Searching and Analyzing Data with Elasticsearch: Getting Started

INTRODUCING ELASTICSEARCH

Janani Ravi
CO-FOUNDER, LOONYCORN
www.loonycorn.com

Overview

A little search engine history and the importance of search

Basics steps involved in indexing and searching documents

The inverted index, the heart of a search engine

An introduction to Elasticsearch and its basic building blocks

Set up and install Elasticsearch on your local machine and check cluster health

What You Need in Your Toolkit

Prerequisites

Familiarity with the command line on a Mac, Linux or Windows machine

Familiarity with using RESTful APIs to perform actions

A very basic understanding of distributed computing

Install and Setup

The latest version of Elasticsearch, 5.4.0 requires Java version 8

A Mac, Linux or Windows machine on which Elasticsearch can be installed

Course Overview

Introduction to basic concepts in Elasticsearch, download and install

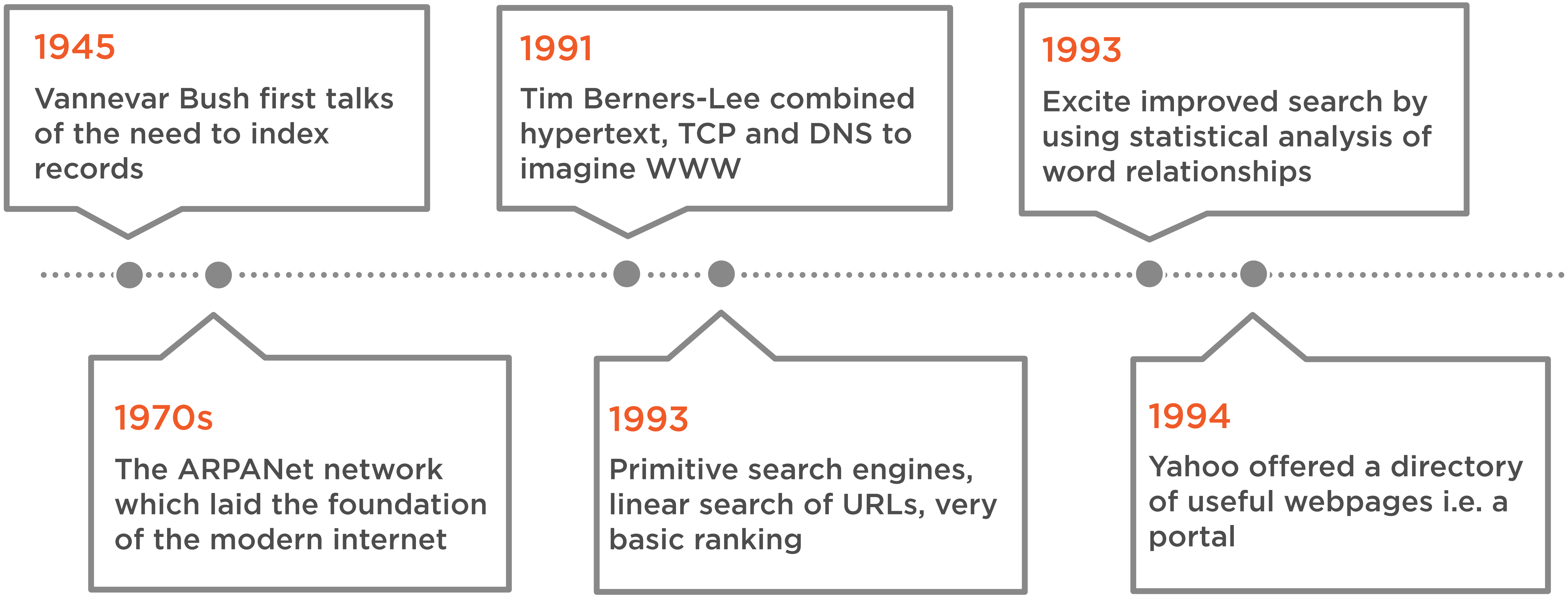
Building an index, adding documents to it both individually and in bulk

Search queries on an index using the Query DSL

Analysis of data on an index using aggregations

A Brief History of Search

Brief History of Search



1945

Vannevar Bush first talks of the need to index records

1970s

The ARPANet network which laid the foundation of the modern internet

1991

Tim Berners-Lee combined hypertext, TCP and DNS to imagine WWW

1993

Primitive search engines, linear search of URLs, very basic ranking

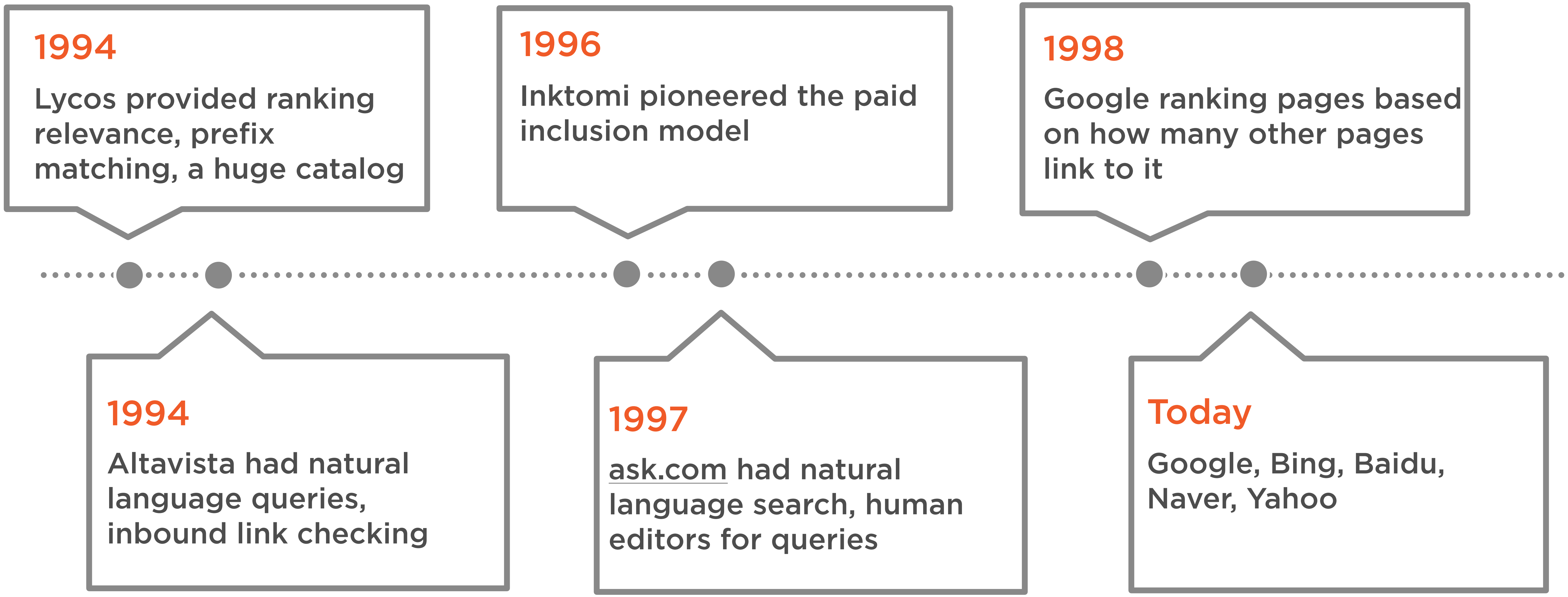
1993

Excite improved search by using statistical analysis of word relationships

1994

Yahoo offered a directory of useful webpages i.e. a portal

Brief History of Search



1994

Lycos provided ranking relevance, prefix matching, a huge catalog

1994

Altavista had natural language queries, inbound link checking

1996

Inktomi pioneered the paid inclusion model

1997

ask.com had natural language search, human editors for queries

1998

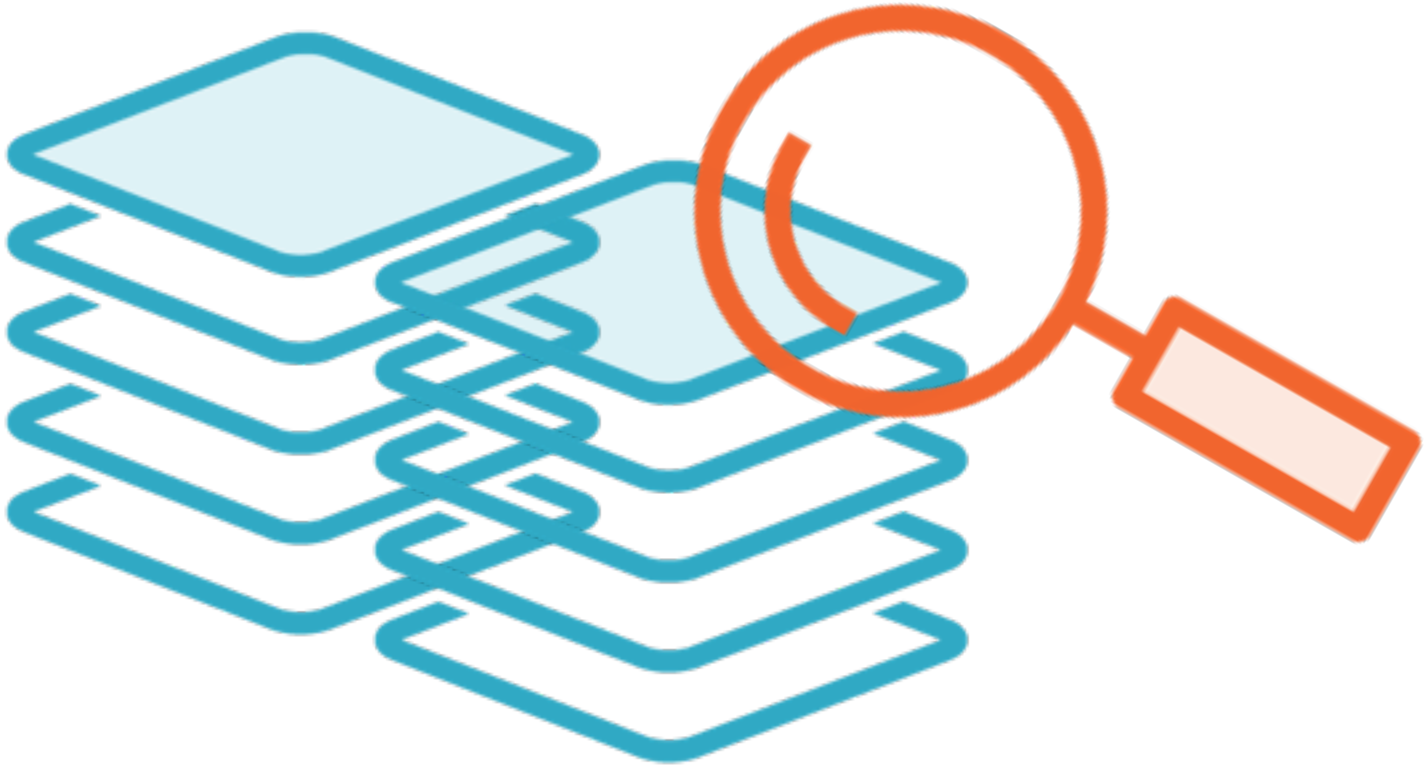
Google ranking pages based on how many other pages link to it

Today

Google, Bing, Baidu, Naver, Yahoo

How Does Search Work?

What Is the Objective of Search?



Find the most relevant documents with your search terms

Most Relevant Document for Search Terms

Know of the document's existence

Index the document for lookup

Know how relevant the document is

Retrieve ranked by relevance

Most Relevant Document for Search Terms

Web crawler

Index the document for lookup

Know how relevant the document is

Retrieve ranked by relevance

Most Relevant Document for Search Terms

Web crawler

Inverted index

Know how relevant the document is

Retrieve ranked by relevance

Most Relevant Document for Search Terms

Web crawler

Inverted index

Scoring

Retrieve ranked by relevance

Most Relevant Document for Search Terms

Web crawler

Inverted index

Scoring

Search

Most Relevant Document for Search Terms

Web crawler

Inverted index

Scoring

Search

Search Is Not Restricted to The Web

Sites Have Their Own Search

E-commerce

Video

E-learning

The Inverted Index

Documents Have Content

House Stark	House Baratheon	House Tyrell
Winter is coming	Ours is the fury	Growing Strong

Tokenize Text into Words

winter
is
coming
ours
the
fury
growing
strong

split words

lowercased

removed punctuation

Tokenize Text into Words

winter	1
is	2
coming	1
ours	1
the	1
fury	1
growing	1
strong	1

Tokenize Text into Words

winter	1
is	2
coming	1
ours	1
the	1
fury	1
growing	1
strong	1

Tokenize Text into Words

winter	1	Stark
is	2	Stark, Baratheon
coming	1	Stark
ours	1	Baratheon
the	1	Baratheon
fury	1	Baratheon
growing	1	Tyrell
strong	1	Tyrell

Tokenize Text into Words

winter	1	Stark
is	2	Stark, Baratheon
coming	1	Stark
ours	1	Baratheon
the	1	Baratheon
fury	1	Baratheon
growing	1	Tyrell
strong	1	Tyrell

Dictionary

sorted so lookup is easy

coming	1	Stark
fury	1	Baratheon
growing	1	Tyrell
is	2	Stark, Baratheon
ours	1	Baratheon
strong	1	Tyrell
the	1	Baratheon
winter	1	Stark

Postings

coming	1	Stark
fury	1	Baratheon
growing	1	Tyrell
is	2	Stark, Baratheon
ours	1	Baratheon
strong	1	Tyrell
the	1	Baratheon
winter	1	Stark

Search

coming	1	Stark
fury	1	Baratheon
growing	1	Tyrell
is	2	Stark, Baratheon
ours	1	Baratheon
strong	1	Tyrell
the	1	Baratheon
winter	1	Stark

winter

Search

coming	1	Stark
fury	1	Baratheon
growing	1	Tyrell
is	2	Stark, Baratheon
ours	1	Baratheon
strong	1	Tyrell
the	1	Baratheon
winter	1	Stark

fury

Search

coming	1	Stark
fury	1	Baratheon
growing	1	Tyrell
is	2	Stark, Baratheon
ours	1	Baratheon
strong	1	Tyrell
the	1	Baratheon
winter	1	Stark

is

Search

coming	1	Stark
fury	1	Baratheon
growing	1	Tyrell
is	2	Stark, Baratheon
ours	1	Baratheon
strong	1	Tyrell
the	1	Baratheon
winter	1	Stark

coming OR strong

Search

coming	1	Stark
fury	1	Baratheon
growing	1	Tyrell
is	2	Stark, Baratheon
ours	1	Baratheon
strong	1	Tyrell
the	1	Baratheon
winter	1	Stark

fury AND growing

Searches Using Inverted Indices

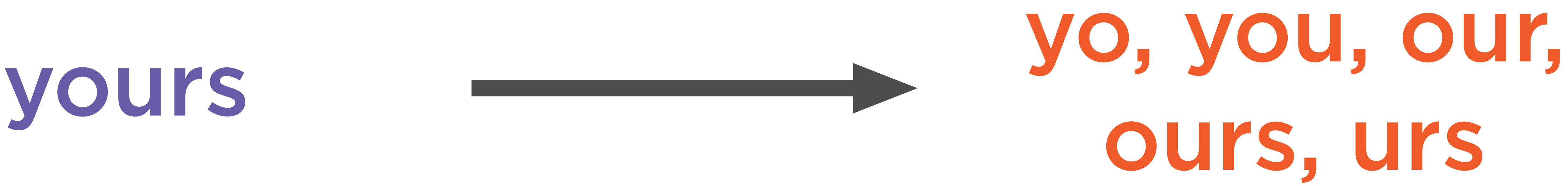
Find all words ending with “ong”

strong → gnorts

Search for all words starting with “gno”

Searches Using Inverted Indices

Split words into n-grams for substring search



yours

yo, you, our, ours, urs

Match substrings with n-grams

Searches Using Inverted Indices

Geo-hashes for geographical search

Algorithms such as Metaphone for phonetic matching

“Did you mean?” searches use a Levenshtein automaton

An inverted index is at the heart of a search engine

Implementing Search

Apache Lucene

The indexing and search library for a high performance, full-text search engine

Apache Lucene

Open source, free to use

written in Java, ported to other languages

Apache Lucene

Just like Hadoop in the distributed computing world, Lucene is the nucleus of several technologies built around it

Apache Lucene

Solr

A **search server** with: distributed indexing, load balancing, replication, automated recover, centralized configuration

Apache Lucene

Nutch

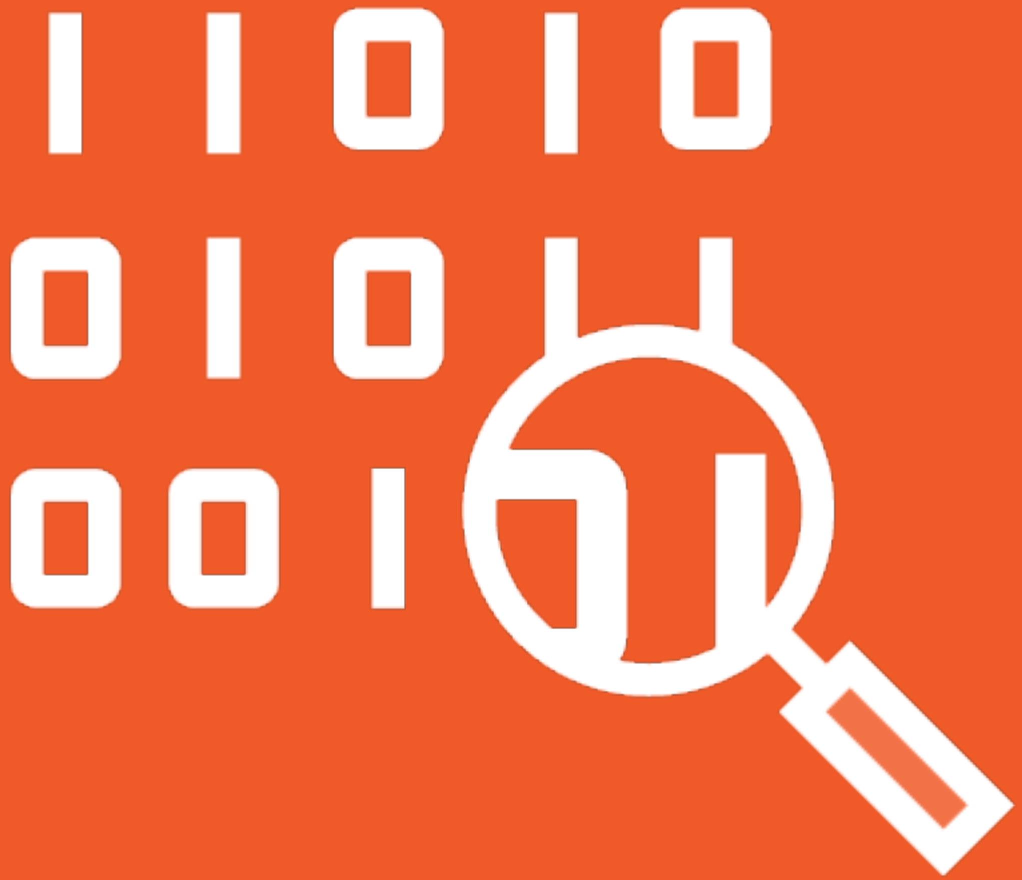
Web crawling and index parsing

Apache Lucene

CrateDB

Open source, SQL distributed database

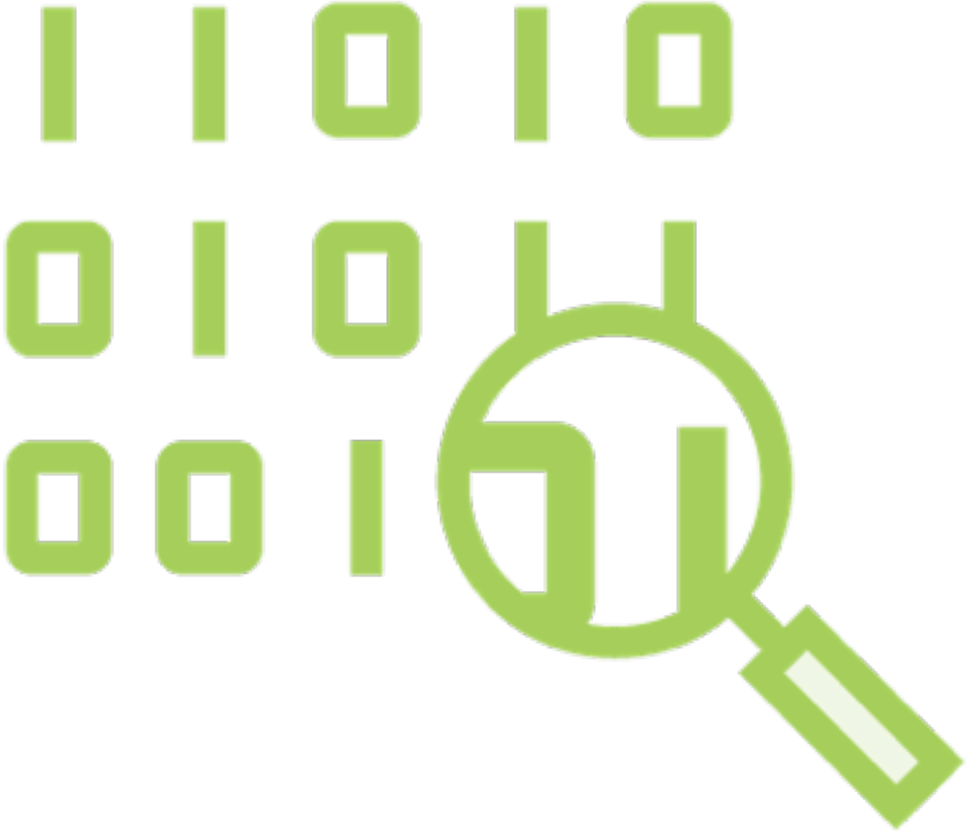
Elasticsearch



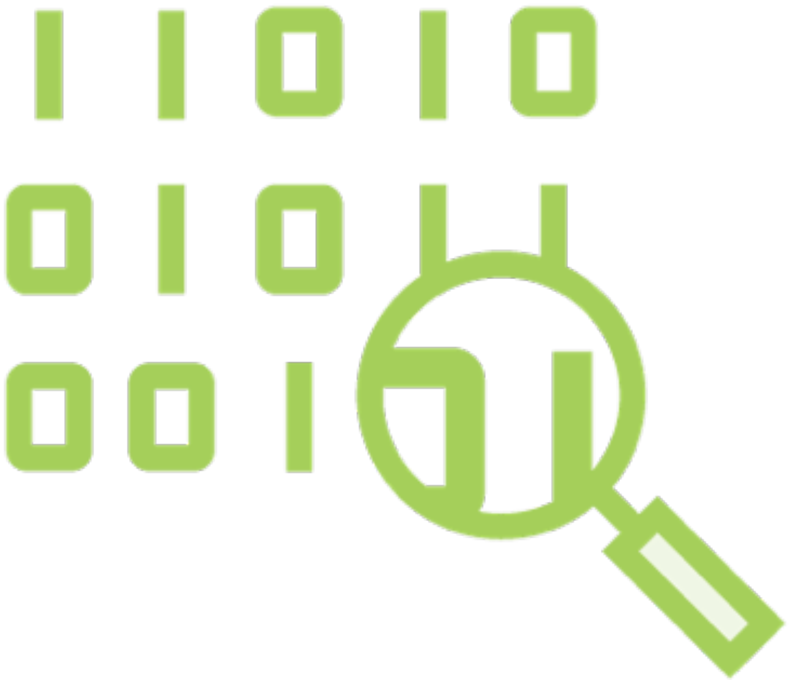
Elasticsearch is a distributed search and analytics engine which runs on Lucene

Introducing Elasticsearch

Elasticsearch



An open source, search and analytics engine, written in Java built on Apache Lucene



Elasticsearch

Distributed: Scales to thousands of nodes

High availability: Multiple copies of data

RESTful API: CRUD, monitoring and other operation via simple JSON-based HTTP calls

Powerful Query DSL: Express complex queries simply

Schemaless: Index data without an explicit schema

Elasticsearch

Product catalog

Inventory

Autocomplete

Video clips

Categories

Tags

Courses

Authors

Topics

Elasticsearch

Mining log data for insights

Price alerting platform

Business analytics and intelligence

Working with Elasticsearch

Working with Elasticsearch

As a service in the cloud

On your local machine

Working with Elasticsearch

As a service in the cloud

On your local machine

Elasticsearch

https://www.elastic.co/cloud/as-a-service

This however is subscription based, you can try it out free for 14 days.

This slide to be removed

Working with Elasticsearch

As a service in the cloud

On your local machine

Demo

Download and install Elasticsearch on your local machine

Basic Concepts of Elasticsearch

Near Realtime Search

Very low latency, ~1 second from the time a document is indexed until it becomes searchable

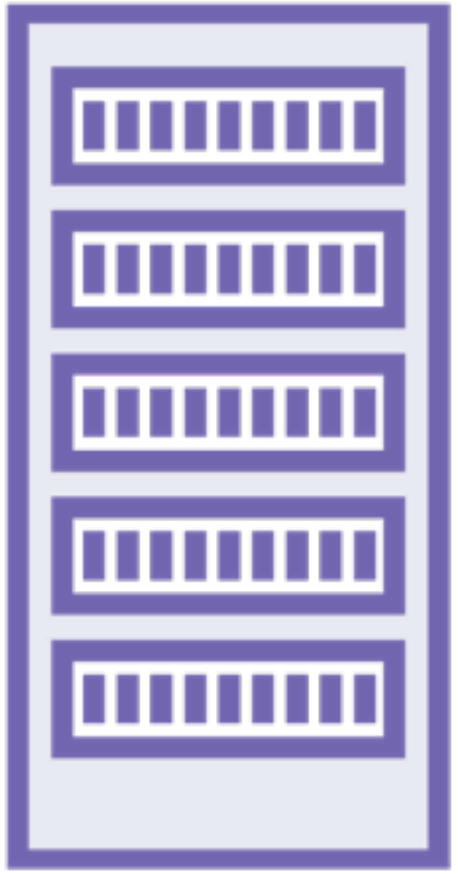
Single server

Performs indexing

Allows search

Has a unique id
and name

Node



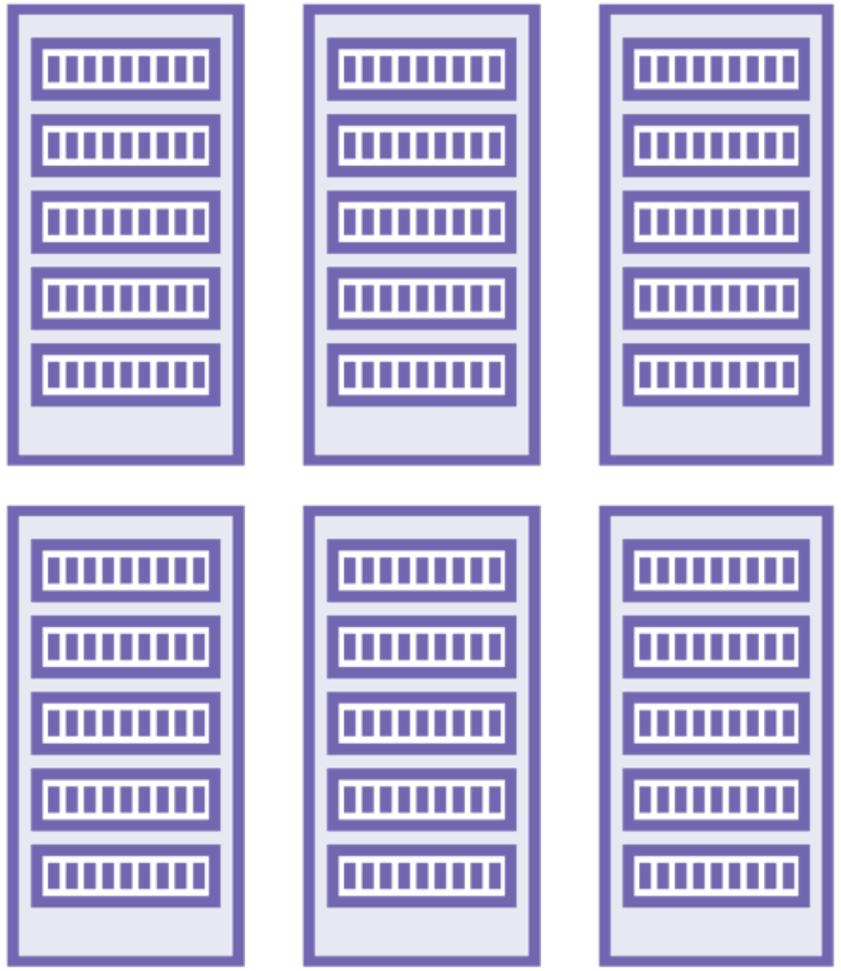
Cluster

Collection of nodes

Holds the entire indexed data

Has a unique name

Nodes join a cluster using the cluster name



Document

A whole bunch of documents that need to be indexed so they can be searched

Document

catalog, reviews

Document

titles, description, comments

Types

Documents are divided into categories or types

Index

All of these types of documents make up an **index**

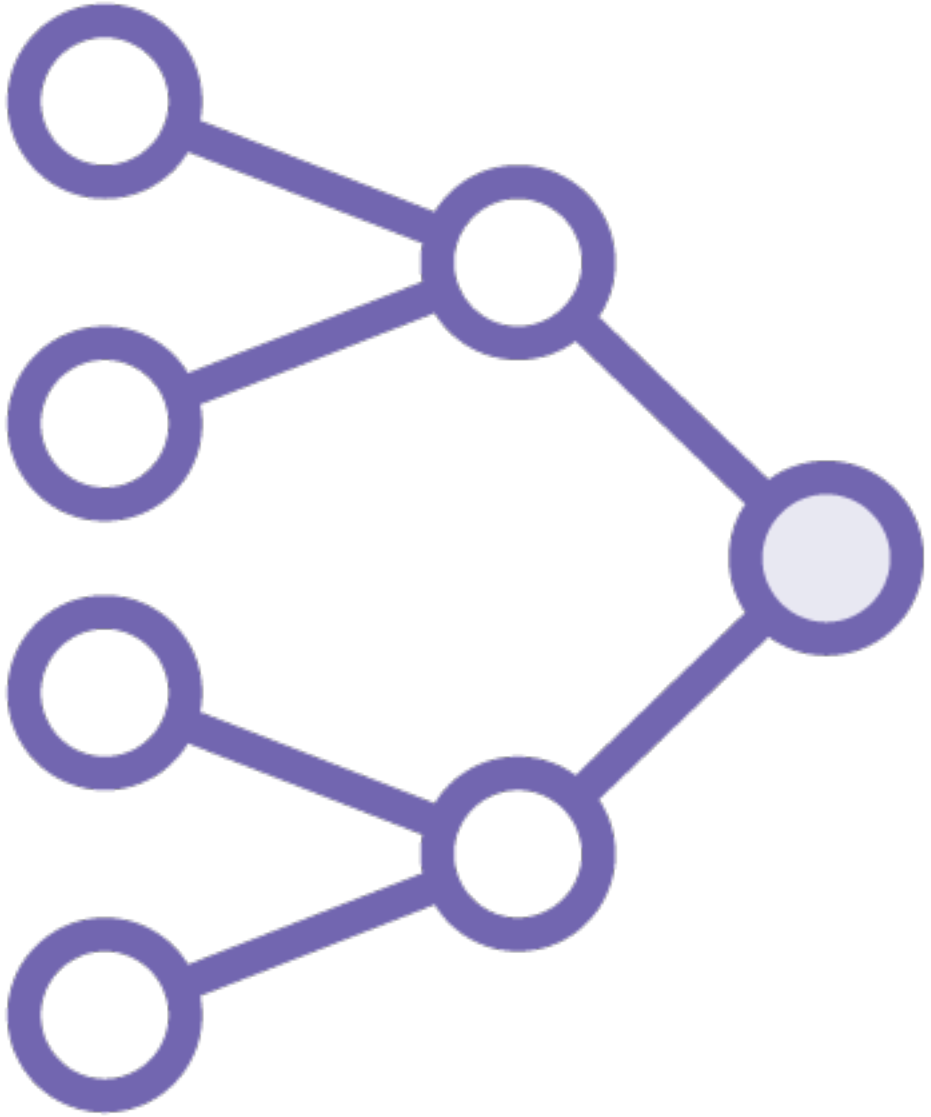
Collection of similar documents

Identified by name

Any number of indices in a cluster

Different indices for different logical groupings

Index

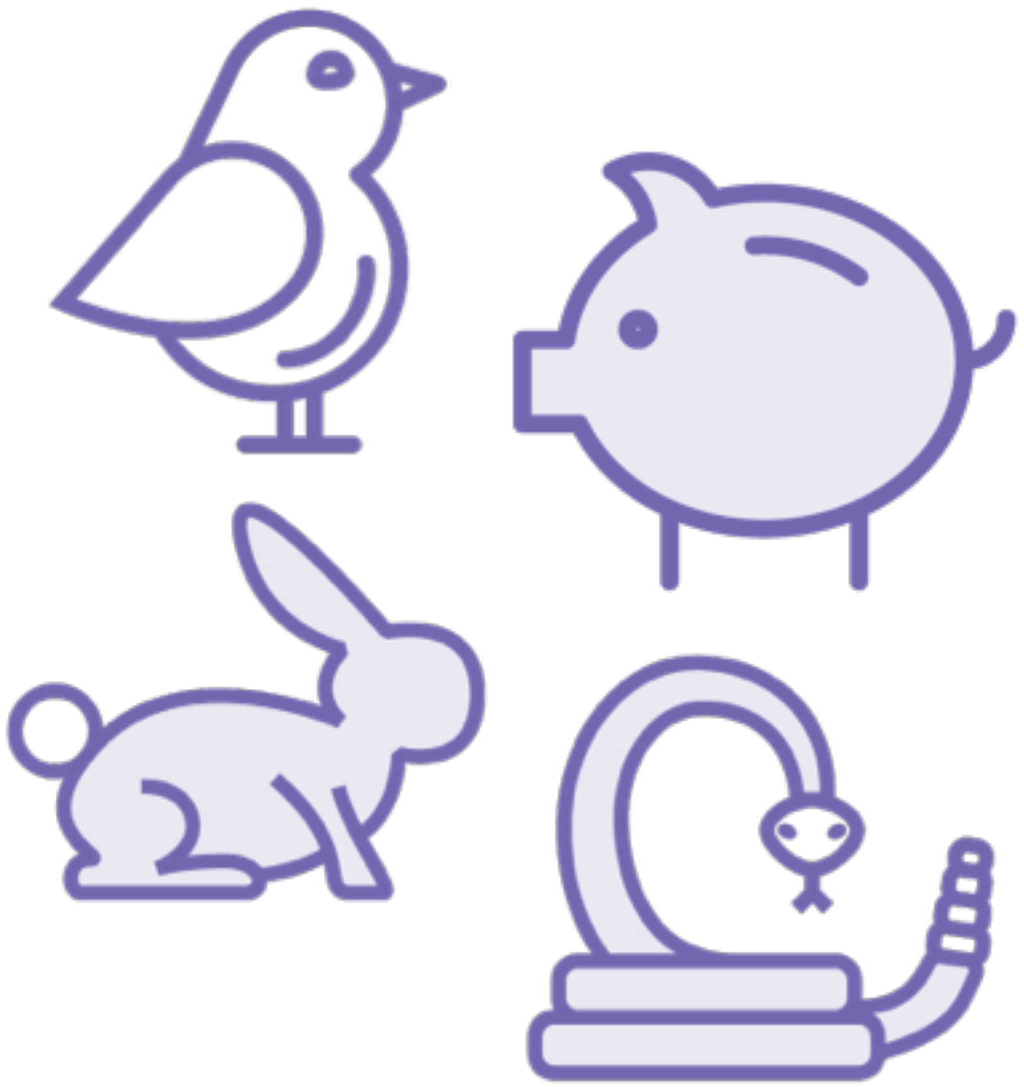


Type

Logical partitioning of documents

User defined grouping semantics

Documents with the same fields belong to one type



Document

Basic unit of information to be indexed

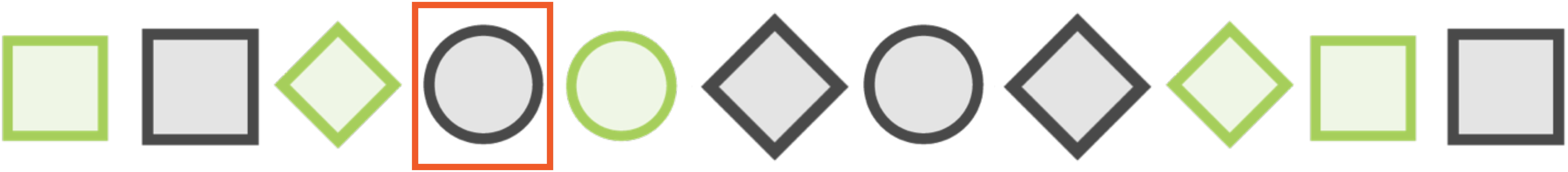
Expressed in JSON

Resides within an index

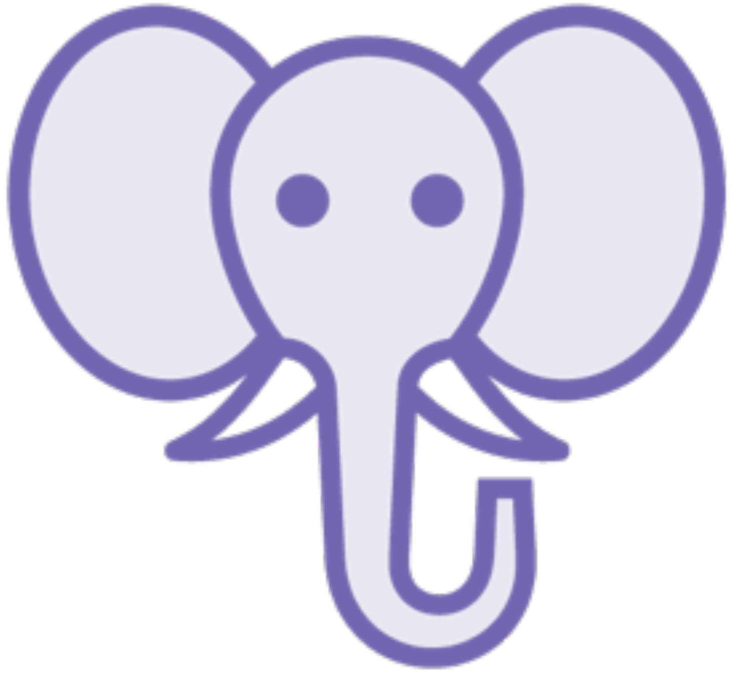
Assigned to a type within an index

Documents in an Index

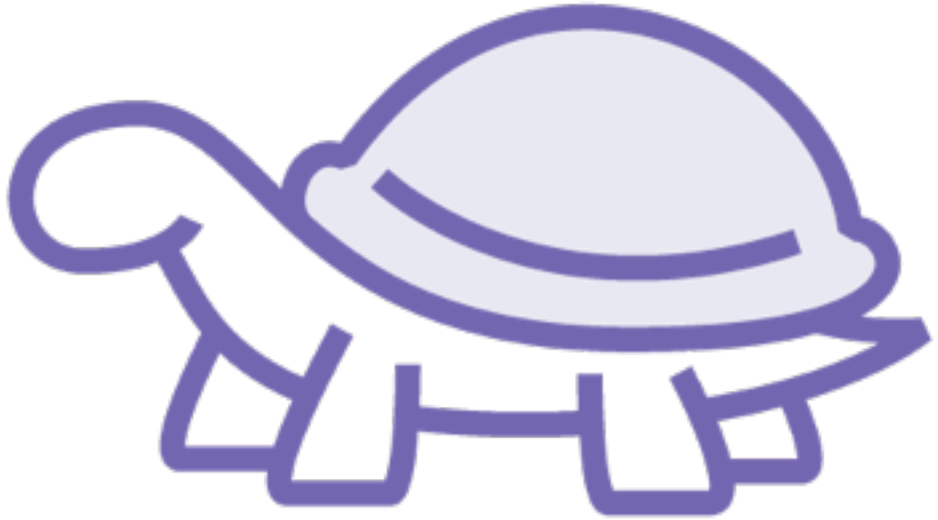
Documents in an Index



Documents in an Index

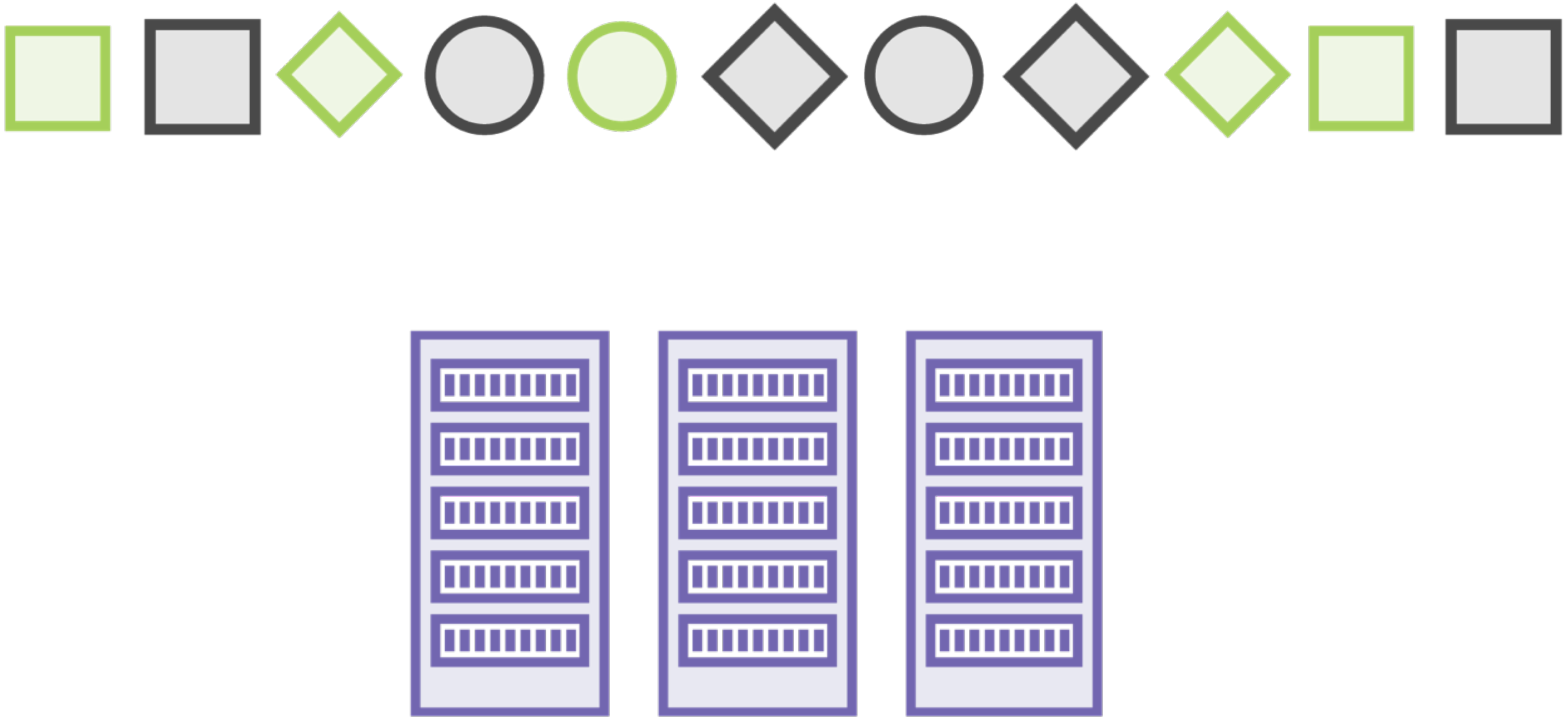


Too large to fit in the hard disk of one node

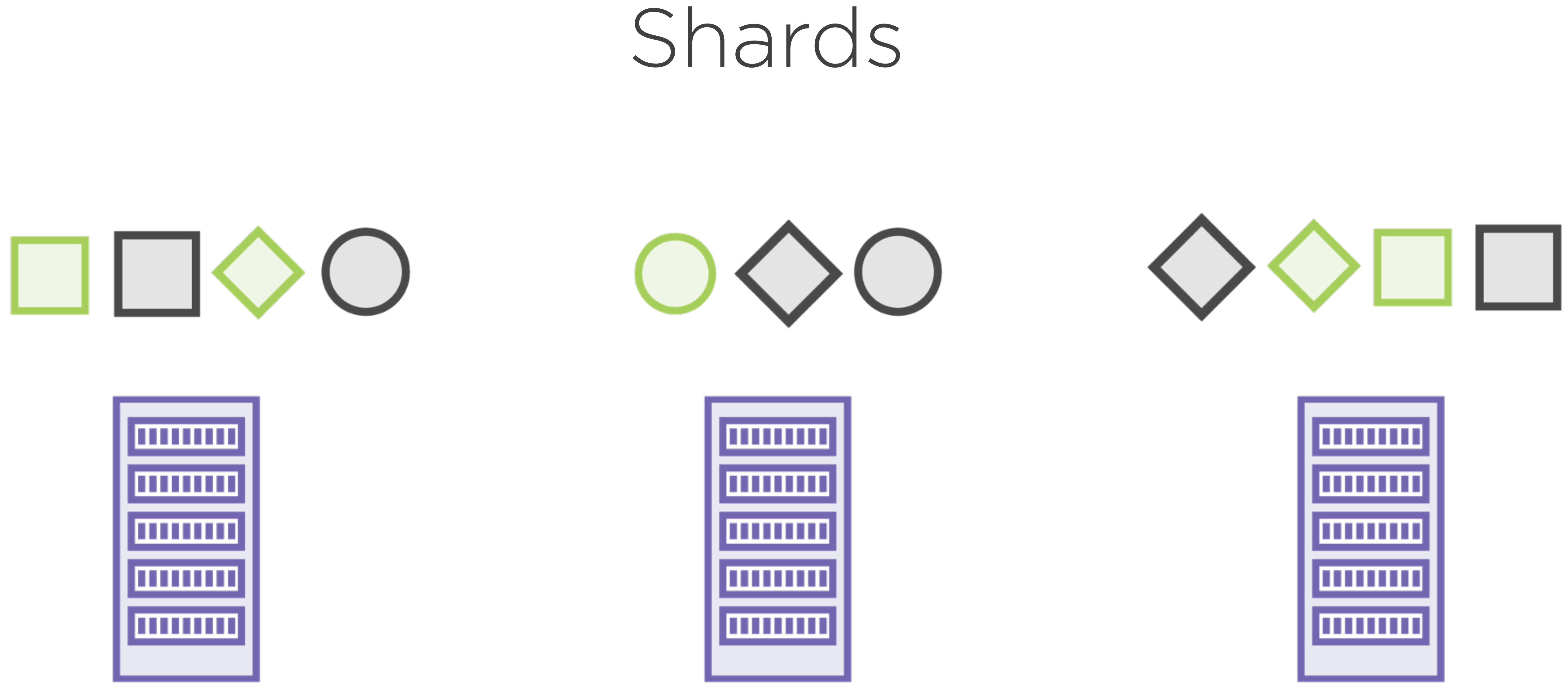


Too slow to serve all search requests from one node

Shards

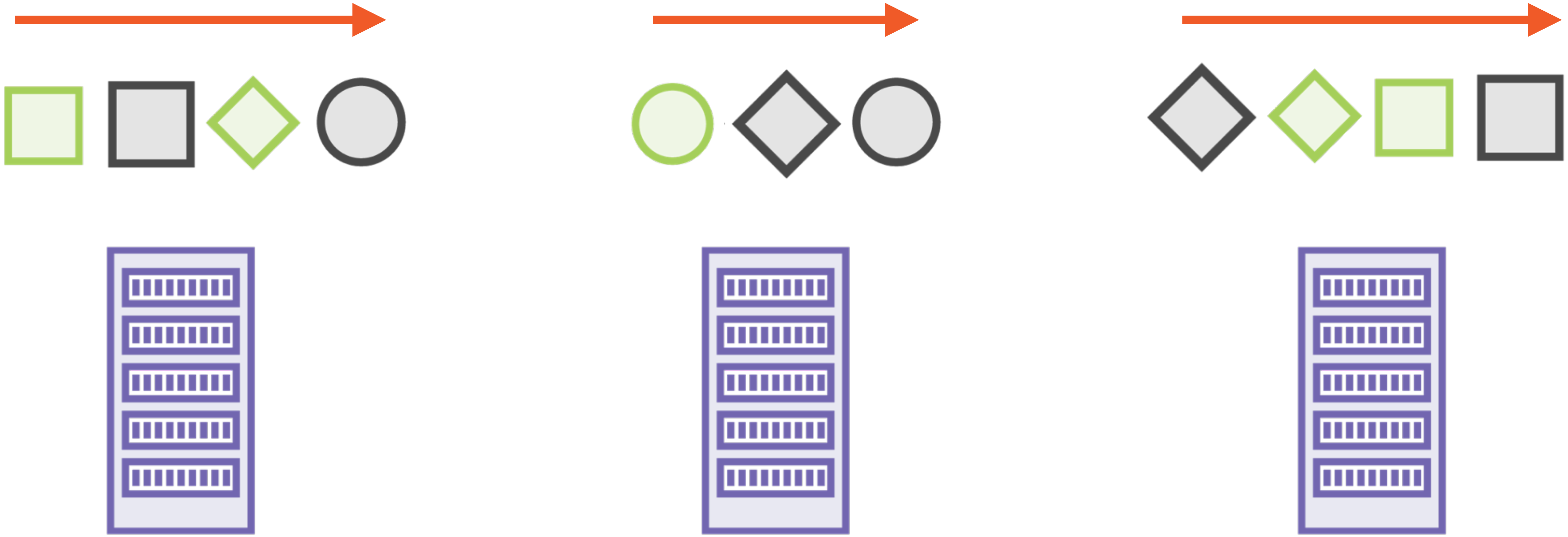


Split the index across multiple nodes in the cluster



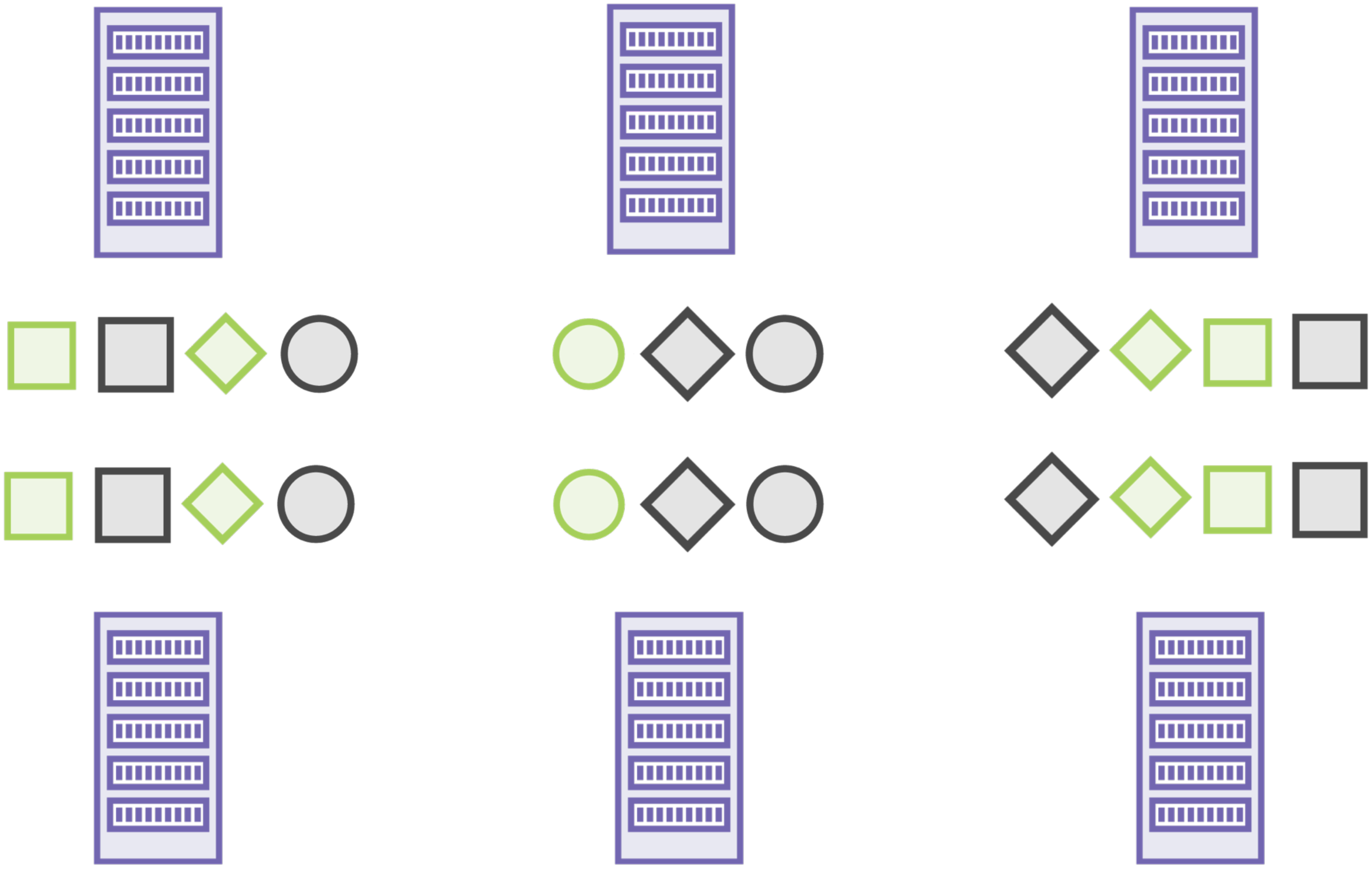
Sharding an index

Shards

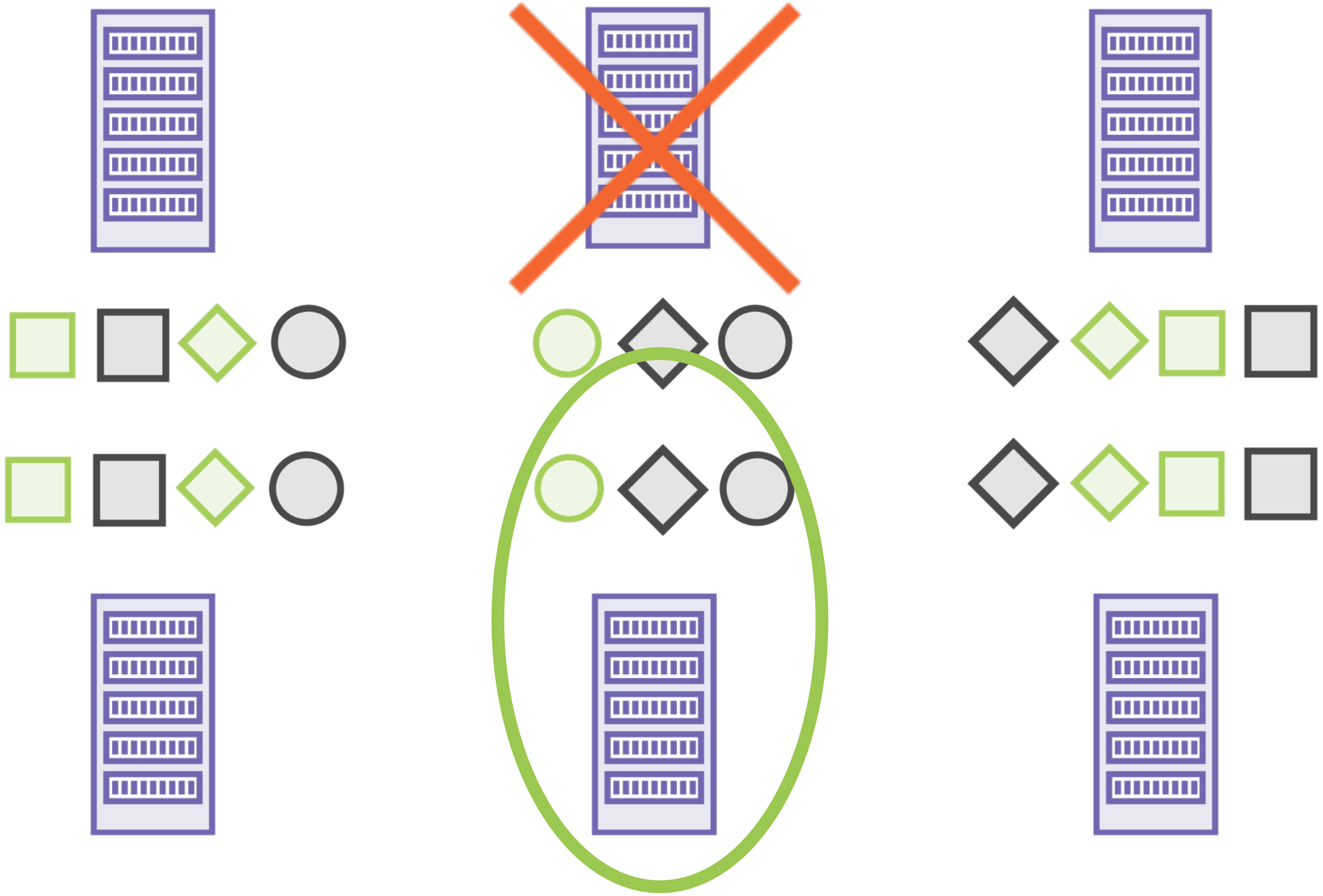


Search in parallel on multiple nodes

Replicas

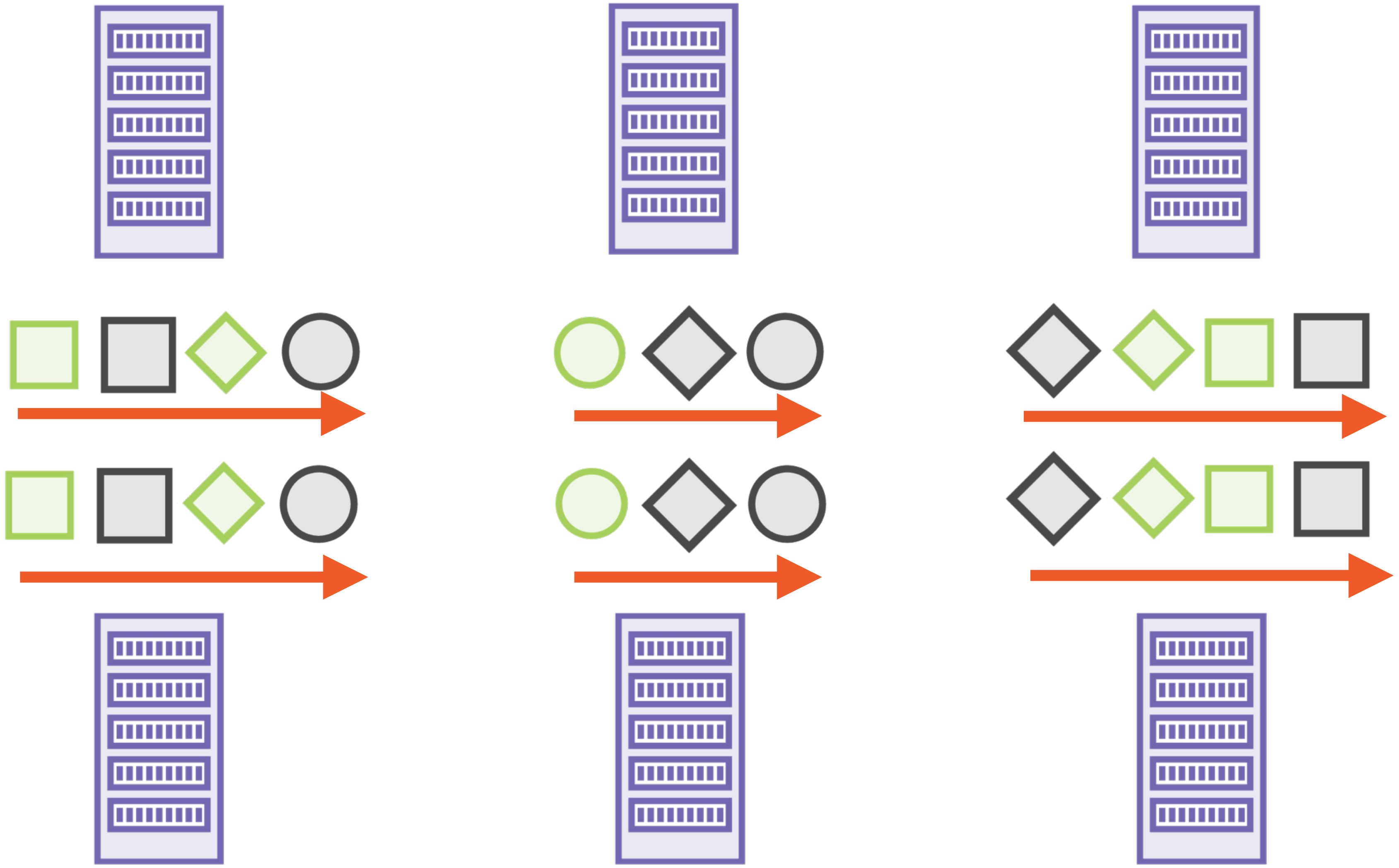


Replicas



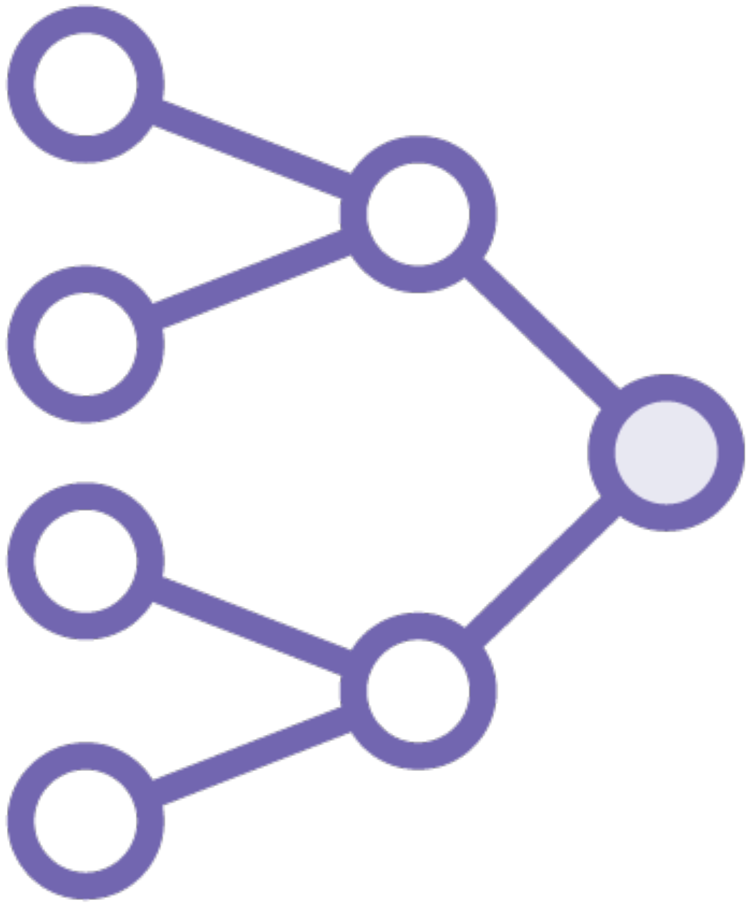
High availability in case a node fails

Replicas



Scale search volume/throughput by searching multiple replicas

Shards and Replicas



An index can be split into multiple shards

A shard can be replicated zero or more times

An index in Elasticsearch has 5 shards and 1 replica by default

Demo

Monitor the health of your cluster using HTTP requests

Summary

Learnt a little search engine history, ubiquitous nature of search

Understood the basics steps involved in indexing and searching documents

Learnt how the inverted index data structure works

Got a brief introduction to Elasticsearch and its building blocks

Set up and installed Elasticsearch on your local machine