Combining Apache Kafka and the Elastic Stack

Apache Kafka
- Brokers
- Topics
- Partitions
- Consuming topics

Elastic Stack
- Ingesting data into Elasticsearch
- Visualizing data

Combining Apache Kafka and the Elastic Stack

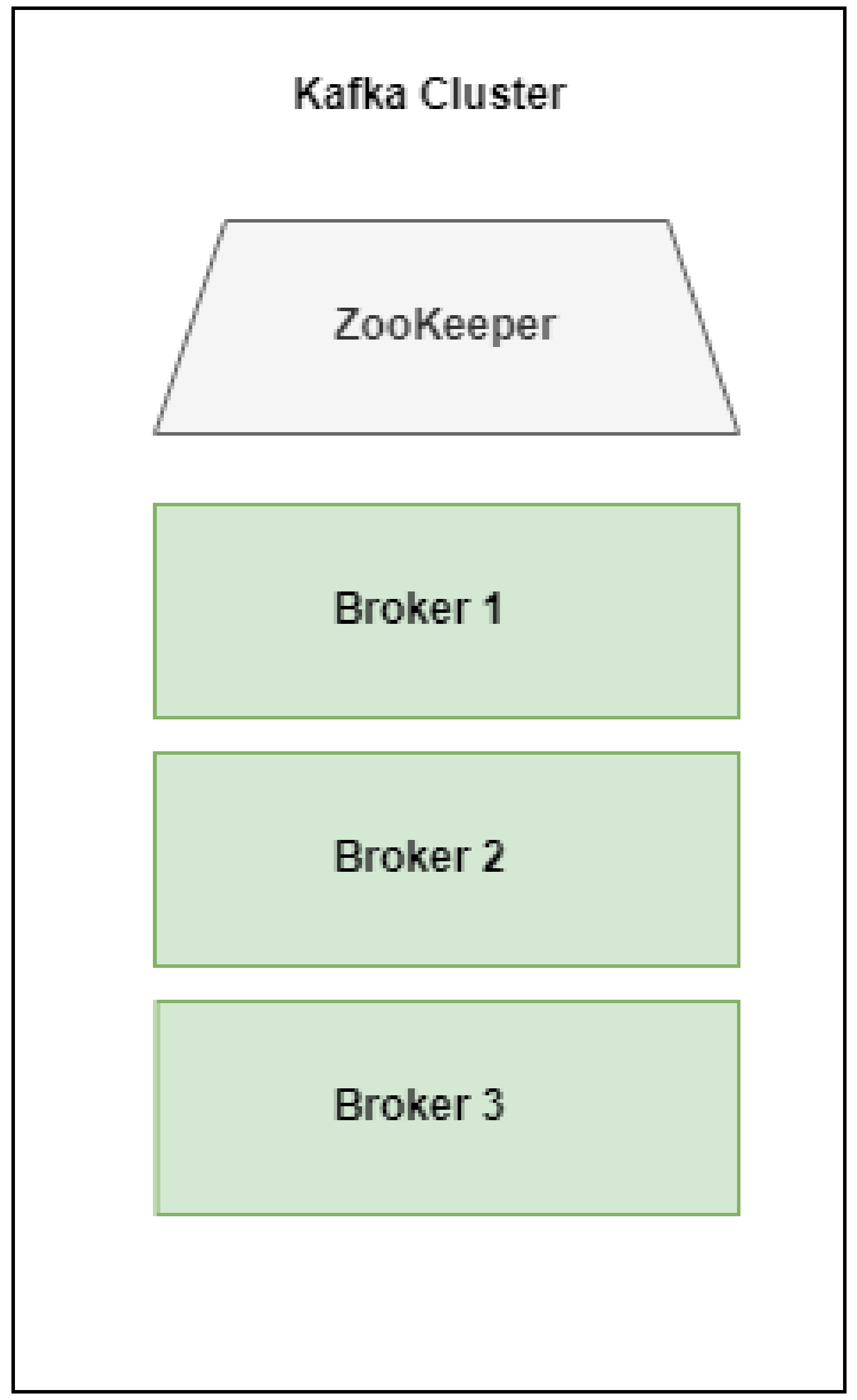
Demo

Apache Kafka

- Open-source distributed event streaming platform
- Provides high-throughput, low-latency
- Kafka Connect can be used to connect with external systems
  - Source Connector and Sink Connector
- REST APIs are also available for producing, consuming and streaming
- Stores data in event logs called **topics**
- Kafka is **pull-based**
  - Consumers are able to ask for new messages when they are ready
  - Makes messages replayable
- Available as both a fully-managed and self-managed service
  - Confluent Cloud is used in the demo

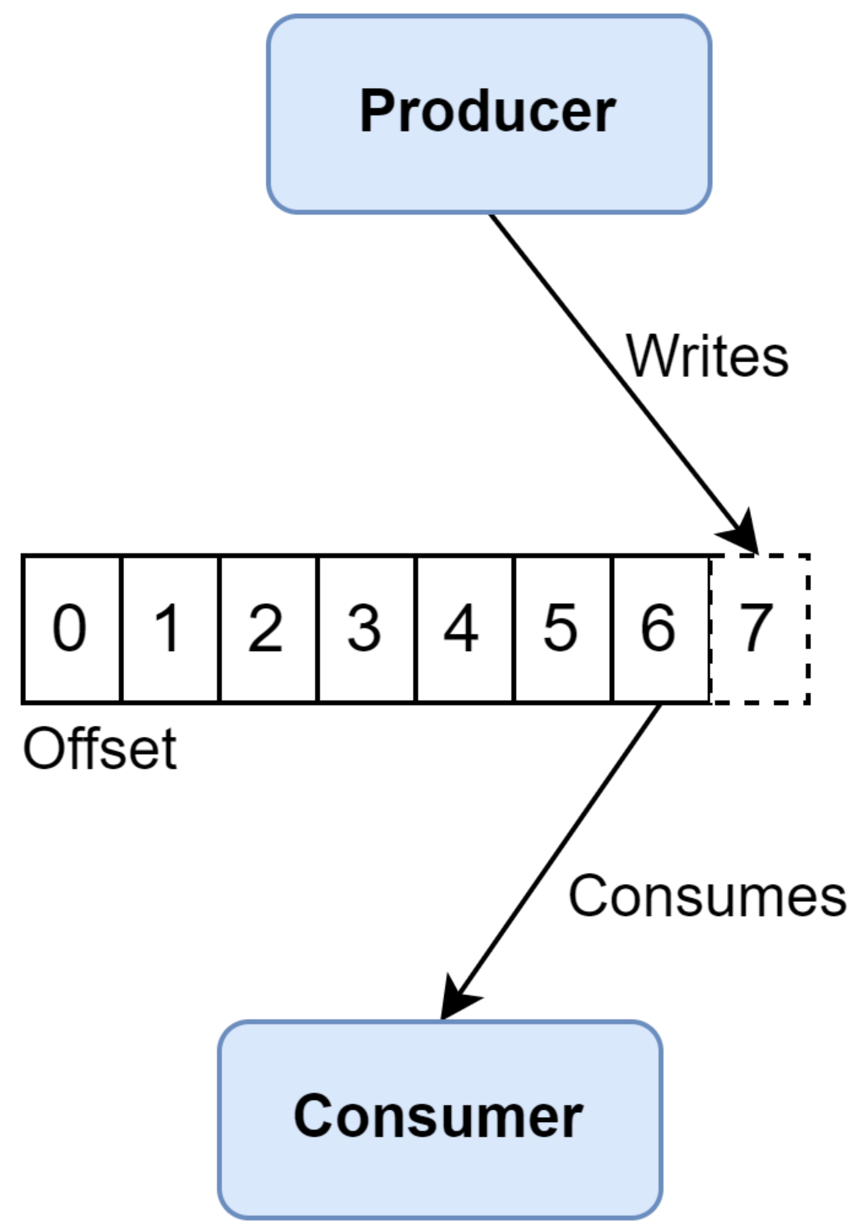
Brokers

- Kafka Brokers are servers that are part of a Kafka Cluster
- **Topics** are stored in brokers
- A Kafka cluster typically consists of several brokers
  - This gives the benefit of data replication as topics are replicated across multiple brokers
- Kafka brokers are stateless, and uses Apache ZooKeeper for maintaining their cluster state



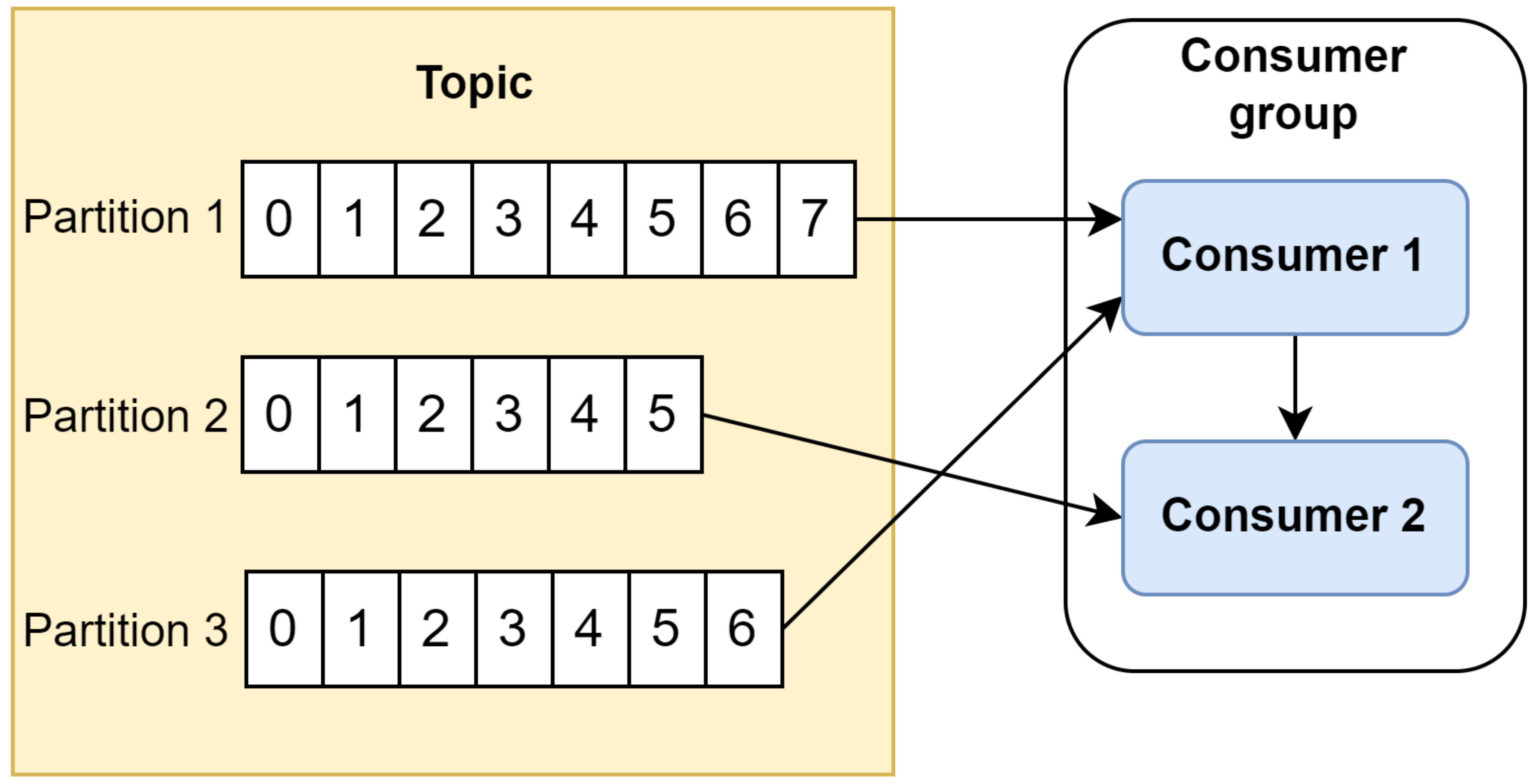
Topics

- Log containing an ordered collection of messages
- Each topic is append-only, and a message is given an incremental **offset** upon storage
  - A message consists of a key, value, offset, timestamp and headers
- Topics can have multiple **producers** and **consumers**
- A topic can store data anywhere from a short amount of time, to indefinitely
  - This is decided upong creation by the chosen retention policy
  - Upon expiry, messages are marked for deletion
  - Compaction policy retains only the most recent message for each key
- **Schema Registry** allows us to enforce schemas on messages
- Topics are divided into **partitions**



Partitions

- The smallest storage unit in Kafka
- Each partition holds a subset of messages in a topic
- A topic can have one or several partitions
  - More partitions allows for more parallelism
- The offset of a message is based on its partition
  - The offset guarantees the order within a partition, but not across the topic
- A message can be assigned to a specific partition by specifying a **partition key**
  - If no partition key is provided, Kafka will use round-robin assignment
- Each partition is assigned to exactly one consumer within a **consumer group**



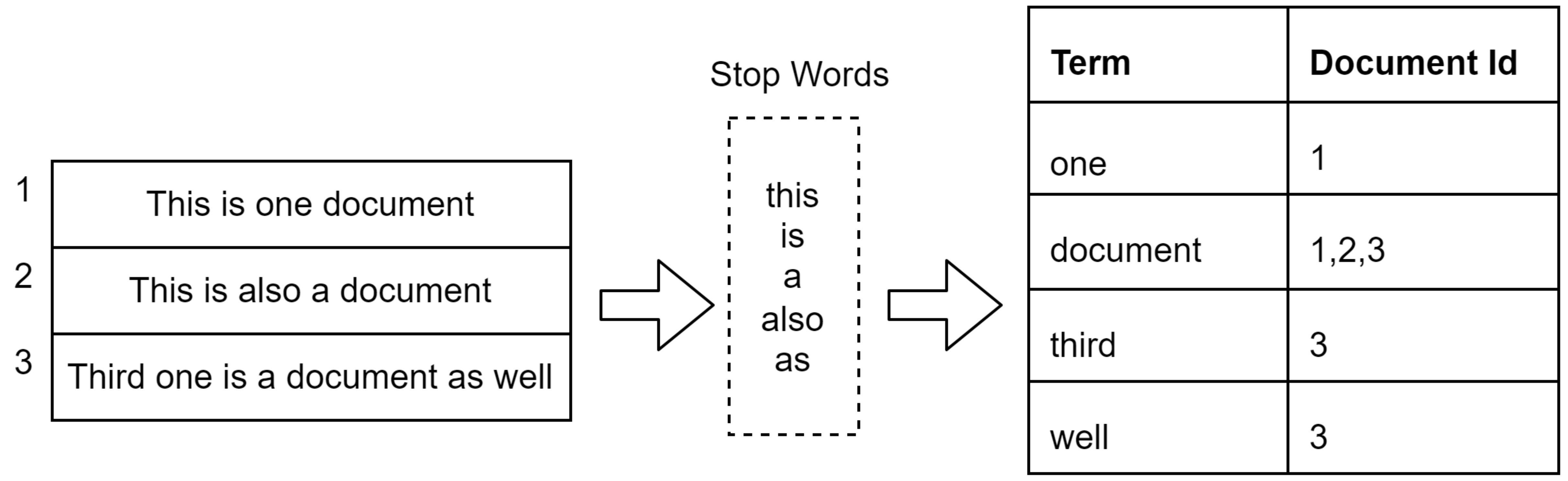
Consuming topics

- Consumers join consumer groups by using the same **group.id**
- The maximum parallelism of a consumer group is the number of partitions for the consumed topic
  - Number of consumers > number of topic partitions will leave consumers idle
- Consume topics through Sink Connectors and the Consume API
- The internal topic `_consumer_offsets` keeps track of each groups current offset
  - Consumers can themselves reset their own offset to any position

- **Kafka Streams** lets us build a streaming application which transforms an input topic into new topics
  - Aggregating, filtering, grouping, joins ++
- **ksqlDB** allows us to perform stream processing tasks using SQL statements

Elastic Stack

- Consists of four tools, **Elasticsearch**, **Logstash**, **Kibana** and **Beats**
- Elasticsearch is a Lucene-based search and analytics engine
- It is distributed and RESTful
- Useful for searching great amounts of data in near-real time
- Documents are indexed and stored in **indices**
  - The generated **inverted indices** tells Elasticsearch which words appear in which document



Term	Document Id
one	1
document	1,2,3
third	3
well	3

Ingesting data into Elasticsearch

- REST API
- Logstash can be used to process each incoming message
  - Powerful and flexible tool
  - Higher hardware requirements than Beats
- Beats has been introduced as lightweight data shipper
- Both Logstash and Beats can be used in combination
- In the demo we will be ingesting data using the Confluent Elasticsearch Sink Connector
  - Very easy to setup and fully-managed in Confluent Cloud

elasticsearch

Visualizing data

- Kibana is a data visualization dashboard software
- Provides a UI to explore the data in the Elasticsearch indices
- Has a number of different features included
  - Metrics
  - Charts
  - Maps
  - Anomaly detection
  - ++

kibana

Combining Apache Kafka and Elastic Stack

- Elasticsearch can be queried through a REST API
  - Elasticsearch clients are available for a lot of different programming languages
- Make real-time events searchable
- Visualize real-time events using Kibana
- Very easy to consume Kafka topics using the fully-managed Elasticsearch Sink Connector in Confluent Cloud
  - Many more fully-managed and self-managed connectors are available
  - It is also possible to configure Logstash to consume topics, and preprocess messages before they are stored

Demo