ELASTICSEARCH

SEARCH AND ANALYTICS ENGINE

WHAT IS ELASTICSEARCH?

Open-source search engine focused on big data

Horizontally scalable

REST API web-interface with JSON output

Built on Apache Lucene - data storage and retrieval engine

AGENDA

- What is ElasticSearch?
- ElasticSearch Clients
- ELK Stack
- How does ElasticSearch works?
- Architecture and principles
- How nodes work
- Handling searches
- Inverted Indexing
- Limitations, Scalability and optimization
- Back to the ELK stack - Kibana
- Closing thoughts - Beats & Logstash

ELASTICSEARCH CLIENTS

ELK STACK

- ElasticSearch
- LogStash
- Kibana

HOW DOES ELASTICSEARCH WORKS?

BASIC CONCEPTS

Cluster

A collection of **nodes** - holds data and provides joined indexing and search capabilities.

Node

Server instance of elasticsearch. A server created when an elasticsearch instance begins.

Index

Internal Elasticsearch structure used to store and locate **documents**. A collection of documents with similar characteristics. e.g., videos or channels.

Document

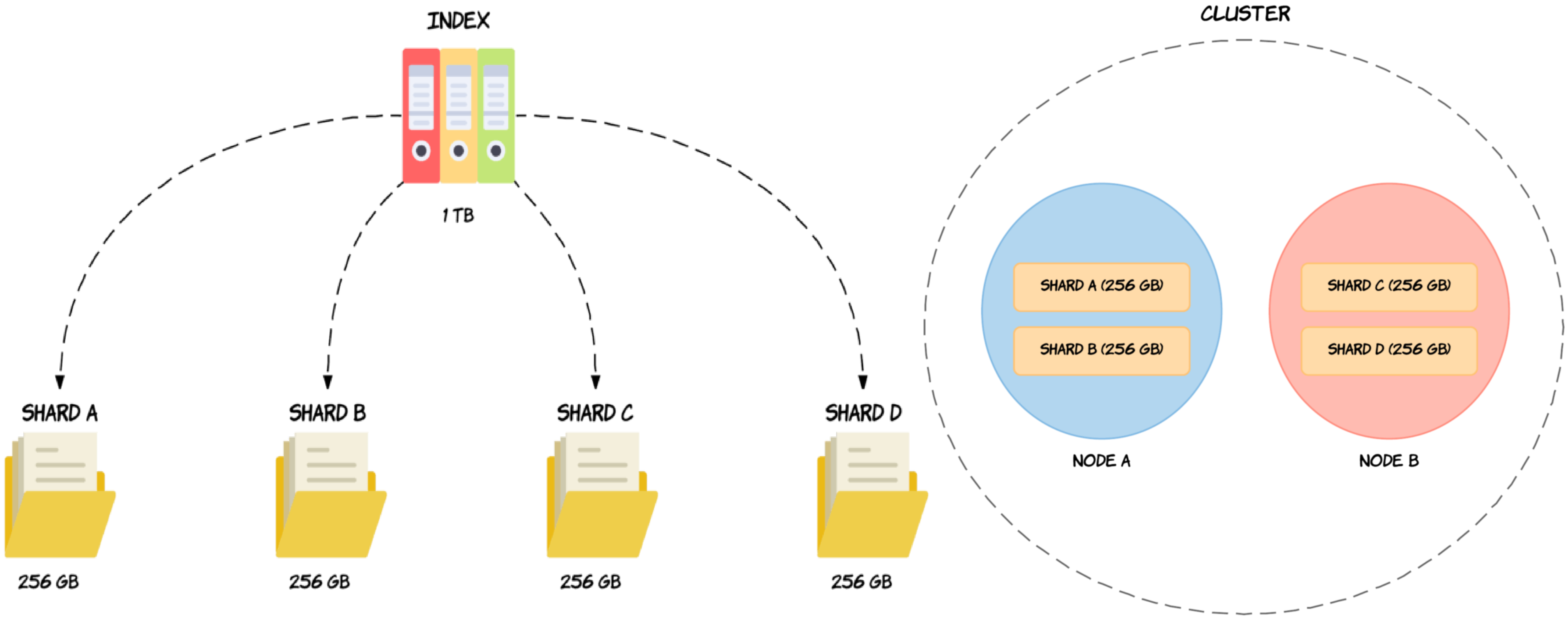
Basic unit of information which can be indexed. It is expressed in JSON (key: value) pair. '{"user": "nullcon"}'. Every single Document is associated with a type and a unique id.

Shard

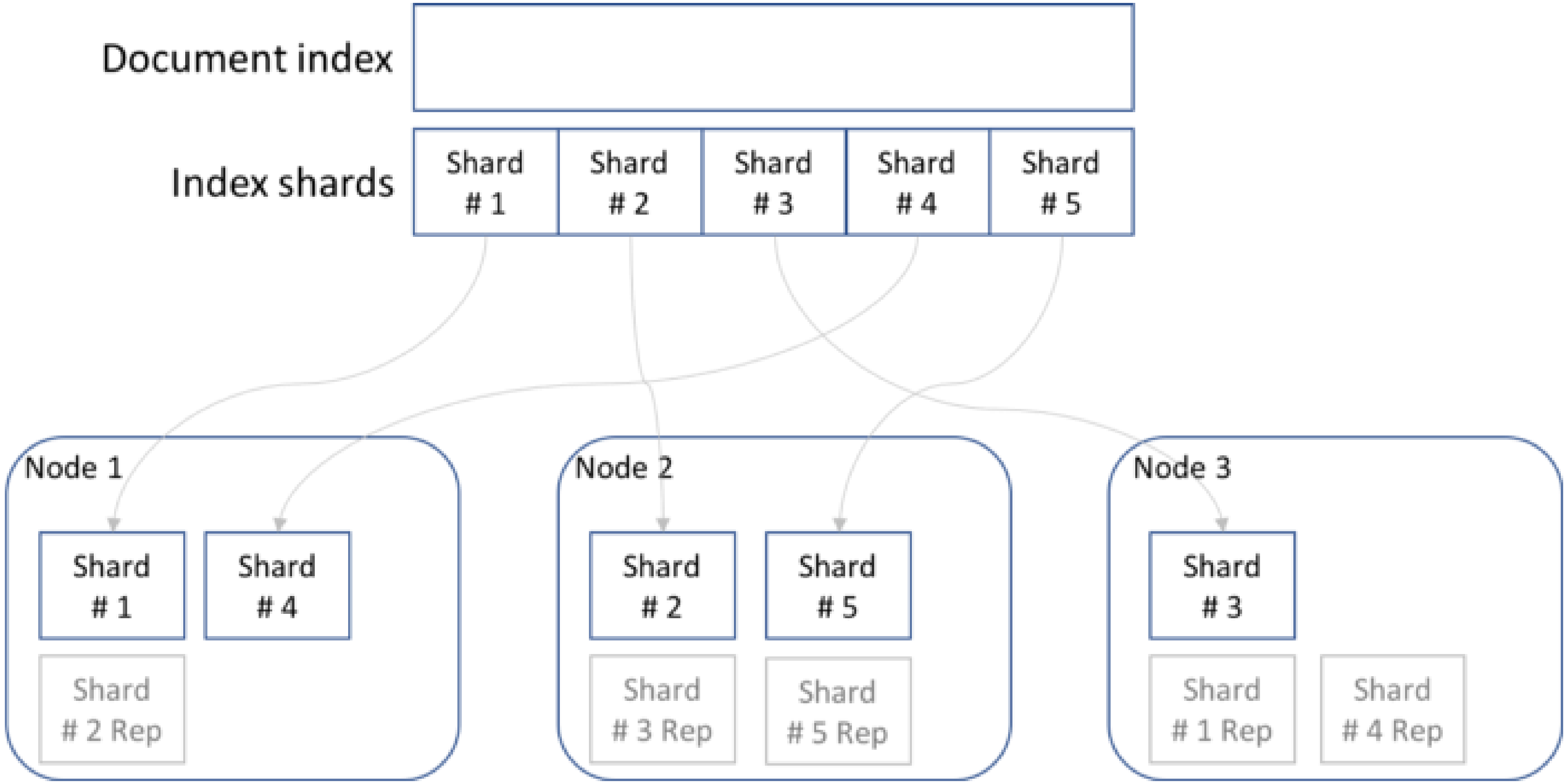
Every **index** can be split into several **shards** to distribute data. The shard is the atomic part of an index, which can be distributed over the cluster to add more nodes.

ARCHITECTURE AND PRINCIPLES

- **Indexes** are split into **shards -** self-contained Lucene indexes
- Shards can be spread across multiple clusters across different machines
- More machines → more clusters → more shards → index is further spread and becomes more efficient
- ES implements redundancy by using Replica shards
- The ES server fetches data by hashing the shard ID containing requested content



HTTPS://CODINGEXPLAINED.COM/CODING/ELASTICSEARCH/UNDERSTANDING-SHARDING-IN-ELASTICSEARCH



HTTPS://MEDIUM.COM/HIPAGES-ENGINEERING/SCALING-ELASTICSEARCH-B63FA400EE9E

HOW NODES WORK

Master Node — Controls the Elasticsearch cluster and is responsible for all cluster-wide operations like creating/deleting an index and adding/removing nodes.

Data Node — Stores data and executes data-related operations such as search and aggregation.

Client Node — Forwards cluster requests to the master node and data-related requests to data nodes.

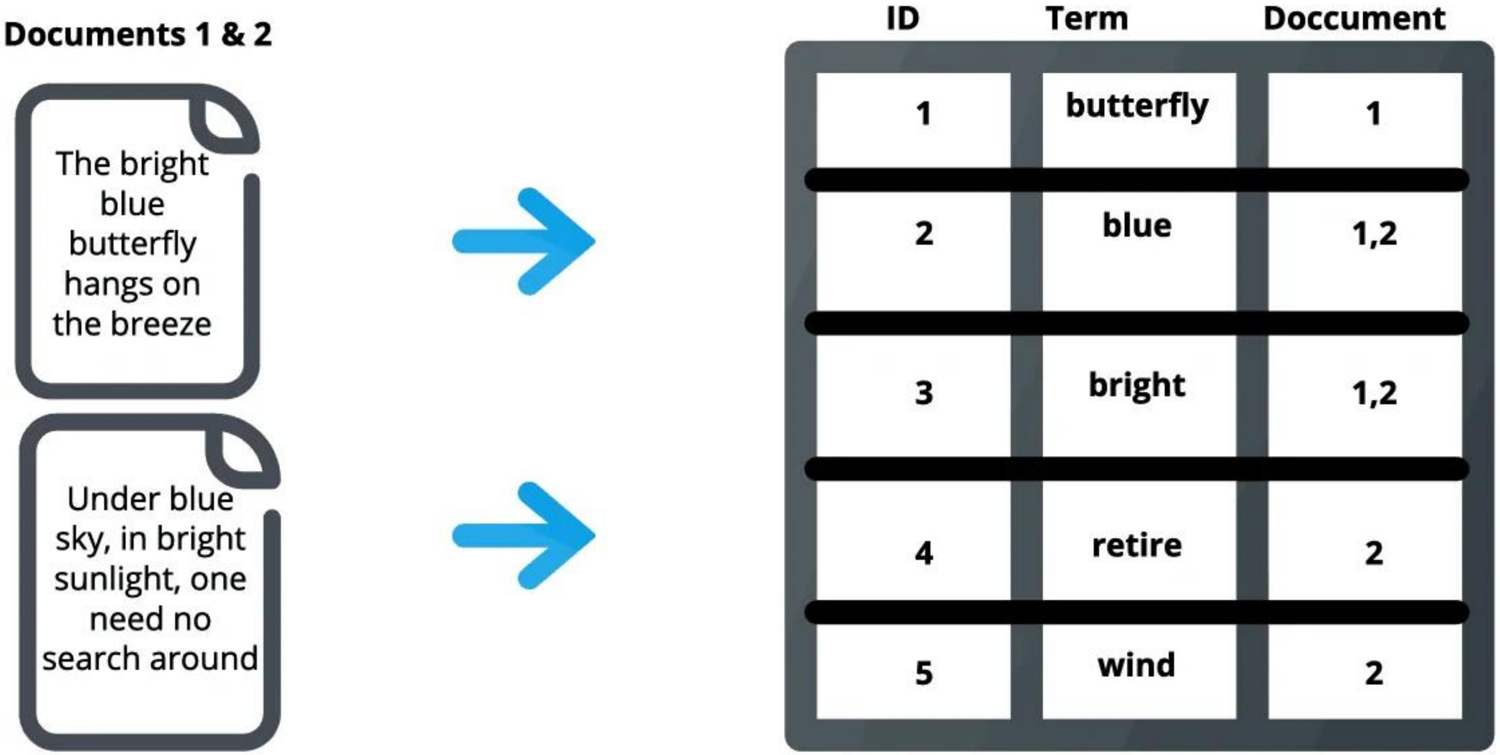
Ingest Node — For pre-processing documents before indexing

HANDLING SEARCHES

- How does ES know where to look for a document? Routing. Routing is handled automatically, Elasticsearch uses a simple hashing formula for determining the appropriate shard.
- Query phase: searches are transformed into a set of searches (one on each shard). Each shard returns its matching documents, the lists are merged, rank, and sorted
- Fetch phase: Get the documents by id from their owning shards and return to the client
- Other functions:
  - Buckets
  - Aggregations
  - Histograms
  - …

INVERTED INDEXING

- On ingestion, simple text processing is applied (lowercasing, removing punctuation, etc.) and the "inverted index" is constructed.
- This dictionary approach is most efficient to find matches based on prefixes.
- Finding terms starting with “b” is easy *O*(log *n*) while finding a substring such as “fly” is a more expensive query *O*(*n*)



Documents 1 & 2

The bright blue butterfly hangs on the breeze

Under blue sky, in bright sunlight, one need no search around

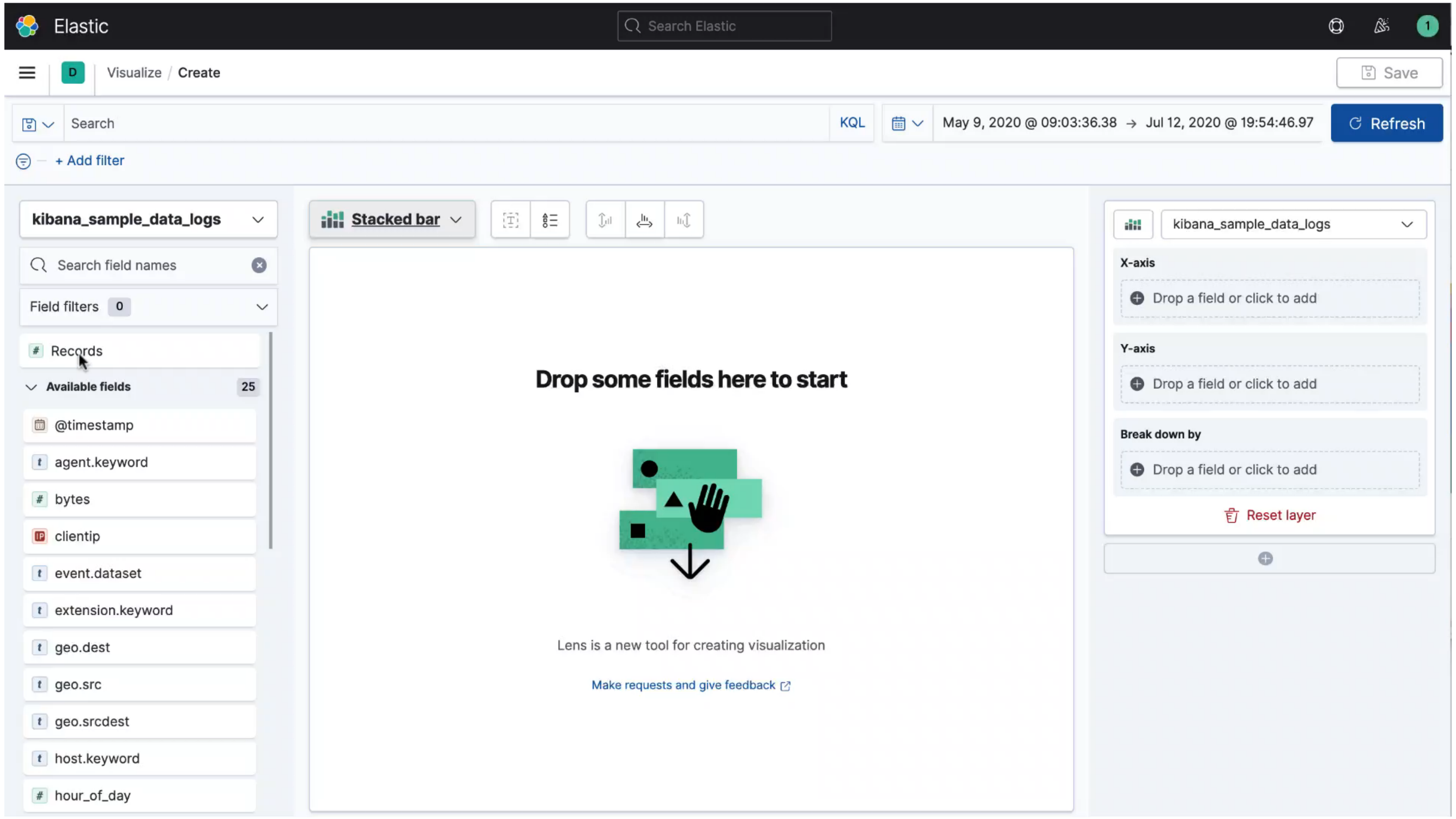
ID	Term	Doccument
1	butterfly	1
2	blue	1,2
3	bright	1,2
4	retire	2
5	wind	2

HTTPS://WWW.KNOWI.COM/BLOG/WHAT-IS-ELASTIC-SEARCH/
HTTPS://WWW.ELASTIC.CO/BLOG/FOUND-ELASTICSEARCH-FROM-THE-BOTTOM-UP

LIMITATIONS, SCALABILITY AND OPTIMIZATION

- *Index size:* Being able to manage huge indexes (in the order of hundreds of Gigabytes or Petabytes)
- *Throughput:* Being able to manage a lot of concurrent searches under a certain response time.
- *Cluster size:* The number of nodes in the system
- These three dimensions of scale are somewhat orthogonal, and an increase on one of them comes at the expense of the others.
- Changing the number of shards does not change how many documents a node manages, but it does change the number of documents per shard. Changing the number of shards trades off search response time with search concurrency
- Increasing the number of nodes in the cluster means that each node manages less documents (horizontal scalability of the index, provided there are enough shards to give something to each node)

BACK TO THE ELK STACK - KIBANA



Drop some fields here to start

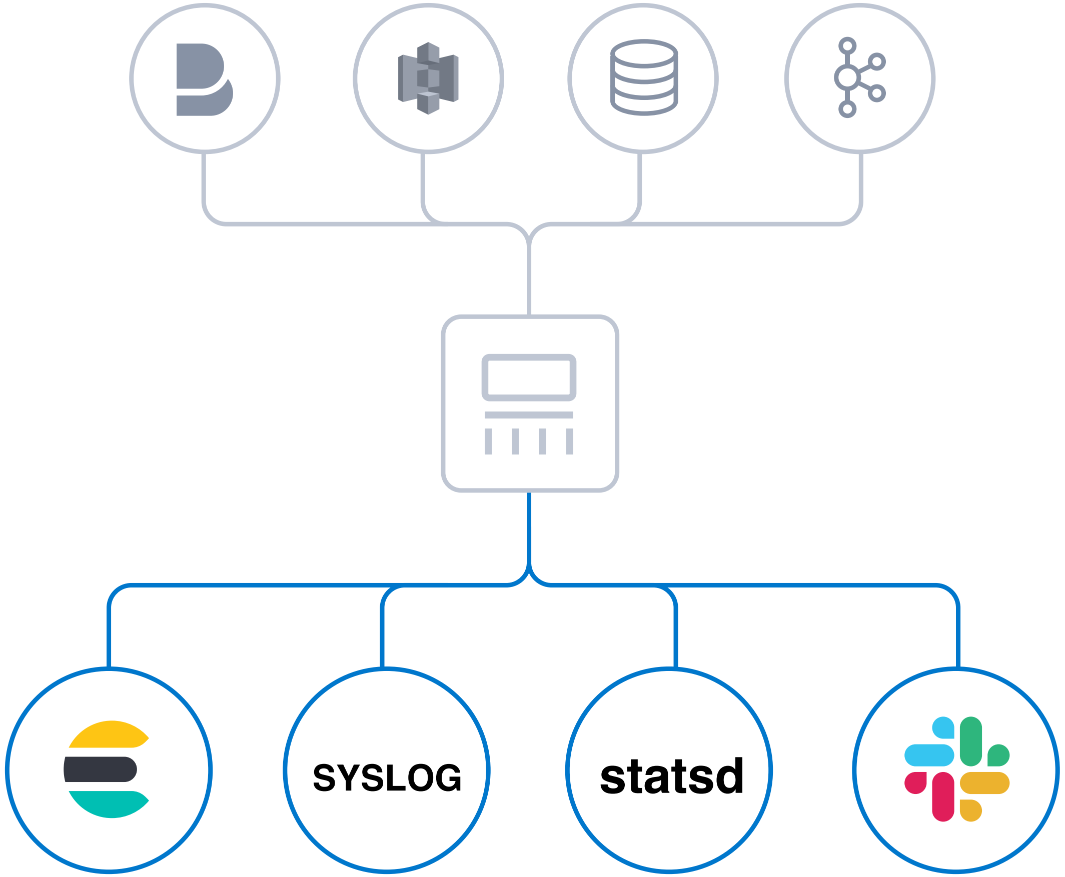
Lens is a new tool for creating visualization

Make requests and give feedback

CLOSING THOUGHTS - BEATS & LOGSTASH

- **Logstash -** Data processing pipeline
- Input, Filter & Output
- Ingests data from a multitude of sources simultaneously, transforms it, then sends it
- Transforms and prepares data regardless of format by identifying named fields to build structure and transform them to converge on a common format.
- Since data is often scattered across different systems in various formats, Logstash ties different systems together like web servers, databases, etc. and publish data to wherever it needs to go in a continuous streaming fashion.

HTTPS://WWW.ELASTIC.CO/LOGSTASH/



BEATS

- Collection of lightweight, single-purpose data shipping agents used to send data from hundreds or thousands of machines and systems to Logstash or Elasticsearch.
- Beats are great for gathering data as they can sit on servers, with containers, or deploy as functions then centralize data in Elasticsearch. For example, Filebeat can sit on a server, monitor log files as they come in, parses them, and import into Elasticsearch in near-real-time.

THANK YOU!

QUESTIONS?