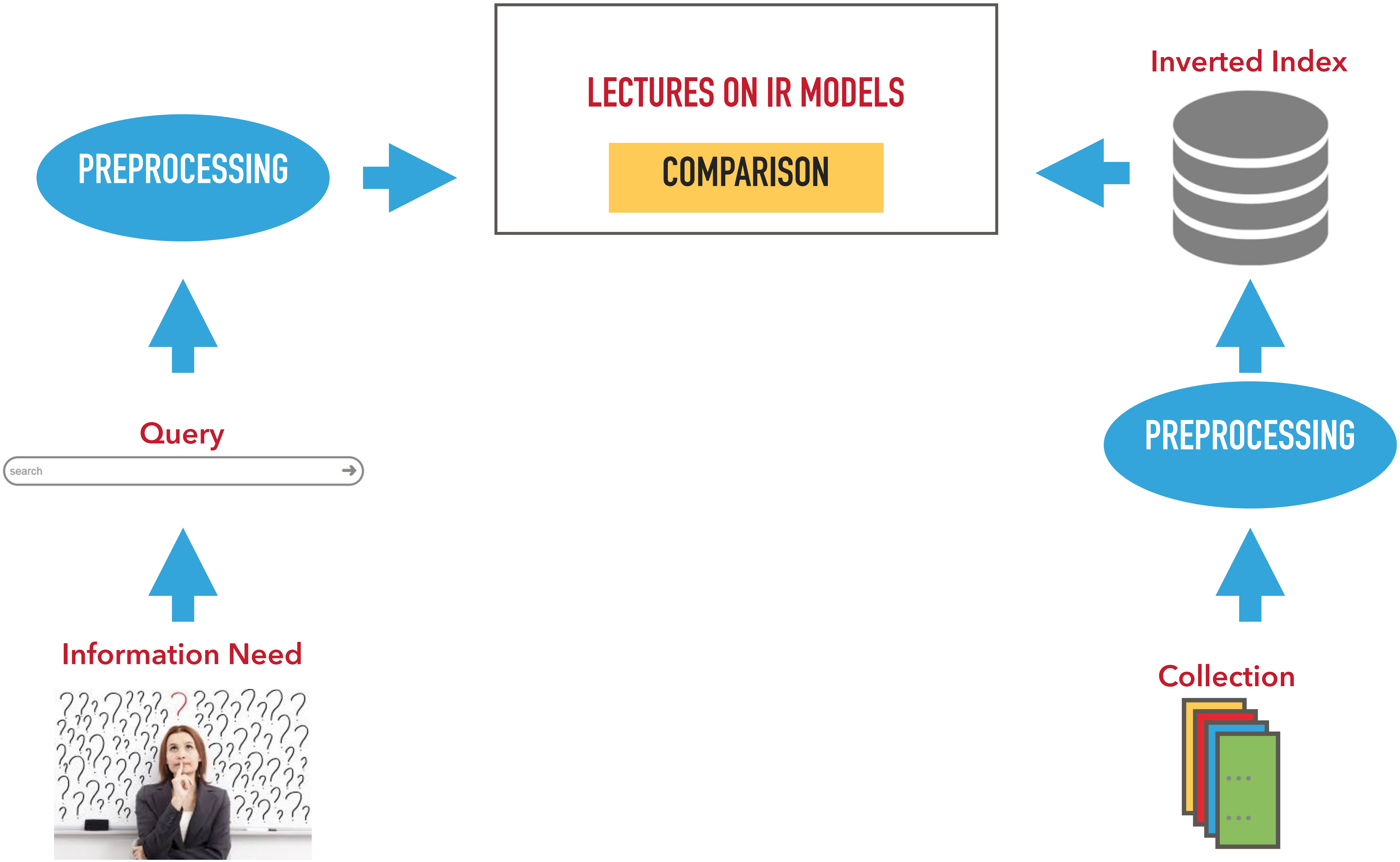
BIG PICTURE


Inverted Index
PREPROCESSING
Collection
Query
search
Information Need

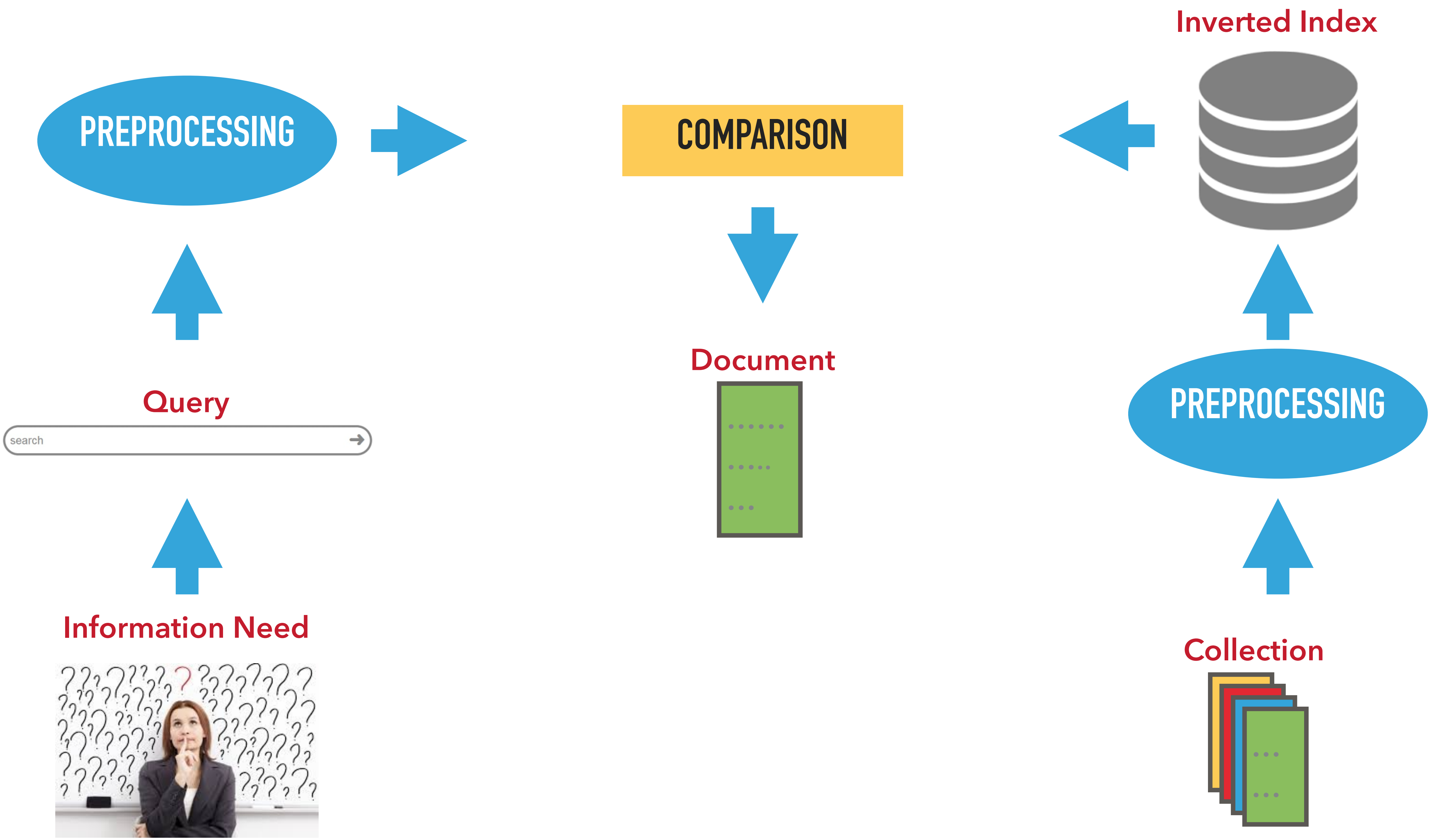
BIG PICTURE


Inverted Index
PREPROCESSING
PREPROCESSING
Query
search
Information Need
Collection

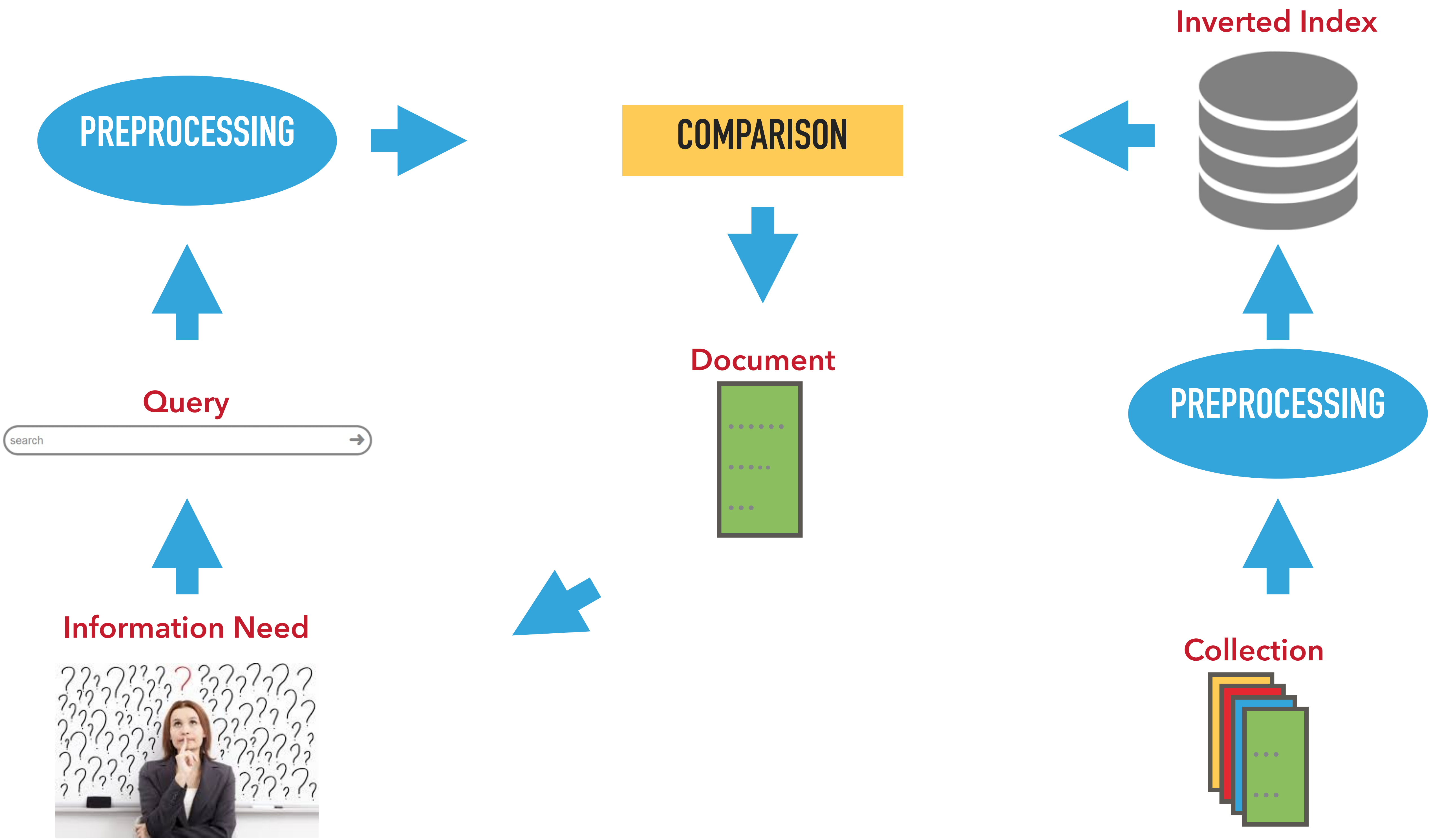
BIG PICTURE


LECTURES ON IR MODELS
COMPARISON
Inverted Index
PREPROCESSING
Query
search
Information Need
PREPROCESSING
Collection

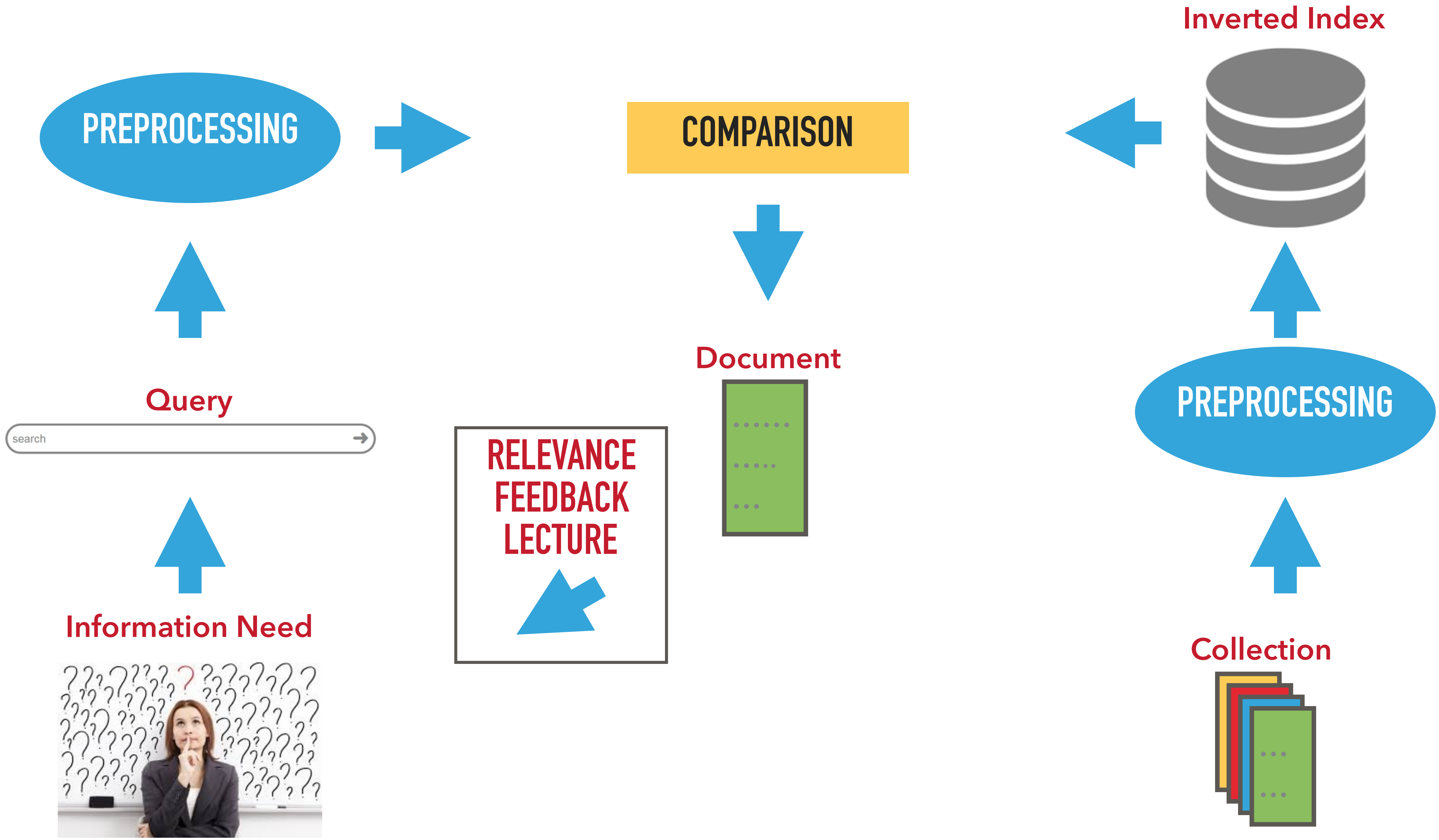
BIG PICTURE


Inverted Index
PREPROCESSING
COMPARISON
Document
Query
search
PREPROCESSING
Information Need
Collection

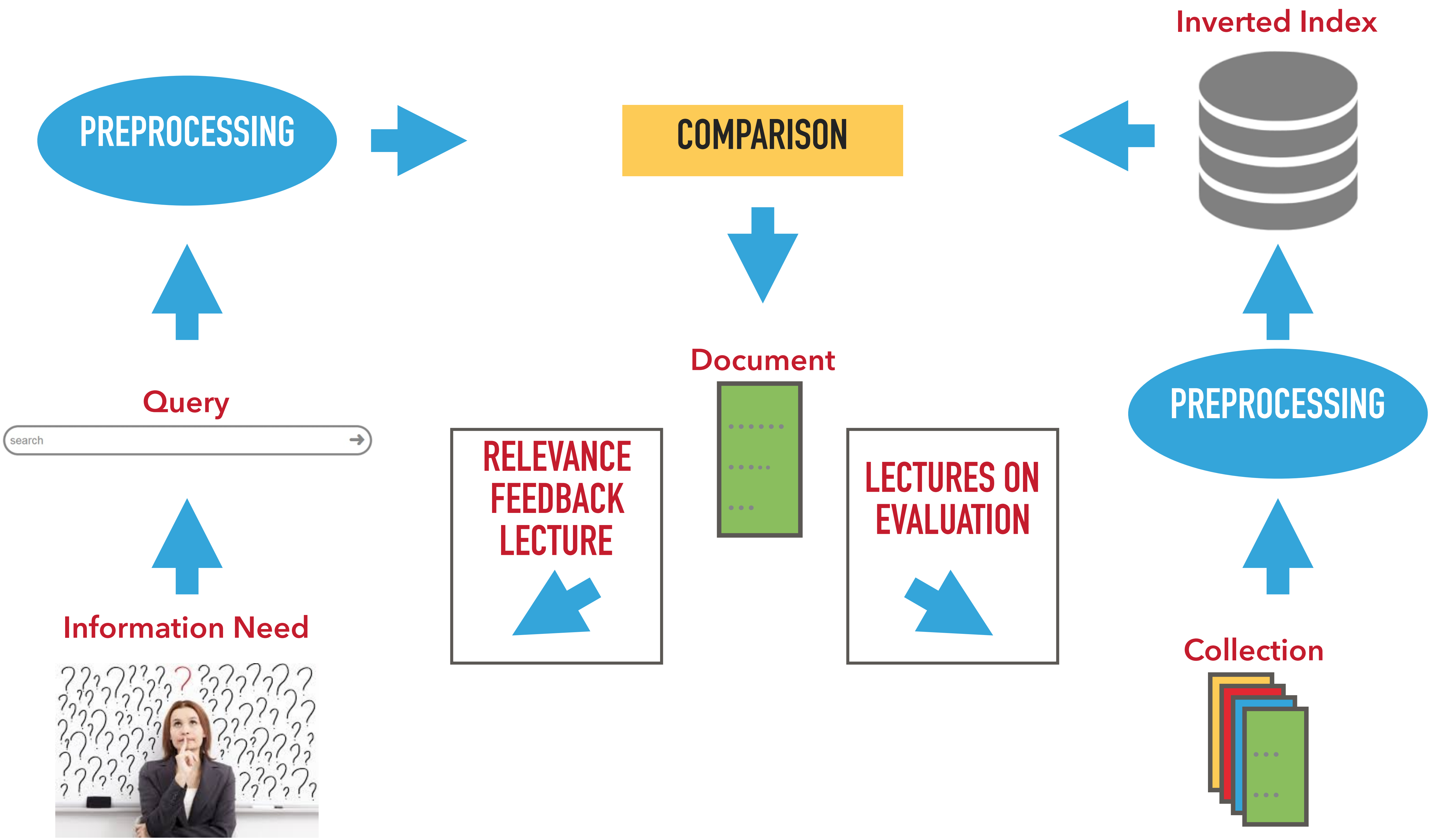
BIG PICTURE


Inverted Index
PREPROCESSING
COMPARISON
Document
Query
search
PREPROCESSING
Information Need
Collection

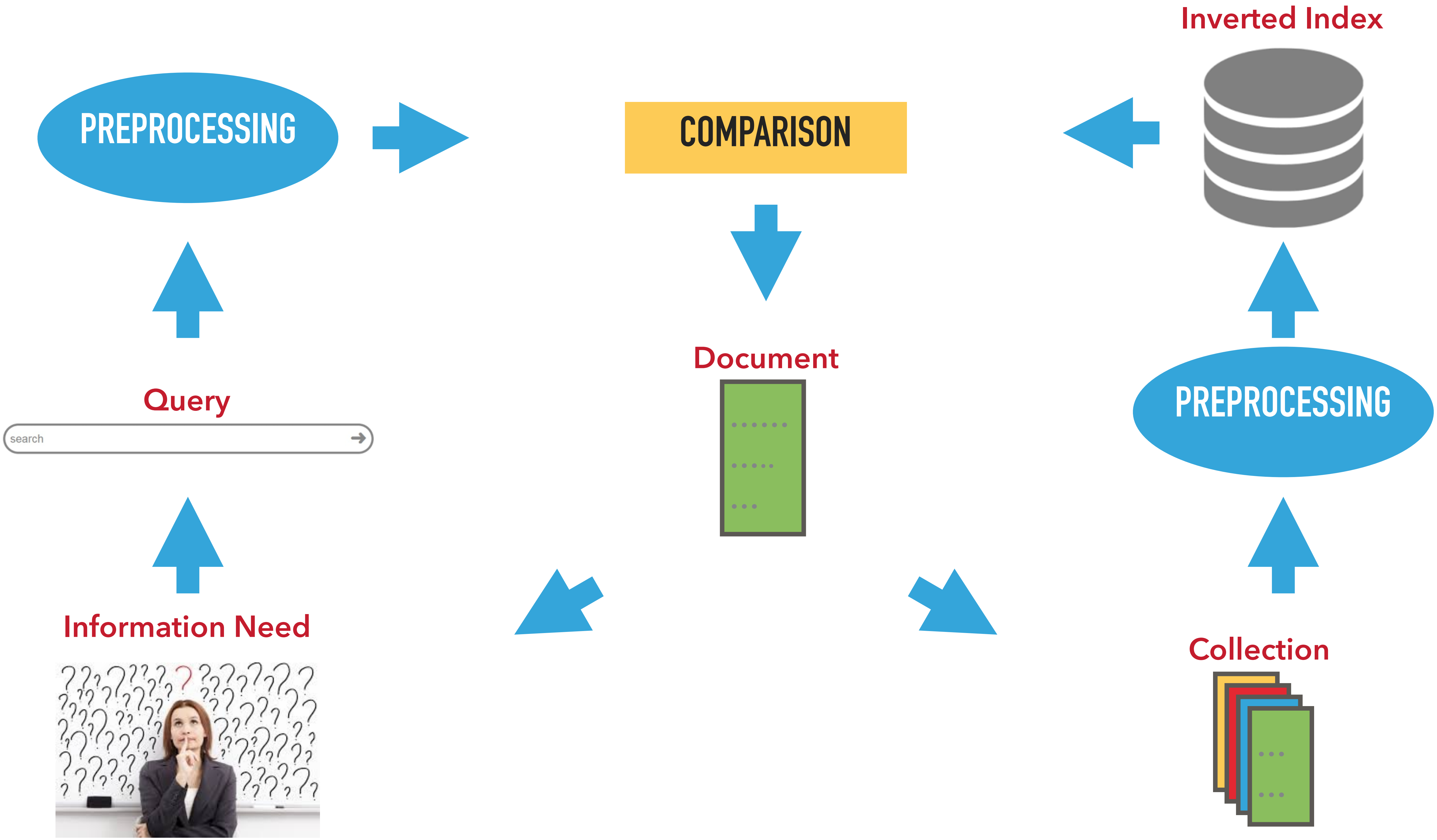
BIG PICTURE


Inverted Index
PREPROCESSING
COMPARISON
Document
Query
search
PREPROCESSING
RELEVANCE FEEDBACK LECTURE
Information Need
Collection

BIG PICTURE


Inverted Index
PREPROCESSING
COMPARISON
Document
Query
search
PREPROCESSING
RELEVANCE FEEDBACK LECTURE
LECTURES ON EVALUATION
Information Need
Collection

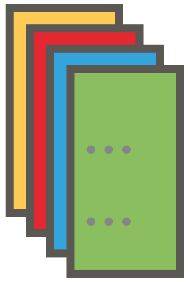
BIG PICTURE



Inverted Index
PREPROCESSING
COMPARISON
Document
Query
search
PREPROCESSING
Information Need
Collection

DOCUMENT REPRESENTATION

RECAP: WHAT ARE DOCUMENTS?



- By **documents** we mean whatever units we have decided to build a retrieval system over. They might be individual memos or chapters of a book (Stanford IR Book)
- Focus on text documents:
  - unstructured or semi-structured data

CASE 2 - BIG PICTURE

Collection

LECTURE 2 - BIG PICTURE

Collection

- Representation:
    - Unstructured data: Title, Author names, short description
    - Structured data: language, price, book type, ISBN
    - Table of contents? Reviews?

LECTURE 2 - BIG PICTURE

Collection

<doc>
<title> Frankenstein </title>
<authors> Mary Shelley </authors>
<lang> en </lang>
<price> 6.00 </price>
<type> Type 1 – Paperback </type>
<isbn_code> 978-0486282114 </
isbn_code>
</doc>

Concept: Searching Fields

Collection

The Anatomy of a Large-Scale Hypertextual Web Search Engine

Sergey Brin and Lawrence Page

Computer Science Department,
Stanford University, Stanford, CA 94305, USA
sergey@cs.stanford.edu and page@cs.stanford.edu

Abstract

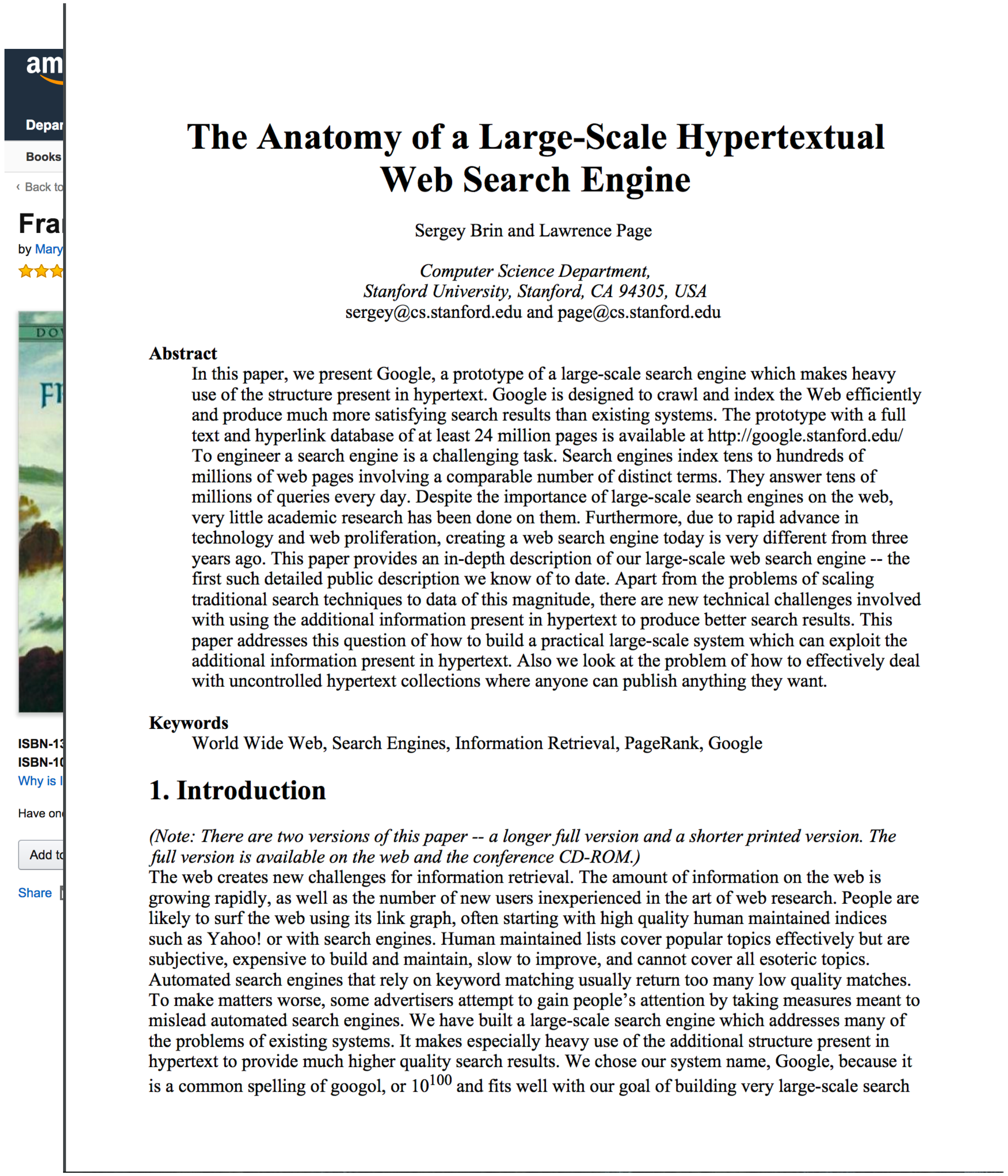
In this paper, we present Google, a prototype of a large-scale search engine which makes heavy use of the structure present in hypertext. Google is designed to crawl and index the Web efficiently and produce much more satisfying search results than existing systems. The prototype with a full text and hyperlink database of at least 24 million pages is available at http://google.stanford.edu/
To engineer a search engine is a challenging task. Search engines index tens to hundreds of millions of web pages involving a comparable number of distinct terms. They answer tens of millions of queries every day. Despite the importance of large-scale search engines on the web, very little academic research has been done on them. Furthermore, due to rapid advance in technology and web proliferation, creating a web search engine today is very different from three years ago. This paper provides an in-depth description of our large-scale web search engine -- the first such detailed public description we know of to date. Apart from the problems of scaling traditional search techniques to data of this magnitude, there are new technical challenges involved with using the additional information present in hypertext to produce better search results. This paper addresses this question of how to build a practical large-scale system which can exploit the additional information present in hypertext. Also we look at the problem of how to effectively deal with uncontrolled hypertext collections where anyone can publish anything they want.

Keywords

World Wide Web, Search Engines, Information Retrieval, PageRank, Google

1. Introduction

(Note: There are two versions of this paper -- a longer full version and a shorter printed version. The full version is available on the web and the conference CD-ROM.)
The web creates new challenges for information retrieval. The amount of information on the web is growing rapidly, as well as the number of new users inexperienced in the art of web research. People are likely to surf the web using its link graph, often starting with high quality human maintained indices such as Yahoo! or with search engines. Human maintained lists cover popular topics effectively but are subjective, expensive to build and maintain, slow to improve, and cannot cover all esoteric topics. Automated search engines that rely on keyword matching usually return too many low quality matches. To make matters worse, some advertisers attempt to gain people's attention by taking measures meant to mislead automated search engines. We have built a large-scale search engine which addresses many of the problems of existing systems. It makes especially heavy use of the additional structure present in hypertext to provide much higher quality search results. We chose our system name, Google, because it is a common spelling of googol, or 10^{100} and fits well with our goal of building very large-scale search



The Anatomy of a Large-Scale Hypertextual Web Search Engine

Sergey Brin and Lawrence Page

Computer Science Department,
Stanford University, Stanford, CA 94305, USA
sergey@cs.stanford.edu and page@cs.stanford.edu

Abstract

In this paper, we present Google, a prototype of a large-scale search engine which makes heavy use of the structure present in hypertext. Google is designed to crawl and index the Web efficiently and produce much more satisfying search results than existing systems. The prototype with a full text and hyperlink database of at least 24 million pages is available at http://google.stanford.edu/ To engineer a search engine is a challenging task. Search engines index tens to hundreds of millions of web pages involving a comparable number of distinct terms. They answer tens of millions of queries every day. Despite the importance of large-scale search engines on the web, very little academic research has been done on them. Furthermore, due to rapid advance in technology and web proliferation, creating a web search engine today is very different from three years ago. This paper provides an in-depth description of our large-scale web search engine -- the first such detailed public description we know of to date. Apart from the problems of scaling traditional search techniques to data of this magnitude, there are new technical challenges involved with using the additional information present in hypertext to produce better search results. This paper addresses this question of how to build a practical large-scale system which can exploit the additional information present in hypertext. Also we look at the problem of how to effectively deal with uncontrolled hypertext collections where anyone can publish anything they want.

Keywords

World Wide Web, Search Engines, Information Retrieval, PageRank, Google

1. Introduction

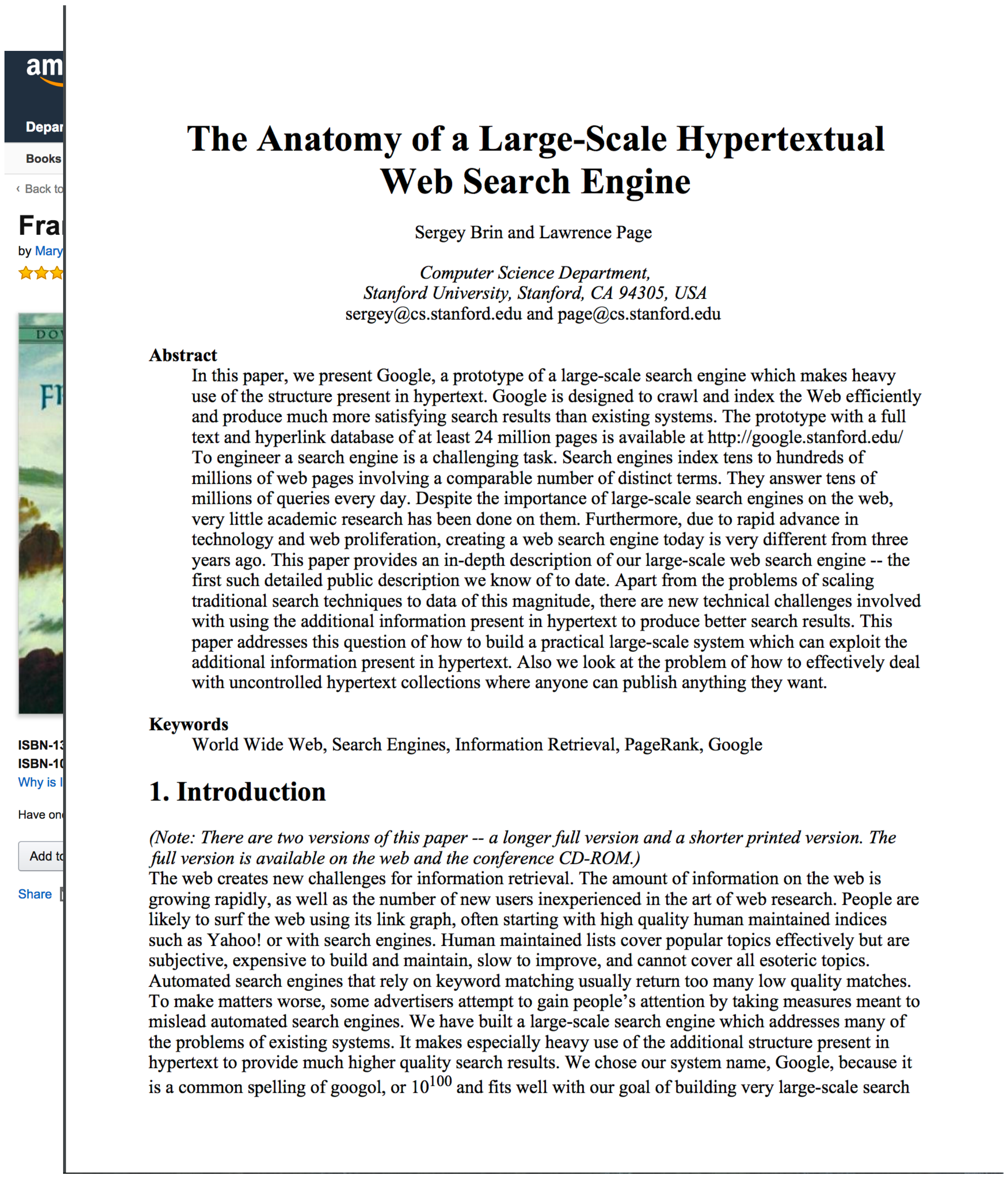
(Note: There are two versions of this paper -- a longer full version and a shorter printed version. The full version is available on the web and the conference CD-ROM.)
The web creates new challenges for information retrieval. The amount of information on the web is growing rapidly, as well as the number of new users inexperienced in the art of web research. People are likely to surf the web using its link graph, often starting with high quality human maintained indices such as Yahoo! or with search engines. Human maintained lists cover popular topics effectively but are subjective, expensive to build and maintain, slow to improve, and cannot cover all esoteric topics. Automated search engines that rely on keyword matching usually return too many low quality matches. To make matters worse, some advertisers attempt to gain people's attention by taking measures meant to mislead automated search engines. We have built a large-scale search engine which addresses many of the problems of existing systems. It makes especially heavy use of the additional structure present in hypertext to provide much higher quality search results. We chose our system name, Google, because it is a common spelling of googol, or 10^{100} and fits well with our goal of building very large-scale search

- Representation:

  1. Title, Authors, Abstract, Keywords, Chapters
  2. Title + Abstract, Text, Authors
  3. Title + Abstract + Text, Authors
  4. ???

Concept: Searching Fields

Collection



The Anatomy of a Large-Scale Hypertextual Web Search Engine

Sergey Brin and Lawrence Page

Computer Science Department,
Stanford University, Stanford, CA 94305, USA
sergey@cs.stanford.edu and page@cs.stanford.edu

Abstract

In this paper, we present Google, a prototype of a large-scale search engine which makes heavy use of the structure present in hypertext. Google is designed to crawl and index the Web efficiently and produce much more satisfying search results than existing systems. The prototype with a full text and hyperlink database of at least 24 million pages is available at http://google.stanford.edu/ To engineer a search engine is a challenging task. Search engines index tens to hundreds of millions of web pages involving a comparable number of distinct terms. They answer tens of millions of queries every day. Despite the importance of large-scale search engines on the web, very little academic research has been done on them. Furthermore, due to rapid advance in technology and web proliferation, creating a web search engine today is very different from three years ago. This paper provides an in-depth description of our large-scale web search engine -- the first such detailed public description we know of to date. Apart from the problems of scaling traditional search techniques to data of this magnitude, there are new technical challenges involved with using the additional information present in hypertext to produce better search results. This paper addresses this question of how to build a practical large-scale system which can exploit the additional information present in hypertext. Also we look at the problem of how to effectively deal with uncontrolled hypertext collections where anyone can publish anything they want.

Keywords

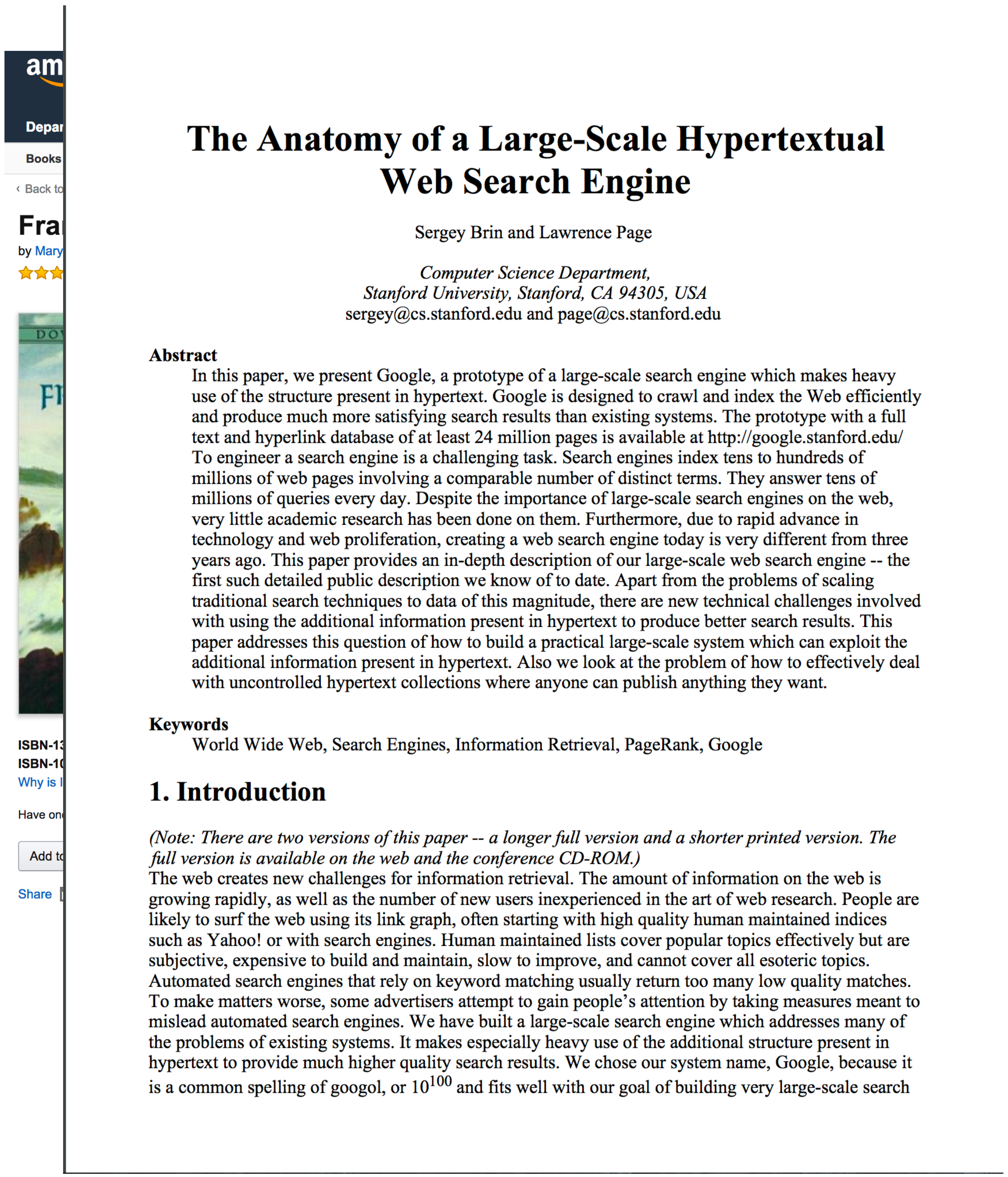
World Wide Web, Search Engines, Information Retrieval, PageRank, Google

1. Introduction

(Note: There are two versions of this paper -- a longer full version and a shorter printed version. The full version is available on the web and the conference CD-ROM.)
The web creates new challenges for information retrieval. The amount of information on the web is growing rapidly, as well as the number of new users inexperienced in the art of web research. People are likely to surf the web using its link graph, often starting with high quality human maintained indices such as Yahoo! or with search engines. Human maintained lists cover popular topics effectively but are subjective, expensive to build and maintain, slow to improve, and cannot cover all esoteric topics. Automated search engines that rely on keyword matching usually return too many low quality matches. To make matters worse, some advertisers attempt to gain people's attention by taking measures meant to mislead automated search engines. We have built a large-scale search engine which addresses many of the problems of existing systems. It makes especially heavy use of the additional structure present in hypertext to provide much higher quality search results. We chose our system name, Google, because it is a common spelling of googol, or 10^{100} and fits well with our goal of building very large-scale search

<doc>
<title> The anatomy of a large-scale Hypertextual Web Search Engine
</title>
<author> Sergey Brin </author>
<author> Lawrence Page </author>
<address> Computer Science Department, Stanford University, Stanford, CA, 94305, USA </address>
<email> sergey@cs.stanford.edu </email>
<email> page@cs.stanford.edu <email>
<abstract> In this paper, we present Google, a prototype of a large-scale search engine…
<abstract>
<keyword> World Wide Web </keyword>
<keyword> Search Engines </keyword>
<text> The web creates new challenges… </text>
</doc>

Collection



The Anatomy of a Large-Scale Hypertextual Web Search Engine

Sergey Brin and Lawrence Page

Computer Science Department,
Stanford University, Stanford, CA 94305, USA
sergey@cs.stanford.edu and page@cs.stanford.edu

Abstract

In this paper, we present Google, a prototype of a large-scale search engine which makes heavy use of the structure present in hypertext. Google is designed to crawl and index the Web efficiently and produce much more satisfying search results than existing systems. The prototype with a full text and hyperlink database of at least 24 million pages is available at http://google.stanford.edu/ To engineer a search engine is a challenging task. Search engines index tens to hundreds of millions of web pages involving a comparable number of distinct terms. They answer tens of millions of queries every day. Despite the importance of large-scale search engines on the web, very little academic research has been done on them. Furthermore, due to rapid advance in technology and web proliferation, creating a web search engine today is very different from three years ago. This paper provides an in-depth description of our large-scale web search engine -- the first such detailed public description we know of to date. Apart from the problems of scaling traditional search techniques to data of this magnitude, there are new technical challenges involved with using the additional information present in hypertext to produce better search results. This paper addresses this question of how to build a practical large-scale system which can exploit the additional information present in hypertext. Also we look at the problem of how to effectively deal with uncontrolled hypertext collections where anyone can publish anything they want.

Keywords

World Wide Web, Search Engines, Information Retrieval, PageRank, Google

1. Introduction

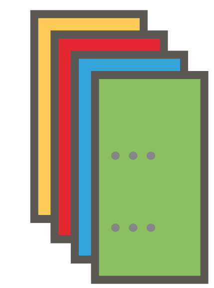
(Note: There are two versions of this paper -- a longer full version and a shorter printed version. The full version is available on the web and the conference CD-ROM.)
The web creates new challenges for information retrieval. The amount of information on the web is growing rapidly, as well as the number of new users inexperienced in the art of web research. People are likely to surf the web using its link graph, often starting with high quality human maintained indices such as Yahoo! or with search engines. Human maintained lists cover popular topics effectively but are subjective, expensive to build and maintain, slow to improve, and cannot cover all esoteric topics. Automated search engines that rely on keyword matching usually return too many low quality matches. To make matters worse, some advertisers attempt to gain people's attention by taking measures meant to mislead automated search engines. We have built a large-scale search engine which addresses many of the problems of existing systems. It makes especially heavy use of the additional structure present in hypertext to provide much higher quality search results. We chose our system name, Google, because it is a common spelling of googol, or 10^{100} and fits well with our goal of building very large-scale search

<doc>

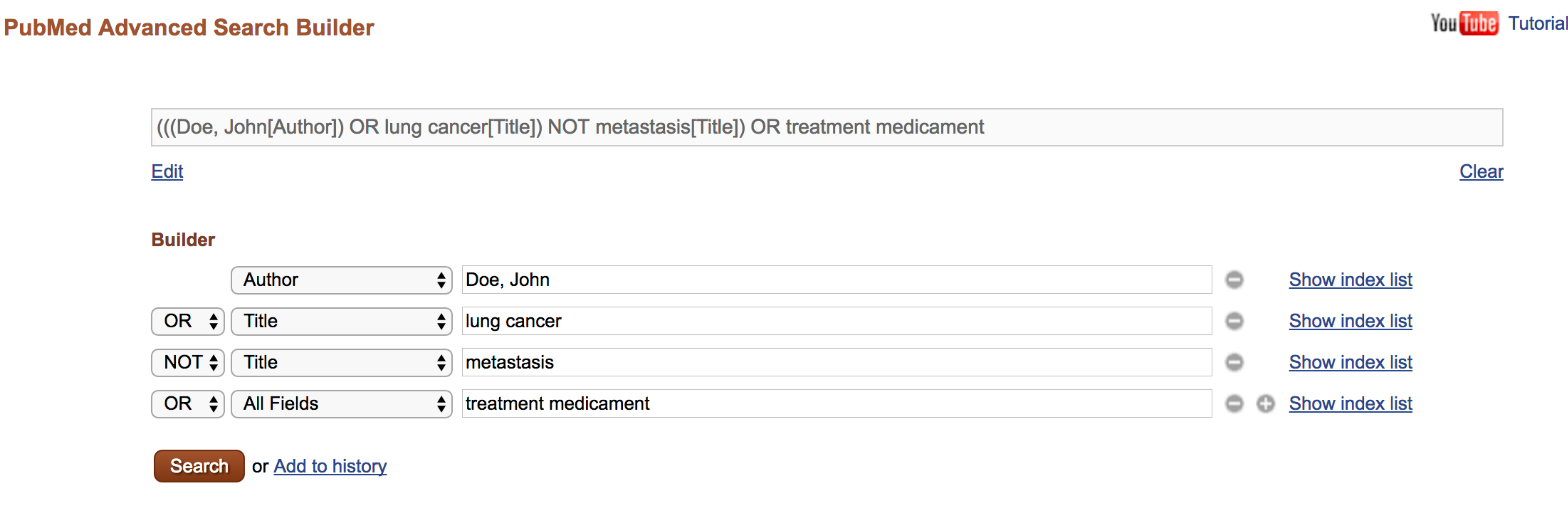
The anatomy of a large-scale Hypertextual Web Search Engine Sergey Brin Lawrence Page Computer Science Department, Stanford University, Stanford, CA, 94305, USA sergey@cs.stanford.edu and page@cs.stanford.edu Abstract In this paper, we present Google, a prototype of a large-scale search engine…

</doc>

Collection



BE AWARE THAT…



INDEXING

INDEXING

Doc 1

A document is about a bug called zyzzyva

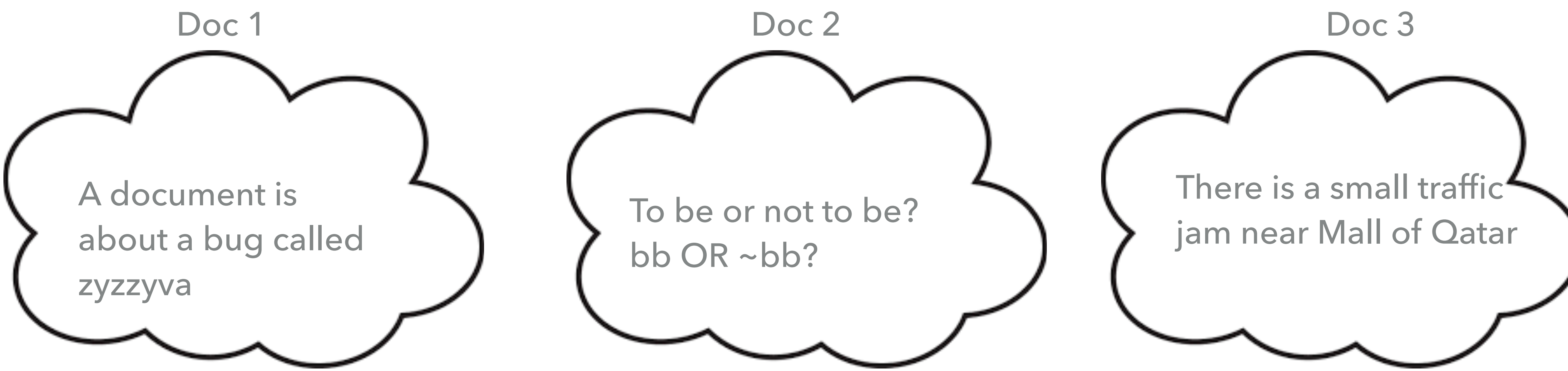
Doc 2

To be or not to be?
bb OR ~bb?

Doc 3

There is a small traffic jam near Mall of Qatar

INDEXING - BINARY FULL-TEXT REPRESENTATION



Vocabulary |V|

Collection |C|

	a	able	about	be	...	qatar	...	zyzzyva
Doc 1	1	0	1	0	...	0	...	1
Doc 2	0	0	0	1	...	0	...	0
Doc 3	1	0	0	0	...	1	...	0
...	...	...	...	...	...	...	...	...
Doc X	0	1	0	1	...	1	...	0

INDEXING - BINARY FULL-TEXT REPRESENTATION

Vocabulary |V|

Collection |C|

	a	able	about	be	...	qatar	...	zyzzyva
Doc 1	1	0	1	0	...	0	...	1
Doc 2	0	0	0	1	...	0	...	0
Doc 3	1	0	0	0	...	1	...	0
...	...	...	...	...	...	...	...	...
Doc X	0	1	0	1	...	1	...	0

INDEXING – BINARY FULL-TEXT REPRESENTATION

Vocabulary |V|

Collection |C|

	a	able	about	be	...	qatar	...	zyzzyva
Doc 1	1	0	1	0	...	0	...	1
Doc 2	0	0	0	1	...	0	...	0
Doc 3	1	0	0	0	...	1	...	0
...	...	...	...	...	...	...	...	...
Doc X	0	1	0	1	...	1	...	0

INDEXING – BINARY FULL-TEXT REPRESENTATION

a qatar

Vocabulary |V|

Collection |C|

	a	able	about	be	...	qatar	...	zyzzyva
Doc 1	1	0	1	0	...	0	...	1
Doc 2	0	0	0	1	...	0	...	0
Doc 3	1	0	0	0	...	1	...	0
...	...	...	...	...	...	...	...	...
Doc X	0	1	0	1	...	1	...	0

INDEXING - BINARY FULL-TEXT REPRESENTATION

- Invented first
- Useful simplification (almost never used - **why?**)

Vocabulary |V|

Collection |C|

	a	able	about	be	...	qatar	...	zyzzyva
Doc 1	1	0	1	0	...	0	...	1
Doc 2	0	0	0	1	...	0	...	0
Doc 3	1	0	0	0	...	1	...	0
...	...	...	...	...	...	...	...	...
Doc X	0	1	0	1	...	1	...	0

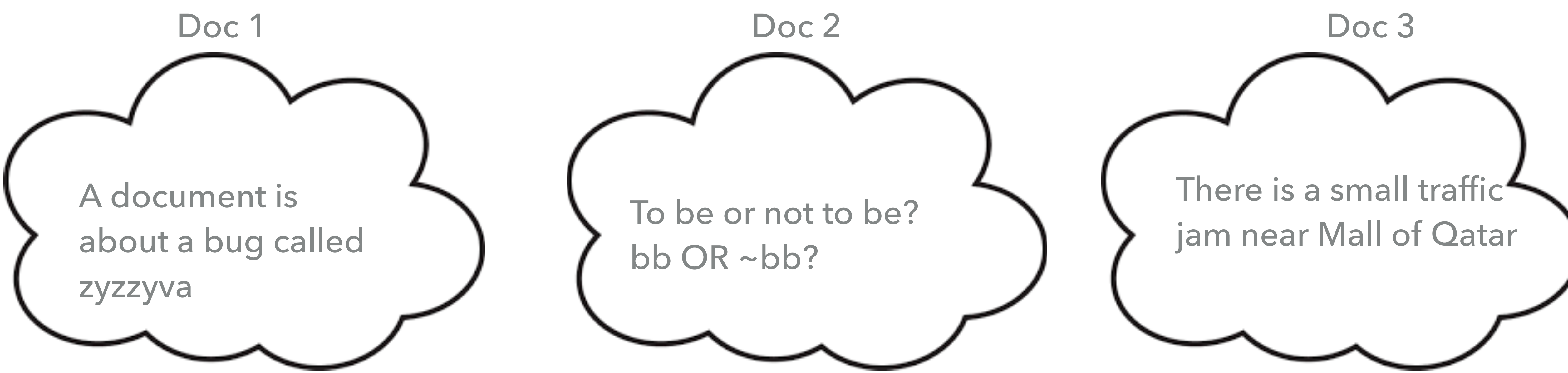
INDEXING – BINARY FULL-TEXT REPRESENTATION

- Position and frequency are ignored
- Can we do something for the frequency?

Vocabulary |V|

Collection |C|

	a	able	about	be	...	qatar	...	zyzzyva
Doc 1	1	0	1	0	...	0	...	1
Doc 2	0	0	0	1	...	0	...	0
Doc 3	1	0	0	0	...	1	...	0
...	...	...	...	...	...	...	...	...
Doc X	0	1	0	1	...	1	...	0



Vocabulary |V|

Collection |C|

	a	able	about	be	...	qatar	...	zyzzyva
Doc 1	**2**	0	1	0	...	0	...	1
Doc 2	0	0	0	**2**	...	0	...	0
Doc 3	1	0	0	0	...	1	...	0
...	...	...	...	...	...	...	...	...
Doc X	0	**5**	0	1	...	1	...	0

FREQUENCY-BASED FULL-TEXT REPRESENTATION

- More effective for search, but…
- Still very very very large table… 90% are zeros…
- Space complexity O(C*V)

Vocabulary |V|

Collection |C|

	a	able	about	be	…	qatar	…	zyzzyva
Doc 1	2	0	1	0	…	0	…	1
Doc 2	0	0	0	2	…	0	…	0
Doc 3	1	0	0	0	…	1	…	0
…	…	…	…	…	…	…	…	…
Doc X	0	5	0	1	…	1	…	0

FREQUENCY-BASED FULL-TEXT REPRESENTATION

- How to process a query: “qatar doha” ?

Vocabulary |V|

Collection |C|

	a	able	about	be	...	qatar	...	zyzzyva
Doc 1	2	0	1	0	...	0	...	1
Doc 2	0	0	0	2	...	0	...	0
Doc 3	1	0	0	0	...	1	...	0
...	...	...	...	...	...	...	...	...
Doc X	0	5	0	1	...	1	...	0

QUERY MATCHING

```
result = []
for query_token in query:
    for document in collection:
        for document_token in document:
            if query_token == document_token:
                result += [document]
                break
```

Vocabulary |V|

Collection |C|

	a	able	about	be	...	qatar	...	zyzzyva
Doc 1	2	0	1	0	...	0	...	1
Doc 2	0	0	0	2	...	0	...	0
Doc 3	1	0	0	0	...	1	...	0
...	...	...	...	...	...	...	...	...
Doc X	0	5	0	1	...	1	...	0

QUERY MATCHING

```
result = []
for query_token in query:
    for document in collection:
        for document_token in document:
            if query_token == document_token:
                result += [document]
                break
```

Most of them will not match

Time complexity: O(|q| * C * |D|)

Vocabulary |V|

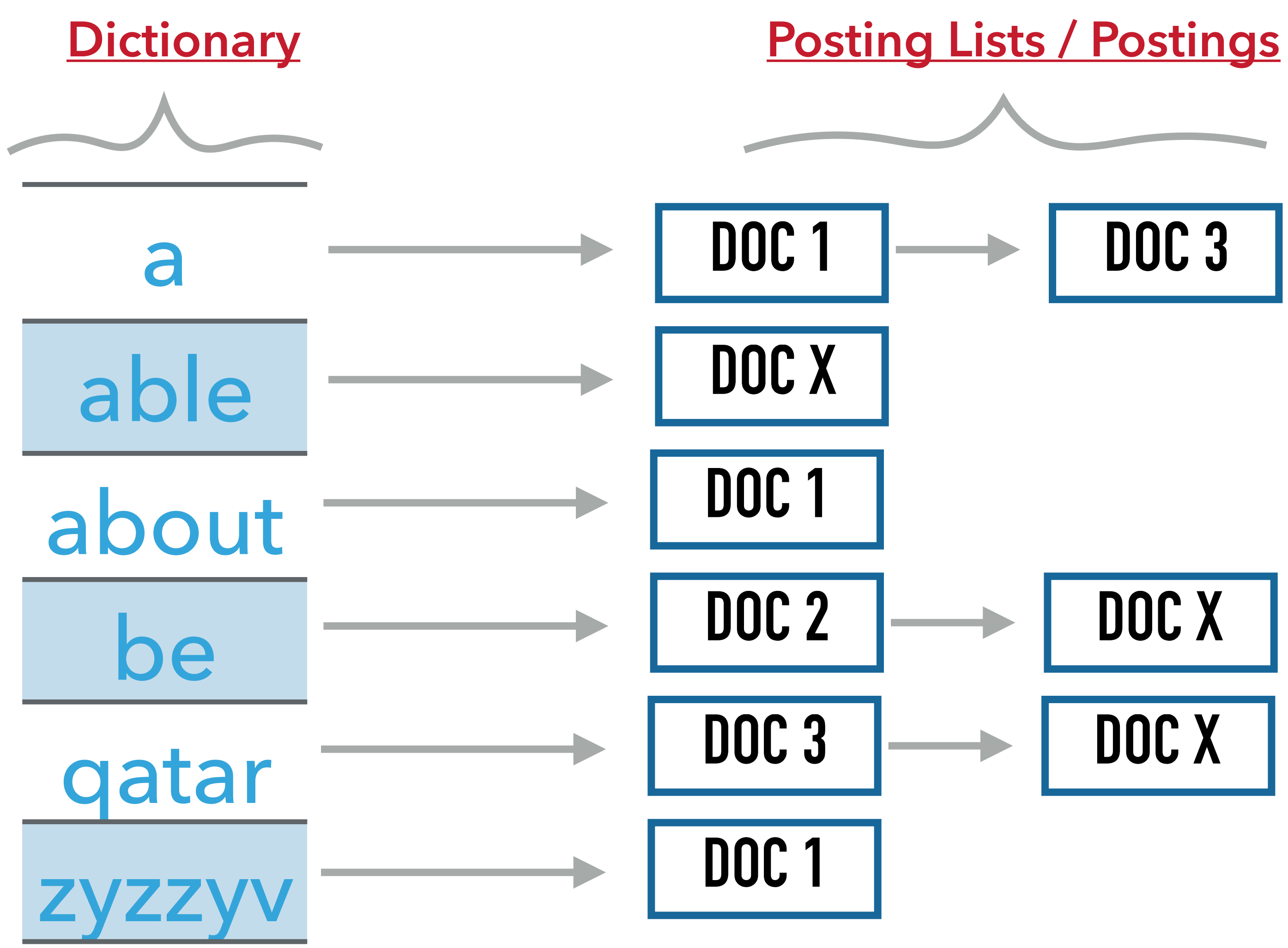
Collection |C|

	a	able	about	be	...	qatar	...	zyzzyva
Doc 1	**2**	0	1	0	...	0	...	1
Doc 2	0	0	0	**2**	...	0	...	0
Doc 3	1	0	0	0	...	1	...	0
...	...	...	...	...	...	...	...	...
Doc X	0	**5**	0	1	...	1	...	0

Inverted Index

SOLUTION: INVERTED INDEX

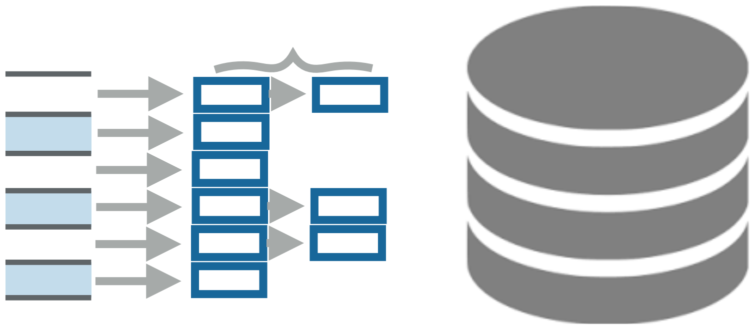
- Look-up table (hash table) for each word in vocabulary



Time complexity: O(|q| * |L|)

|L| is the average length of posting list

INVERTED INDEX STRUCTURES



- **Dictionary:** medium size
  - (largely) stays in memory in large search engines
  - Usually implemented with Hash table, B-tree, trie…
- **Postings:** very large size
  - Stays on disk **(but not for our homework)**
  - Might need compression
  - Usually a lot of statistics are kept there: docId, term freq, term position…

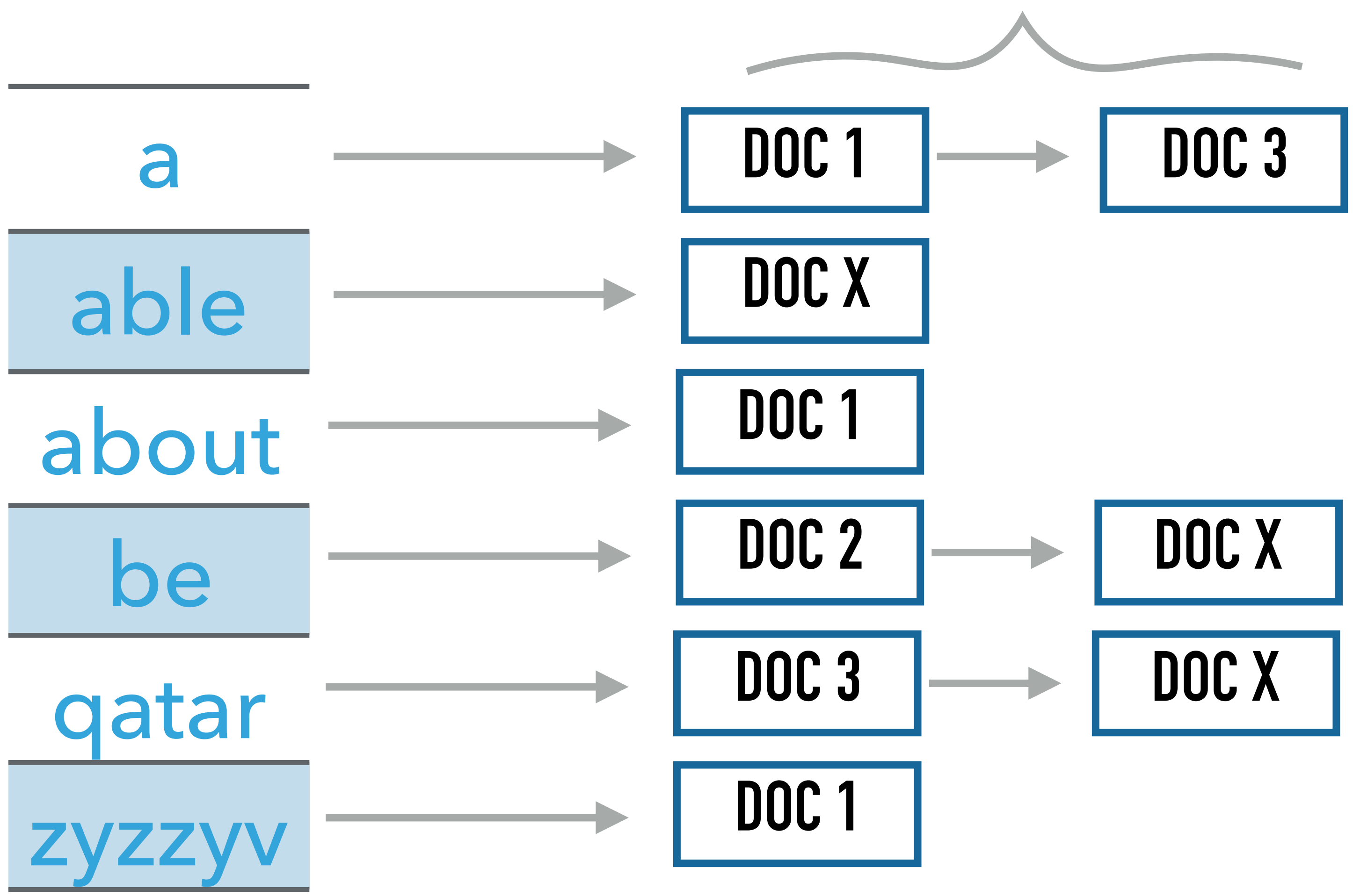
BOOLEAN SEARCH

Inverted Index

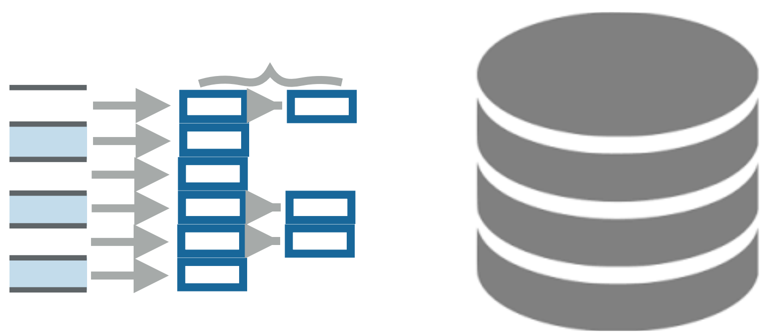
SEARCHING WITH INVERTED INDEX

a qatar

Posting Lists / Postings



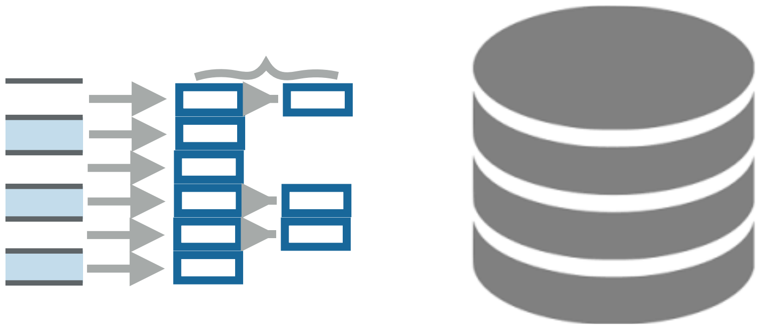
Inverted Index



SEARCHING WITH INVERTED INDEX

- Query syntax and operators:
  - doha OR qatar ; doha AND qatar ; doha NOT qatar
- Same preprocessing procedures as on indexed documents
  - lower case/upper case
  - tokenization
  - stemming
  - stopwords removal

Inverted Index

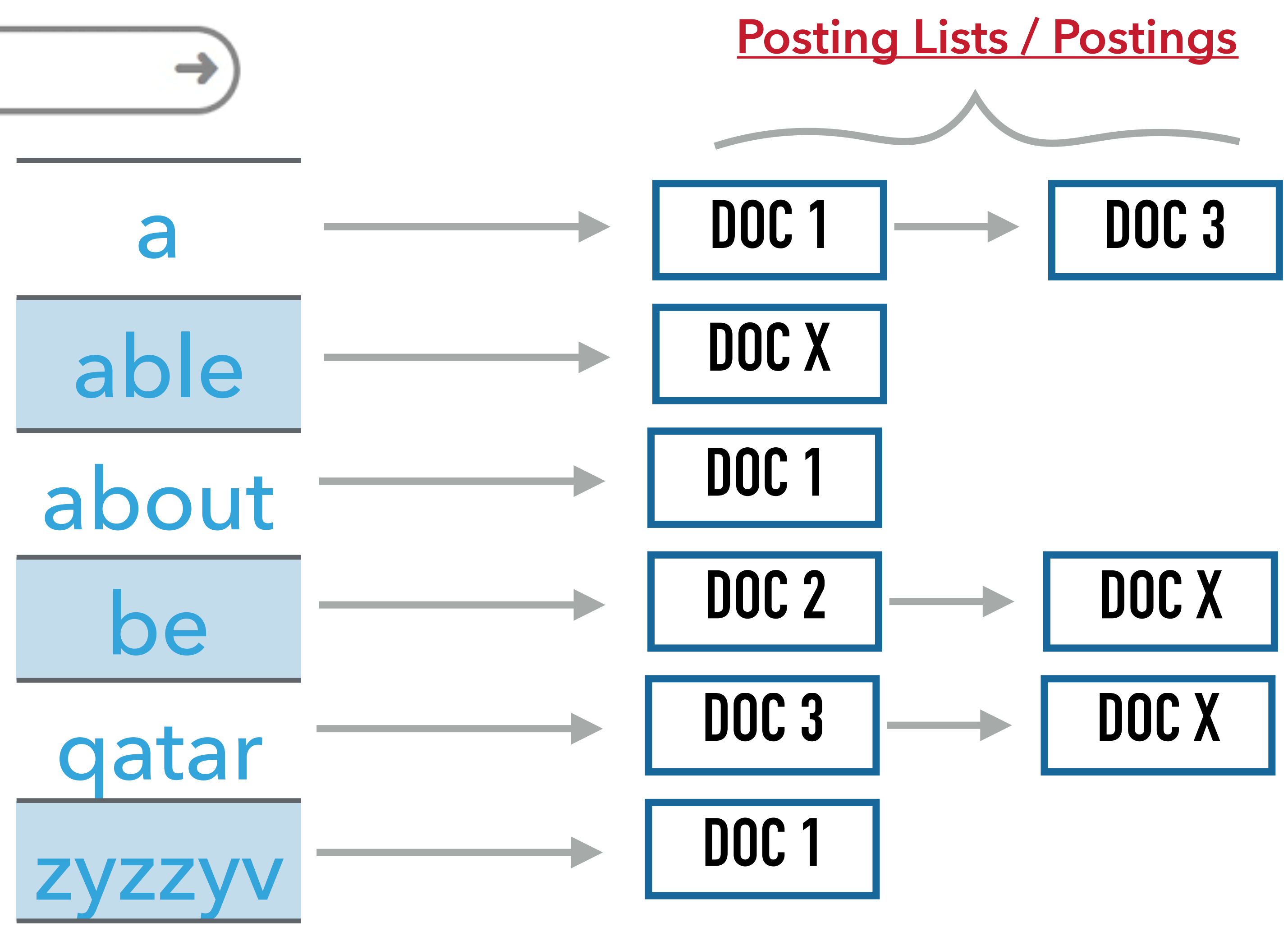


SEARCHING WITH INVERTED INDEX

- How to:
  - Look up each term in the dictionary
  - Retrieve their posting lists
  - Operations:
    - AND: intersection of posting lists
    - OR: union of posting lists
    - NOT: difference of posting lists

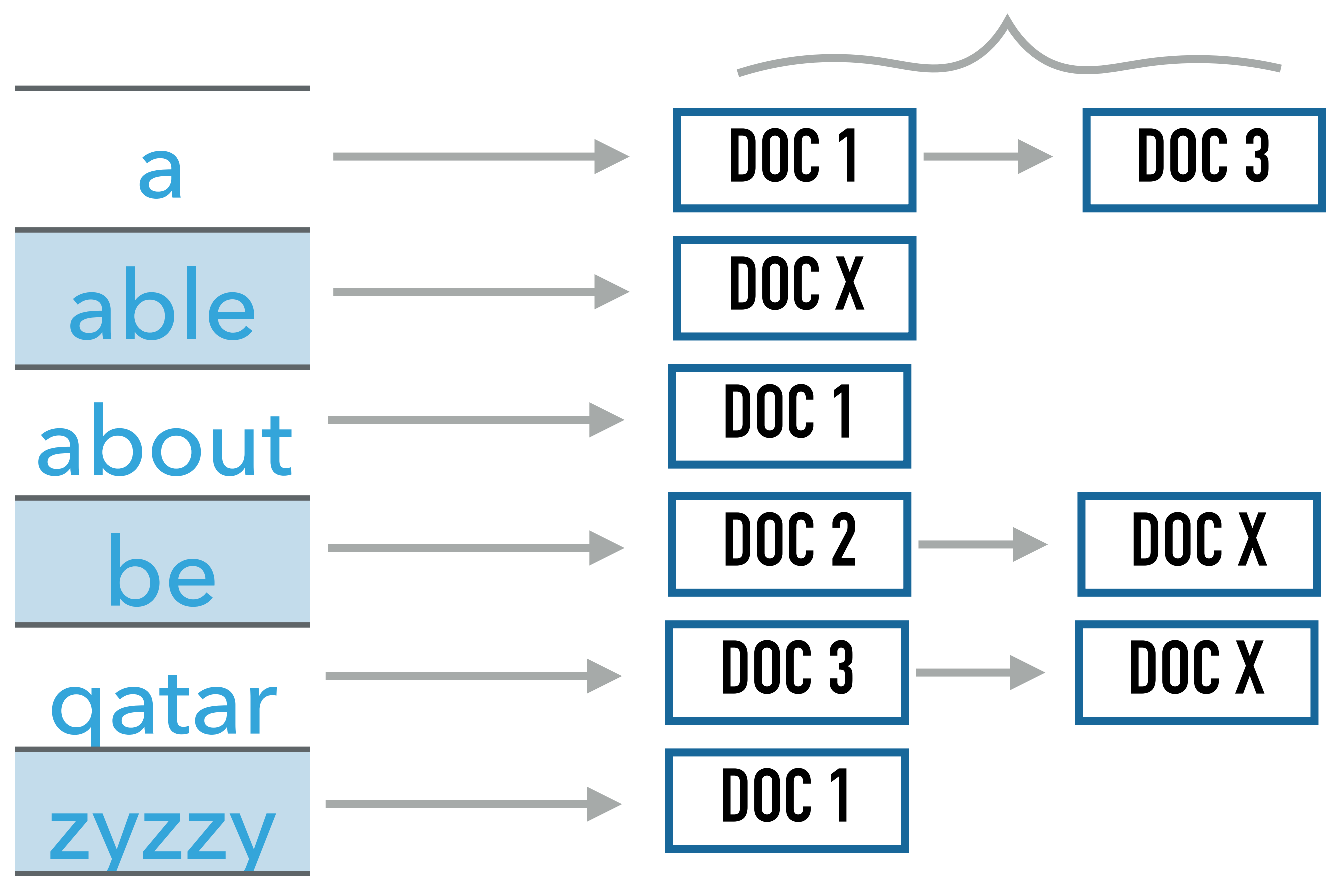
SEARCHING WITH INVERTED INDEX

- Example:
  - about OR able OR qatar
  - be AND be AND able
  - qatar AND NOT about



INCREASING COMPLEXITY

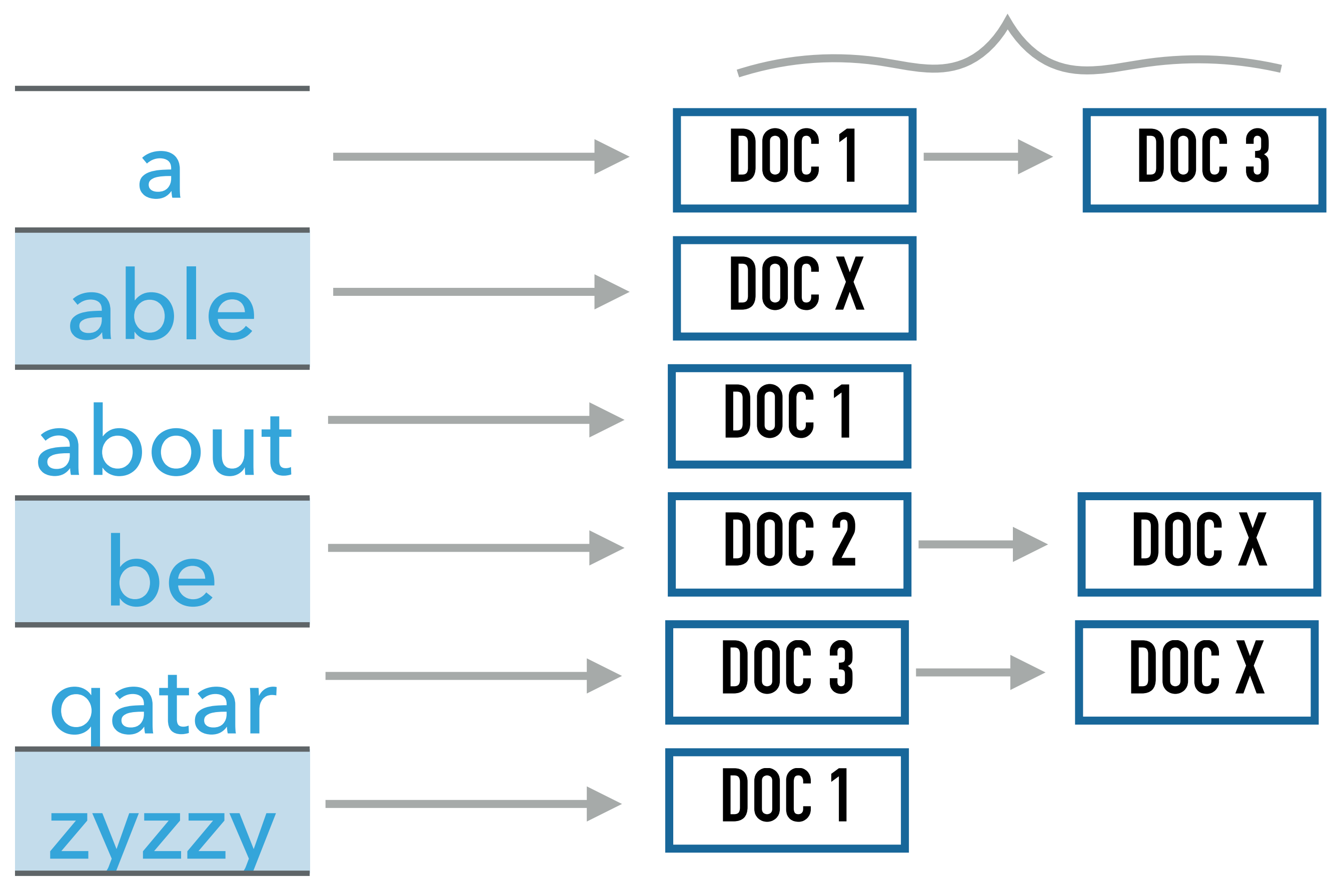
- What would be necessary to be able to deal with phrase queries? Ex. “barack obama”, “search engines”, “new york”



INCREASING COMPLEXITY

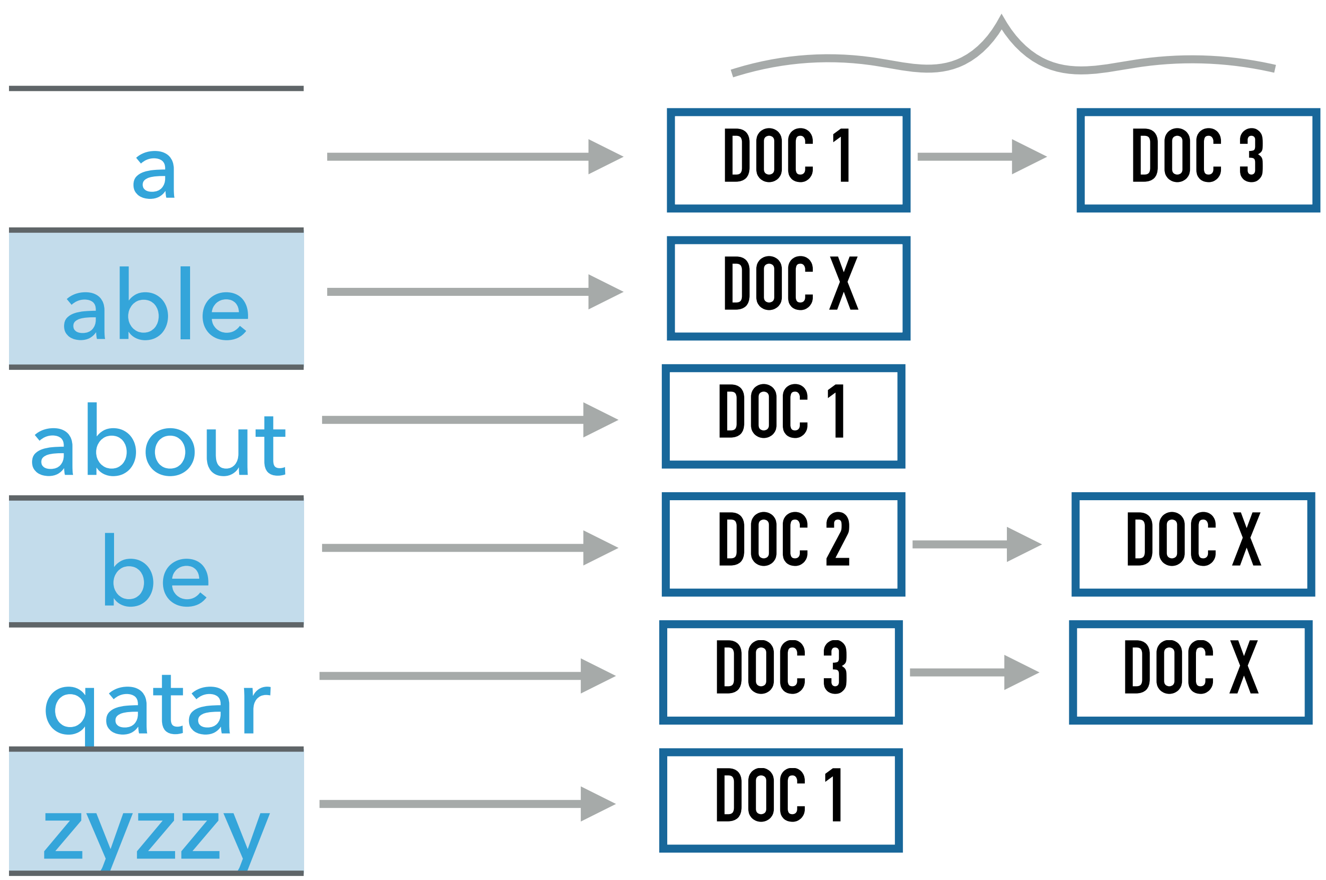
- What would be necessary to be able to deal with phrase queries? Ex. "barack obama", "search engines", "new york"

<u>Save term position in document postings</u>



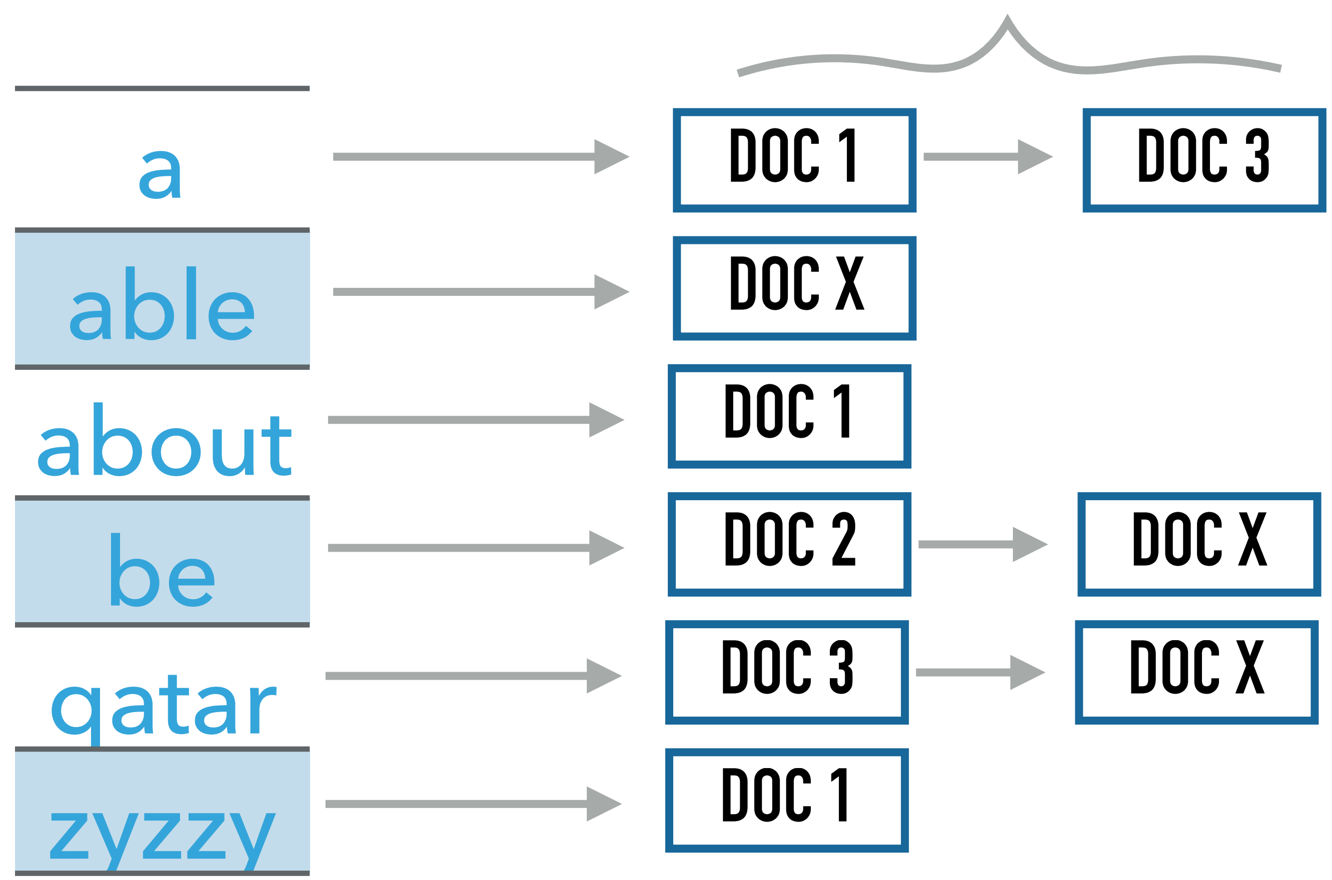
INCREASING COMPLEXITY

- What would be necessary to be able to deal with phrase queries? Ex. “barack obama”, “search engines”, “new york”
  - T2.pos - T1.pos = 1 in the same document



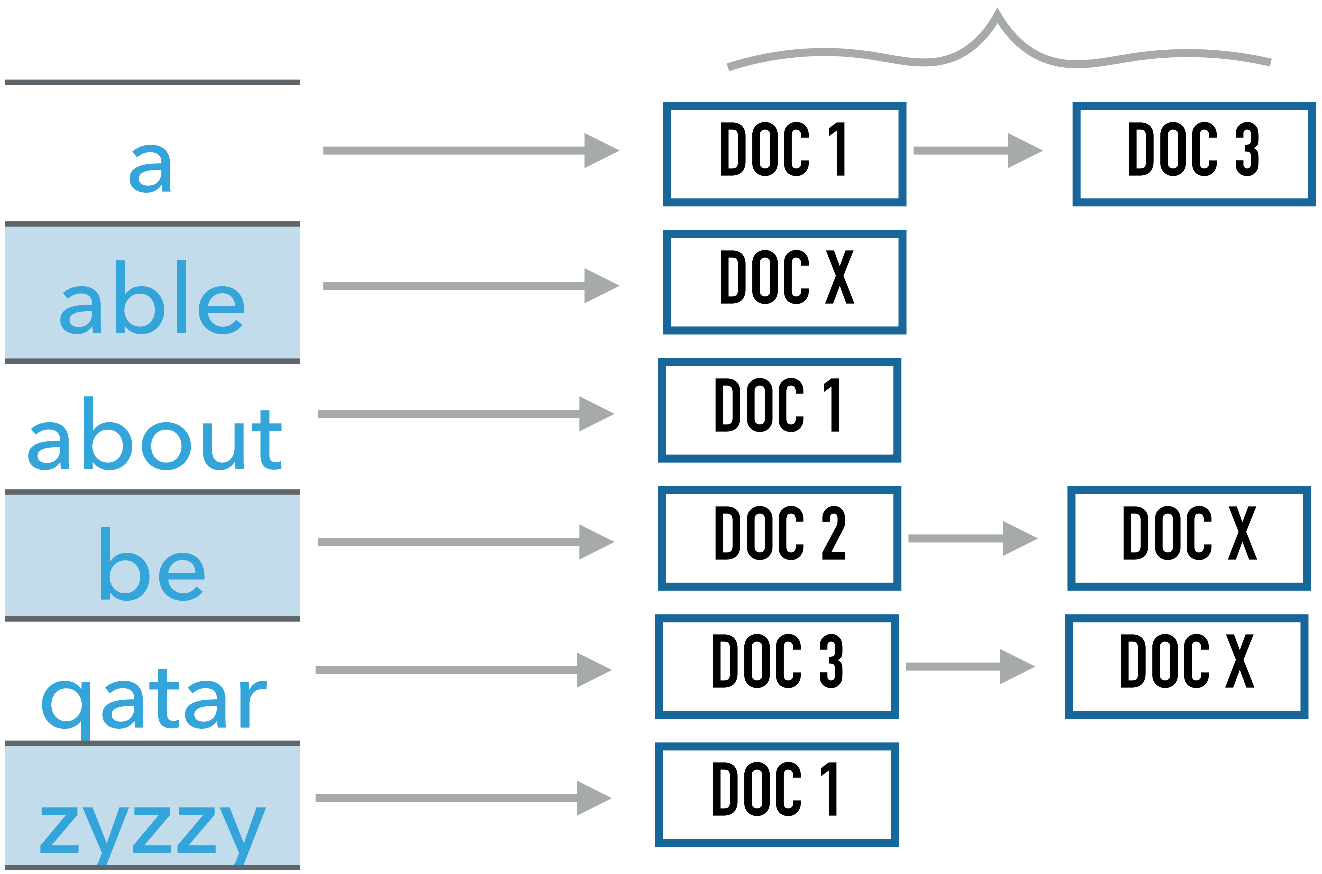
INCREASING COMPLEXITY

- What would be necessary to be able to deal with phrase queries? Ex. “barack obama”, “search engines”, “new york”
  - T2.pos - T1.pos = 1 in the same document
- How about proximity: (cancer)~5w (obama)



INCREASING COMPLEXITY

- What would be necessary to be able to deal with phrase queries? Ex. “barack obama”, “search engines”, “new york”
  - T2.pos - T1.pos = 1 in the same document
- How about proximity: (cancer)~5w (obama)
  - |T1.pos - T2.pos| < k



<u>NOT NECESSARY IN YOUR HOMEWORK ASSINGMENTS</u>

IMPLEMENTATION NOTE

- Boolean Search is quite simple to implement
  - Sets are used to do the main operation
- It is necessary to have a query language:
  - term OR term AND term NOT (term OR term)
  - Processing the query is harder now
  - Normal Formulas can help you doing that

PIECES OF CODE FOR QUERY PROCESSING IN PYTHON

```
from sympy.logic.boolalg import to_dnf

query = "(qatar | qatari) & doha & ~(arabic | desert)"
processed = to_dnf(query)

print processed
```

```
Or(And(Not(arabic), Not(desert), doha, qatar), And(Not(arabic), Not(desert), doha, qatari))
```

```
from sympy.logic import And, Or, Not
from sympy.core.symbol import Symbol

for part in processed.args:
    if type(part) is And:
        print "There is an AND:", part
    if type(part) is Or:
        print "There is a Or:", part
    if type(part) is Not:
        print "There is a Or:", part
    if type(part) is Symbol:
        print "This is a symbol :", part
```

```
There is an AND: And(Not(arabic), Not(desert), doha, qatar)
There is an AND: And(Not(arabic), Not(desert), doha, qatari)
```

BONUS PART

SPELLING

SPELLING CORRECTION

- How to find and deal with misspelled queries:
    - "barck obama" –> did you mean "barack obama"?

SPELLING CORRECTION

- How to find and deal with misspelled queries:
  - “barck obama” –> did you mean “barack obama”?
- Out of English dictionary?
- Look at the size of posting lists for “barck”?
- Look at past query logs

SPELLING CORRECTION

- Edit distance:
  - What is the minimal number of additions/modifications/deletions do I need to do from “barck obama” to “barack obama”?
  - Issue new query and see if result list increased
- Possibilities to apply many heuristics:
  - query context: flew form hamad airport -> flew from hamad airport
  - phonetic similarity: where, wear; to, two, too; …
    - Phonetic hasing - similar sounding terms hash to same value