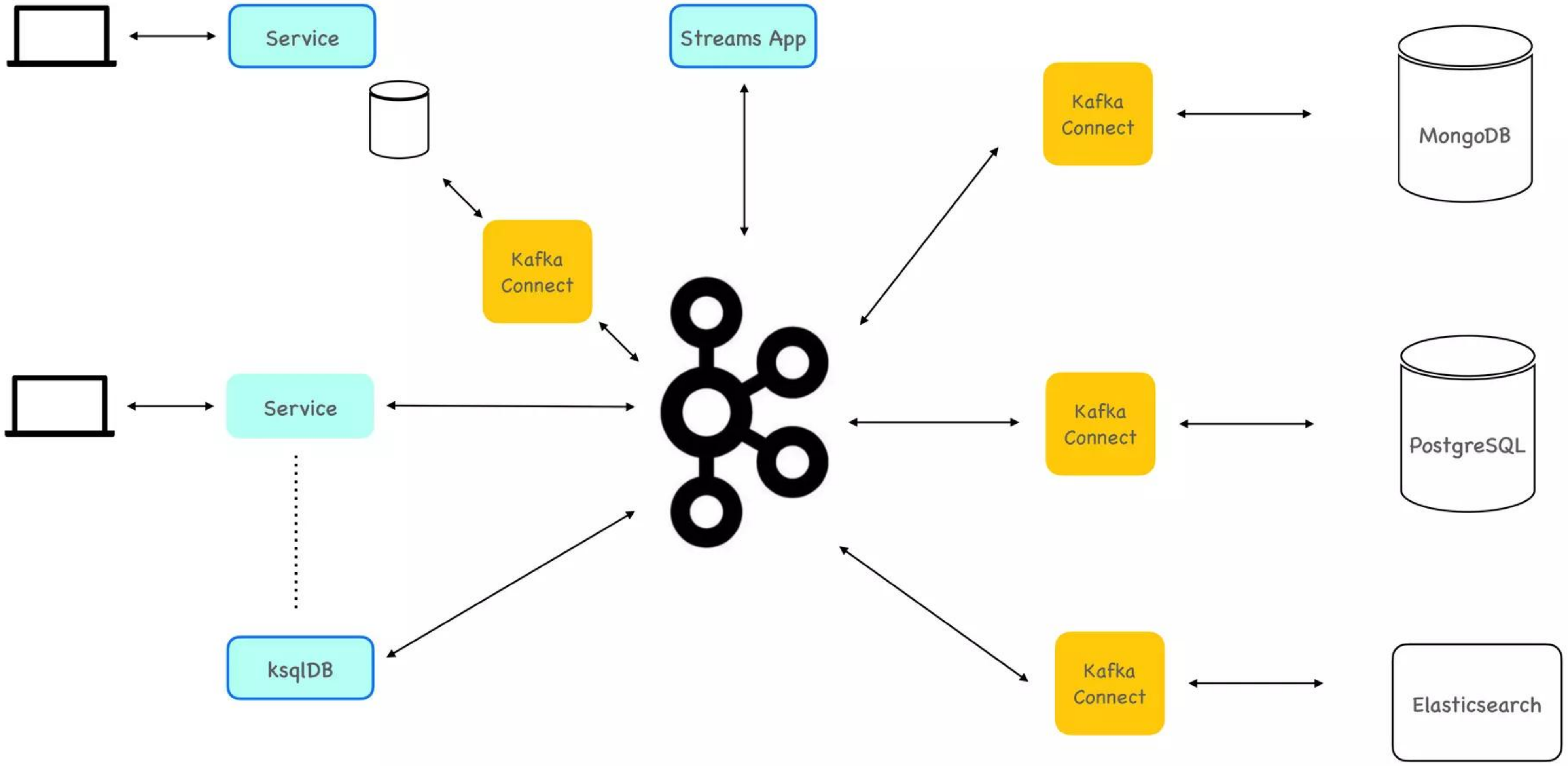
Wonderful World of Apache Kafka

@daveklein

@confluentinc

Event Streaming Platform



Event

Notification

Order Placed

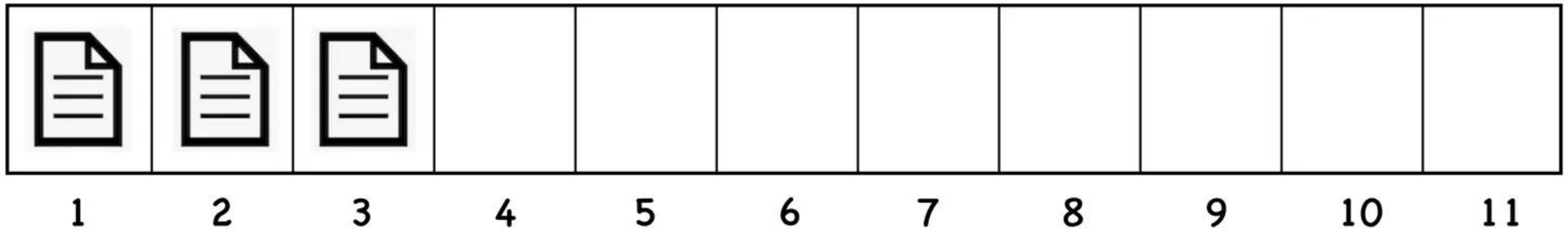
Temperature Read

State

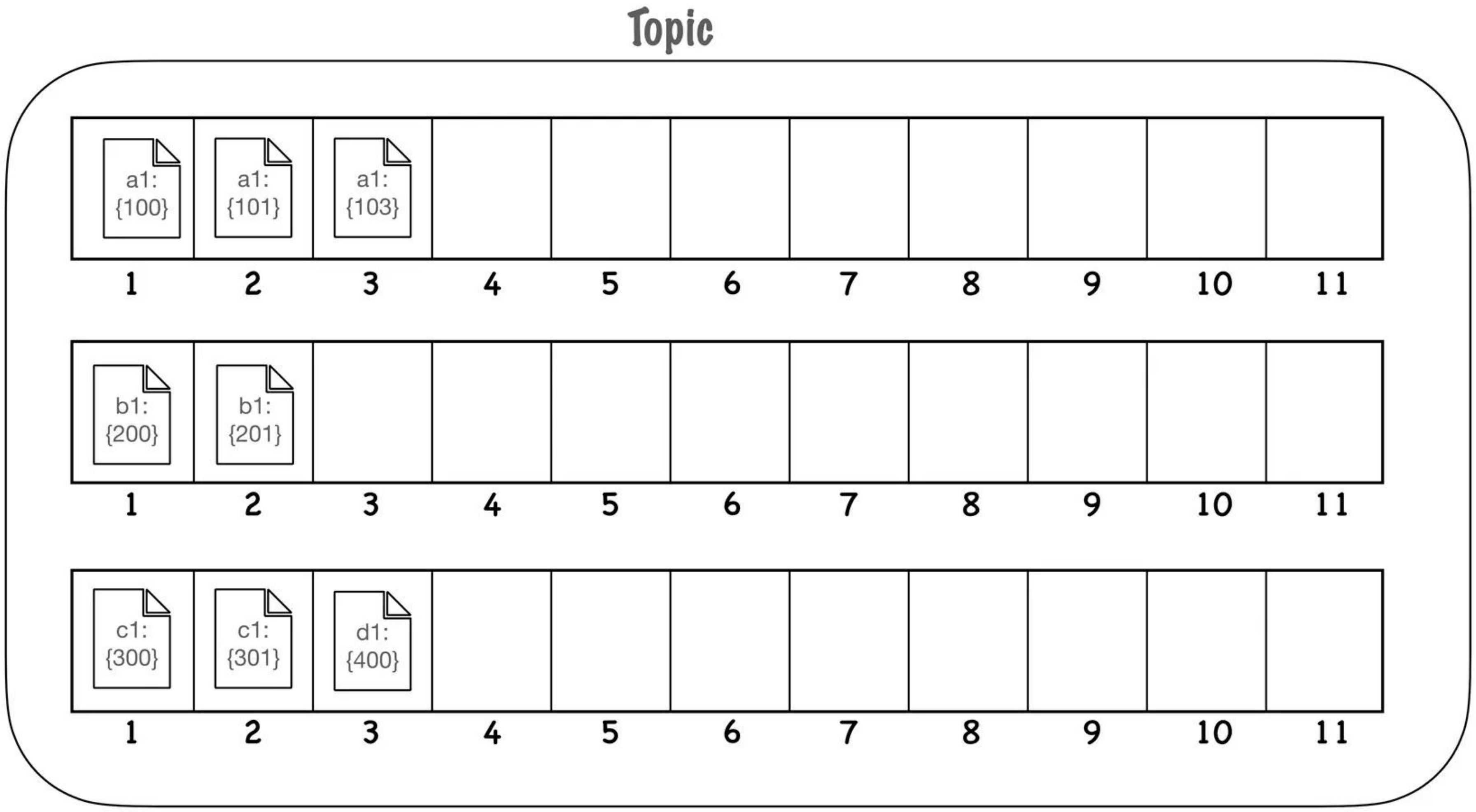
{item:123, price: 29.95,
 qty: 2}

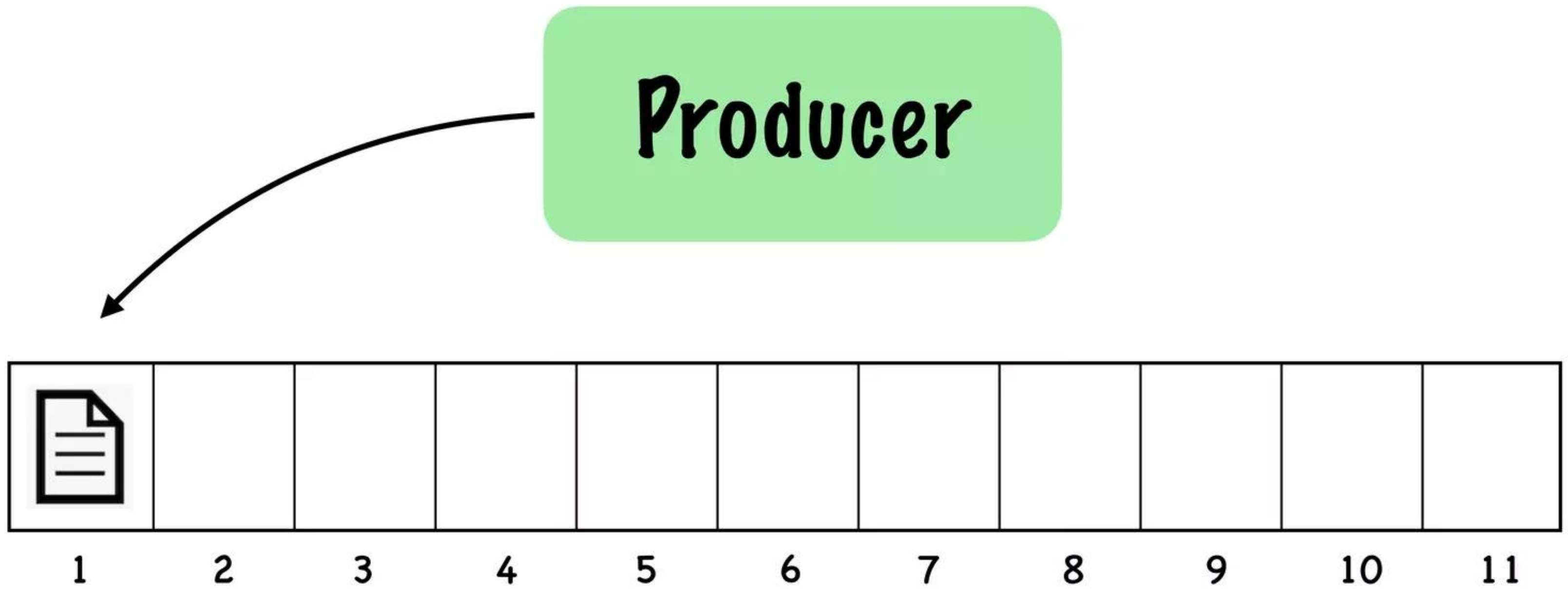
{temp:29, unit: F,
 time: 1606949836369}

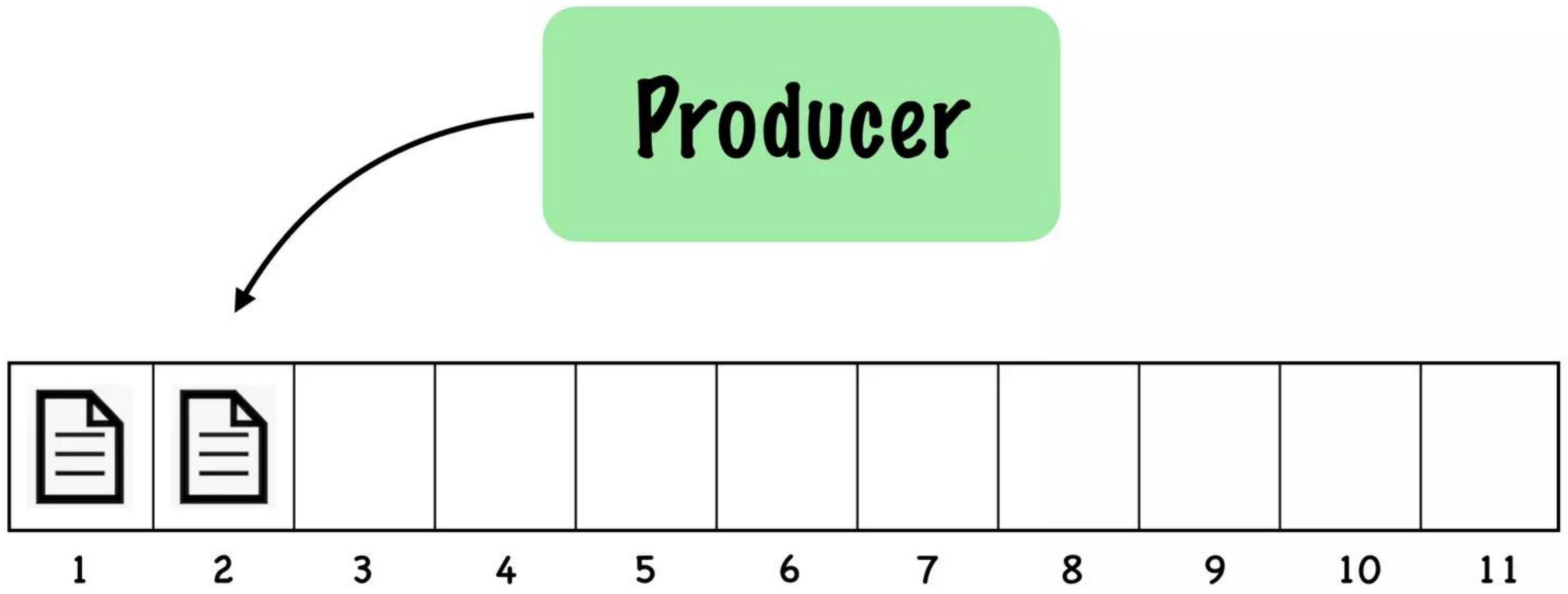
Topic (log)

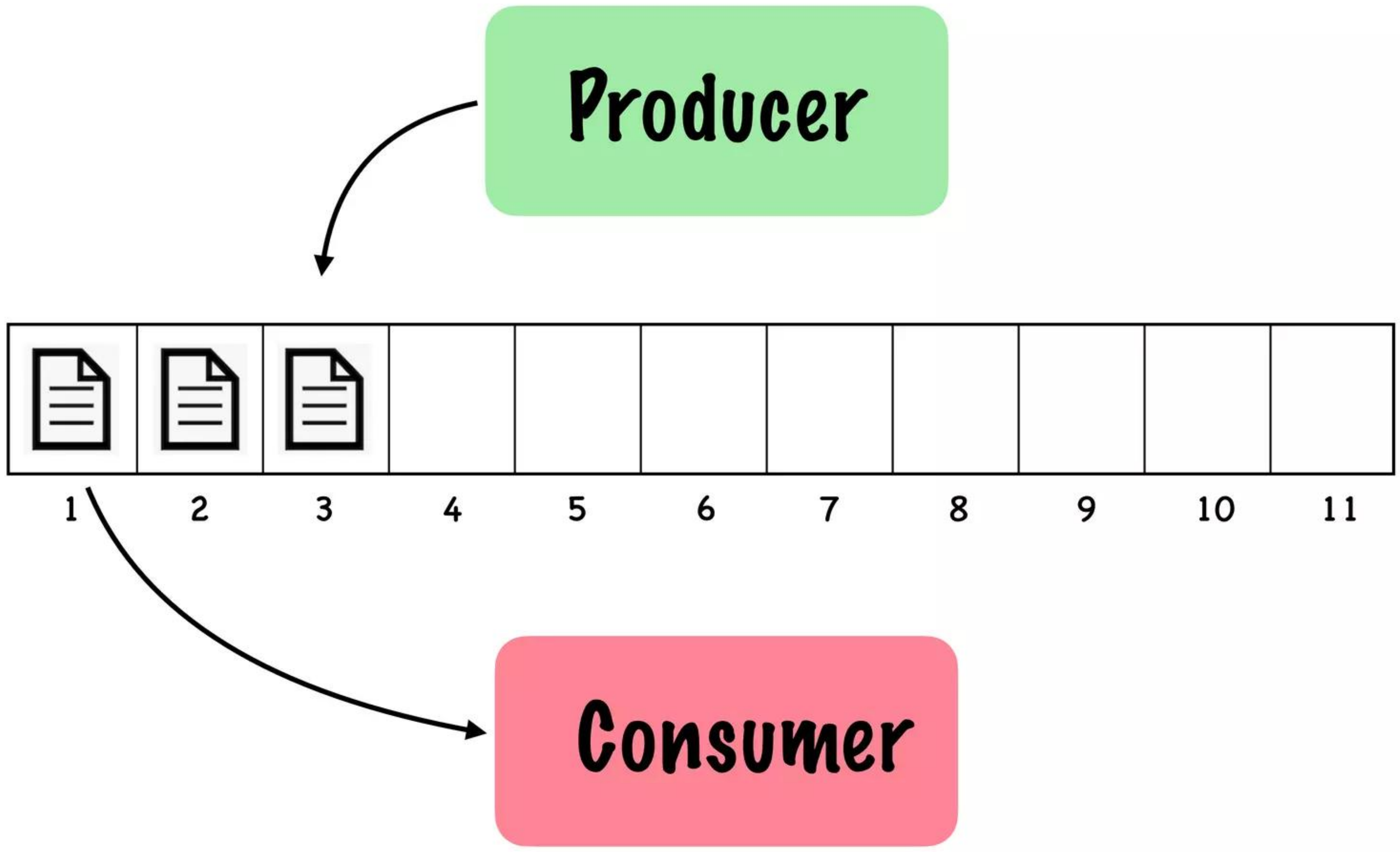


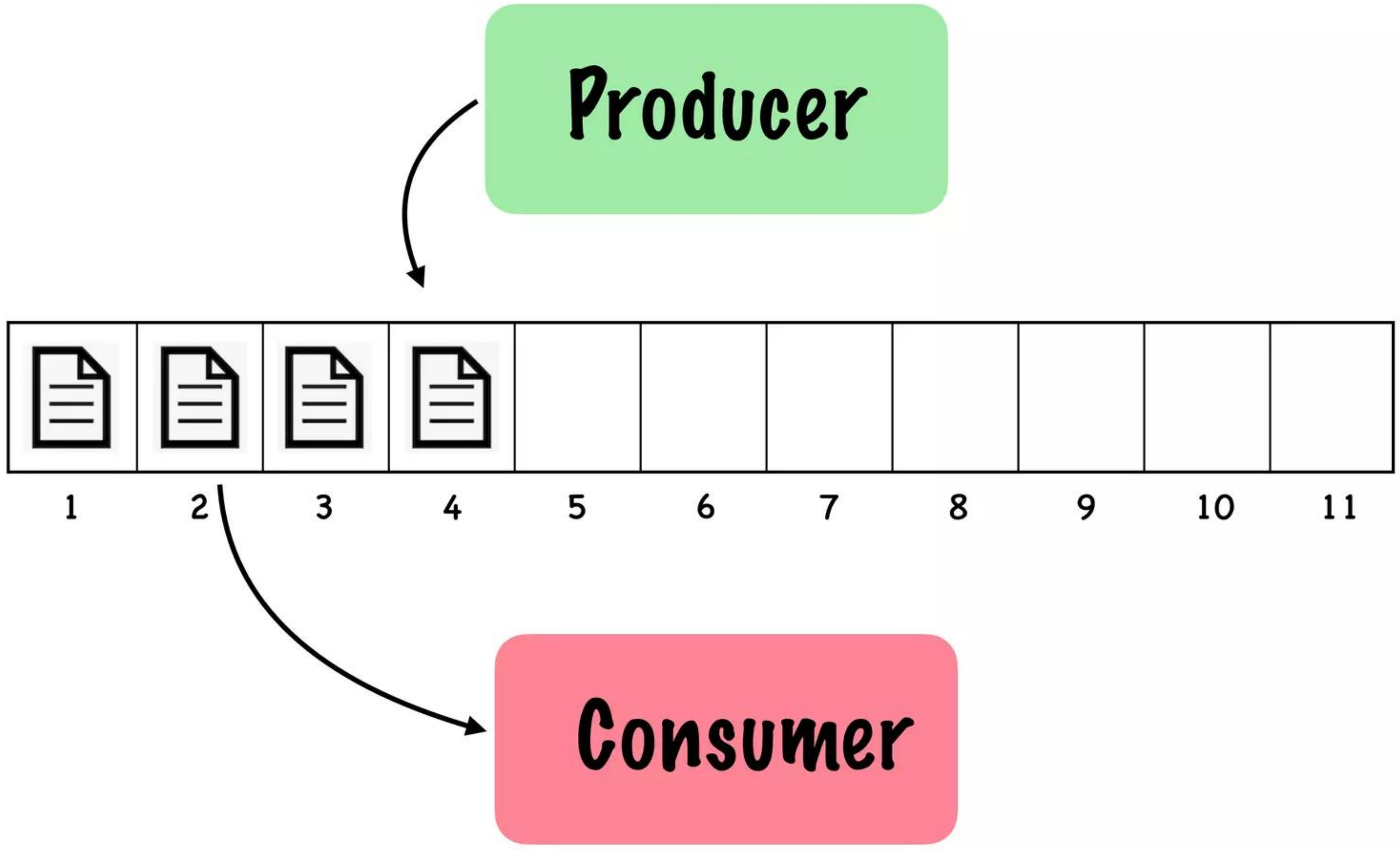
```
"key": "a1",
"value": {
    "eventType": “add-to-cart",
    "title": "Kafka Streams in Action”,
    "author": "Bill Bejeck",
    "price": 44.99
}
```

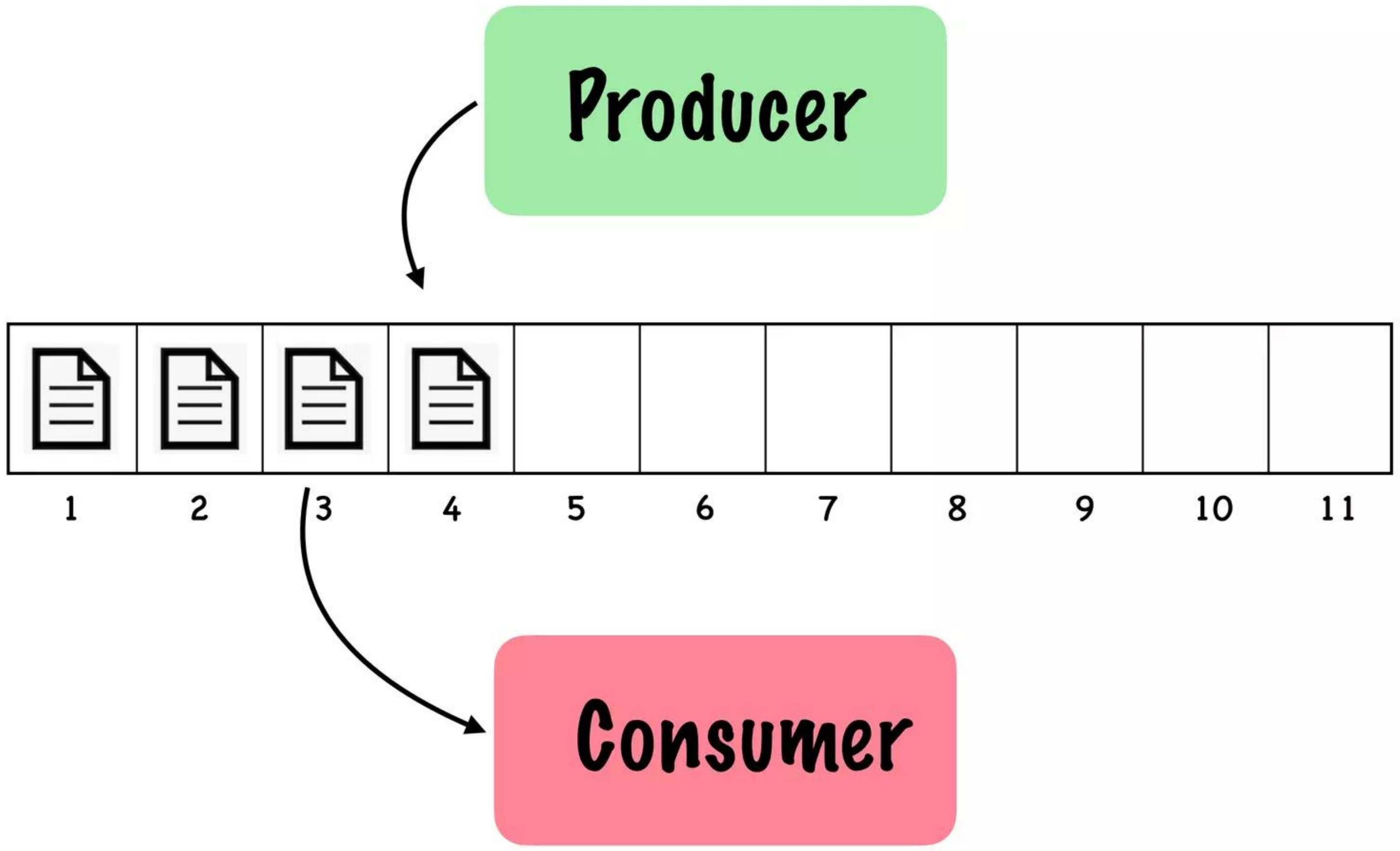

Topic
a1: {100}
a1: {101}
a1: {103}
1
2
3
4
5
6
7
8
9
10
11
b1: {200}
b1: {201}
1
2
3
4
5
6
7
8
9
10
11
c1: {300}
c1: {301}
d1: {400}
1
2
3
4
5
6
7
8
9
10
11

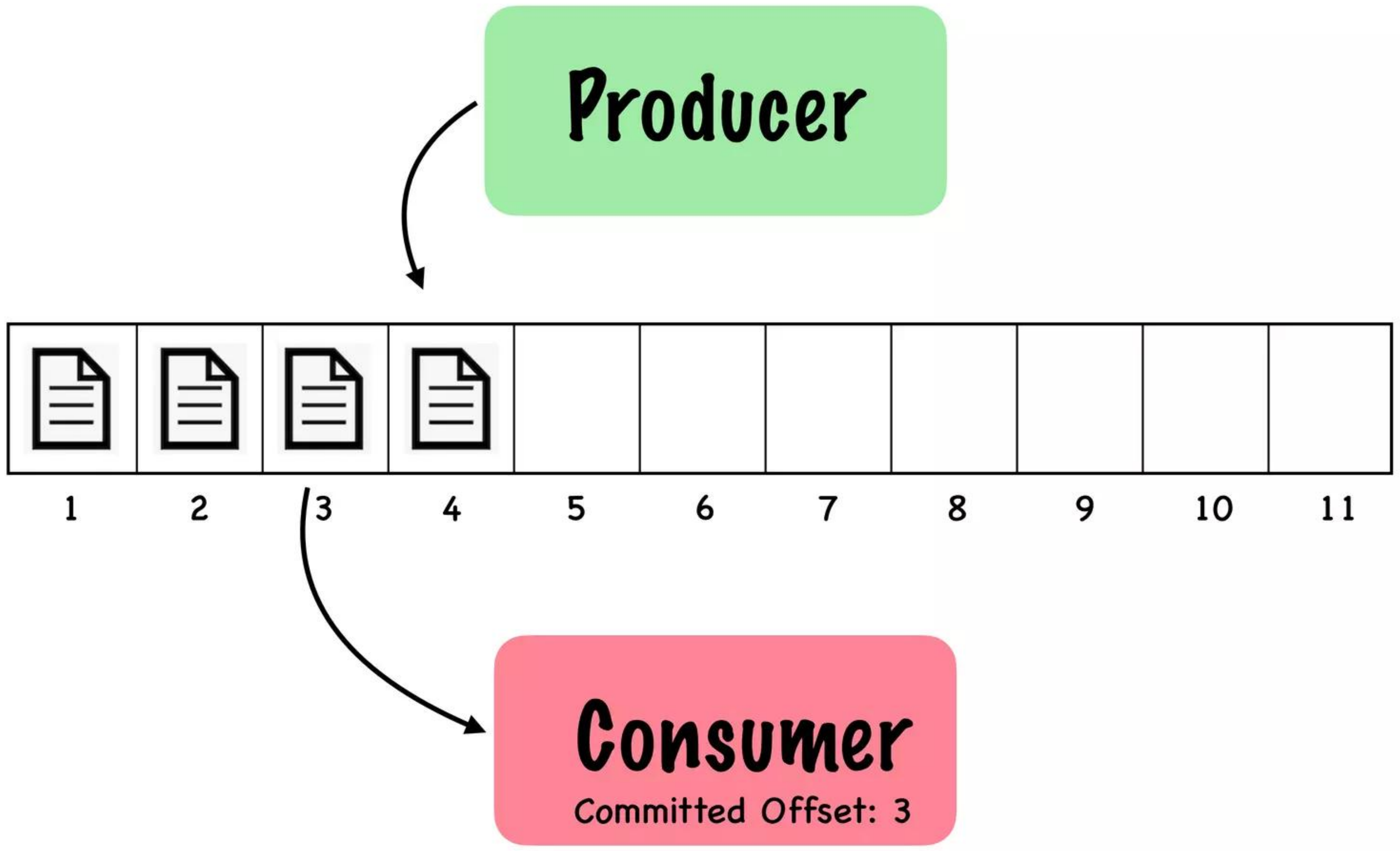

Producer
1
2
3
4
5
6
7
8
9
10
11

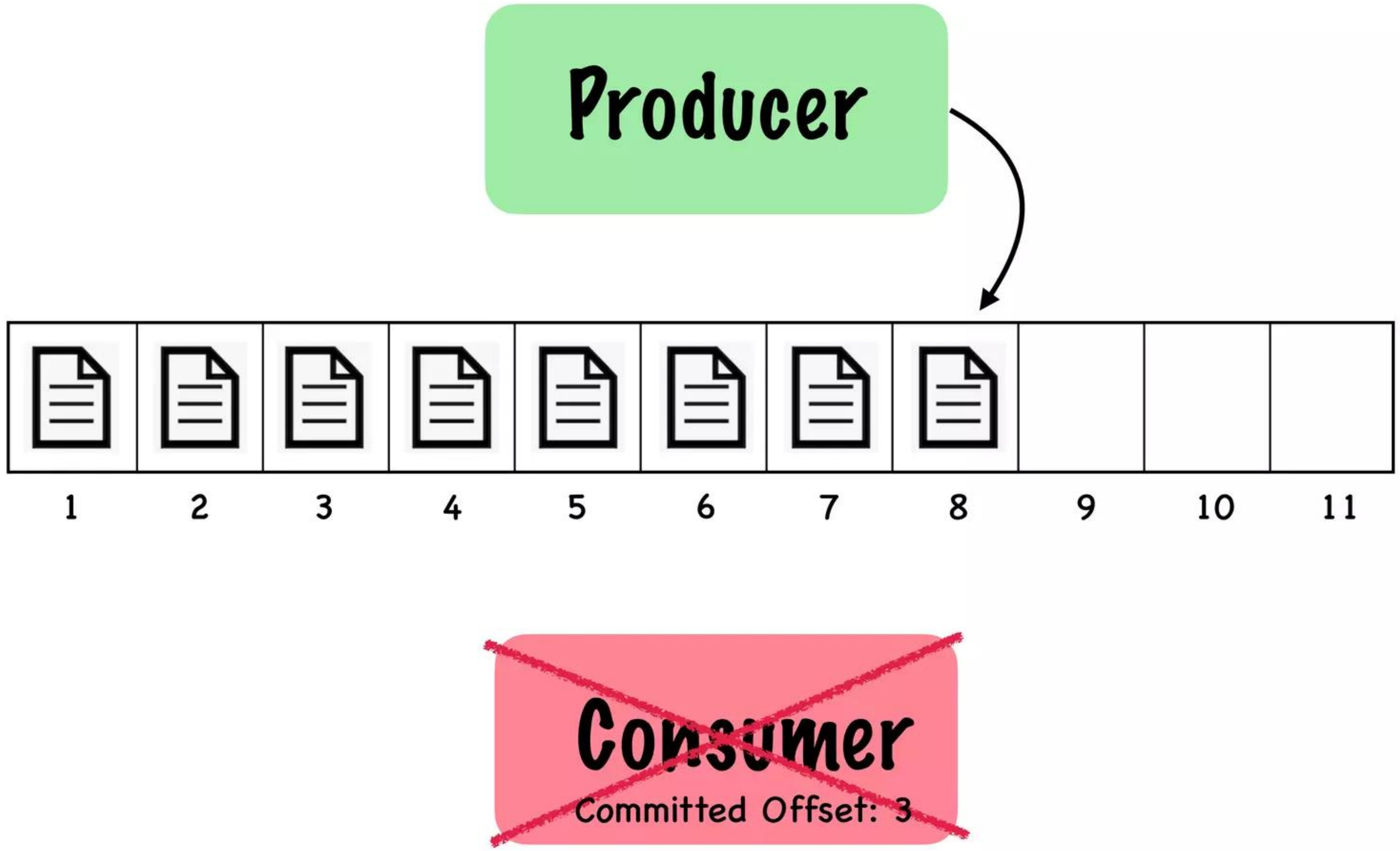

Producer
1
2
3
4
5
6
7
8
9
10
11

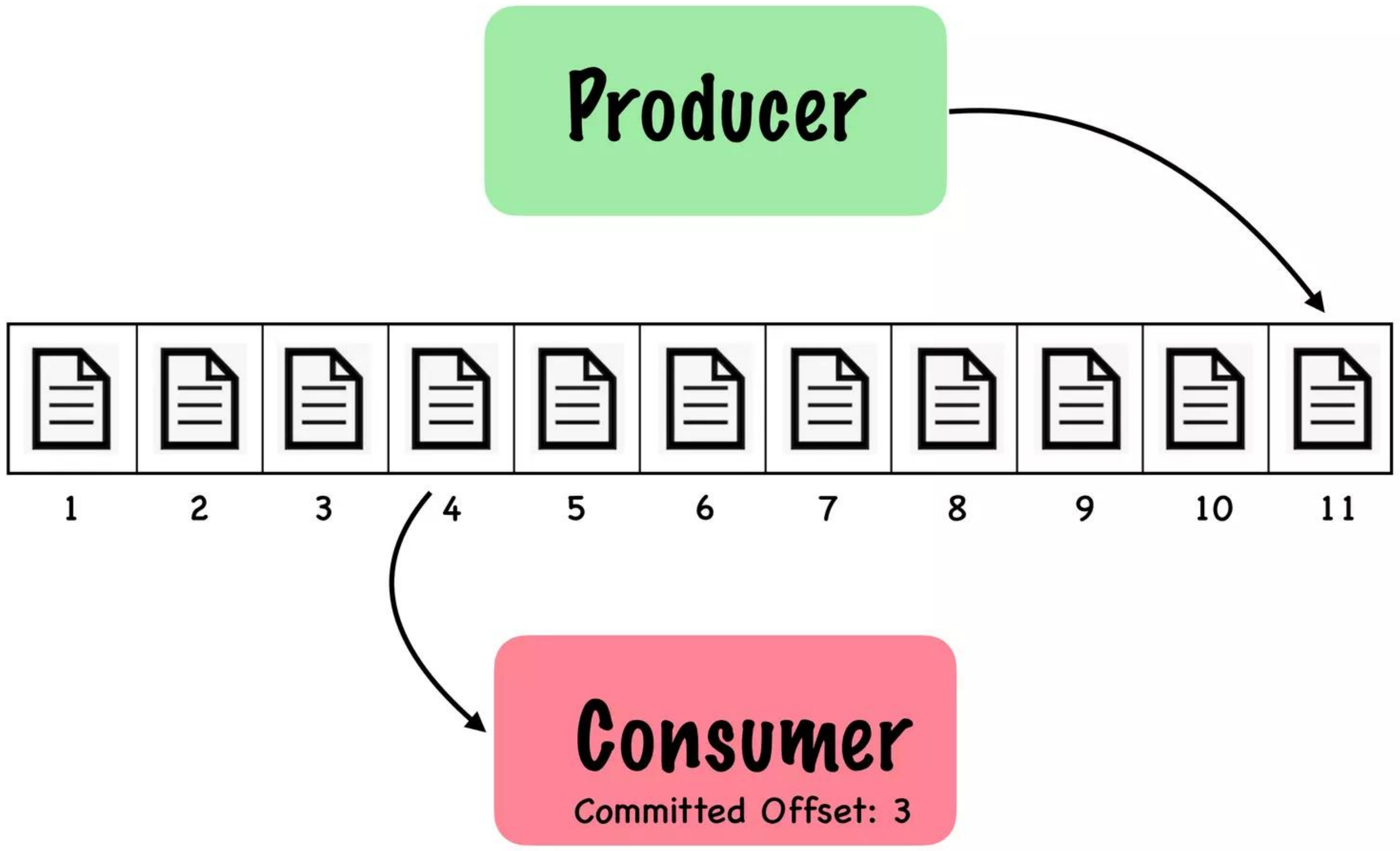

Producer
1
2
3
4
5
6
7
8
9
10
11
Consumer


Producer
1
2
3
4
5
6
7
8
9
10
11
Consumer


Producer
1
2
3
4
5
6
7
8
9
10
11
Consumer


Producer
1
2
3
4
5
6
7
8
9
10
11
Consumer
Committed Offset: 3


Producer
1
2
3
4
5
6
7
8
9
10
11
~~Consumer~~
~~Committed Offset: 3~~

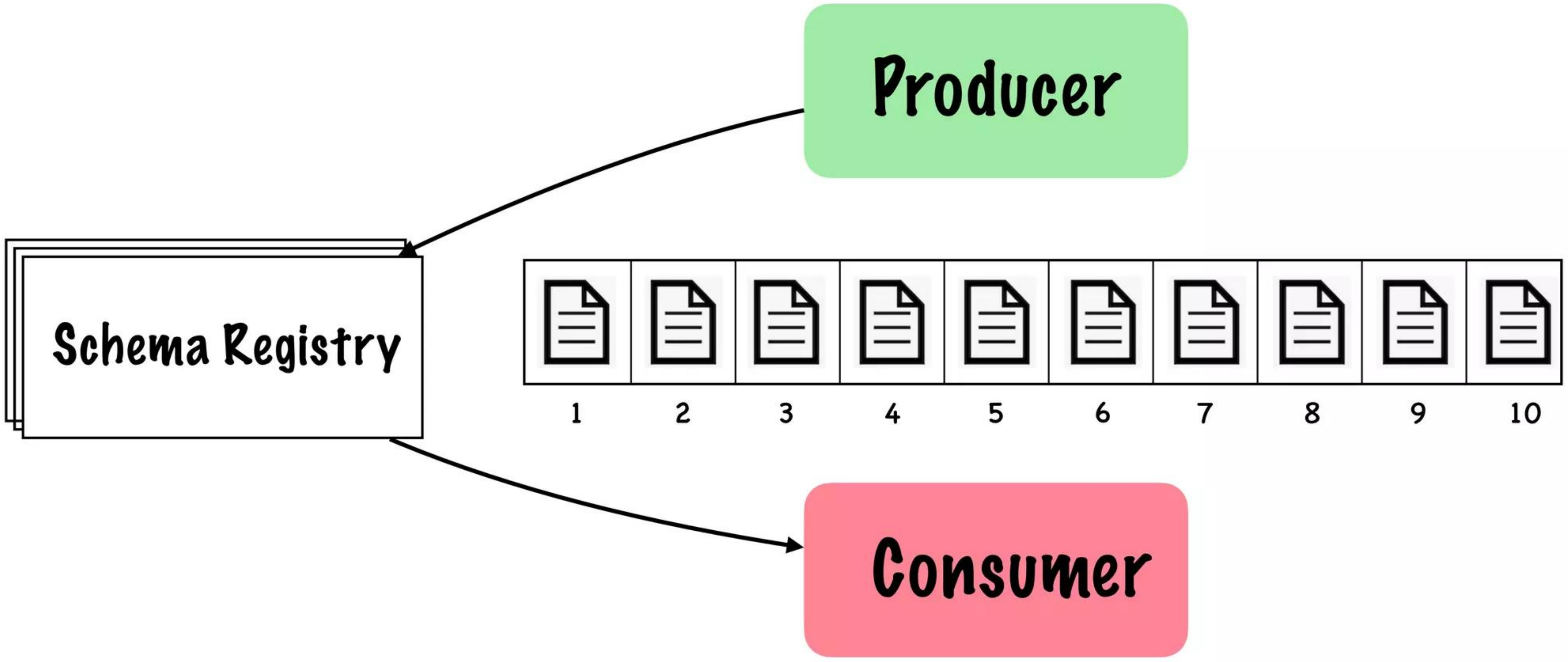

Producer
1
2
3
4
5
6
7
8
9
10
11
Consumer
Committed Offset: 3

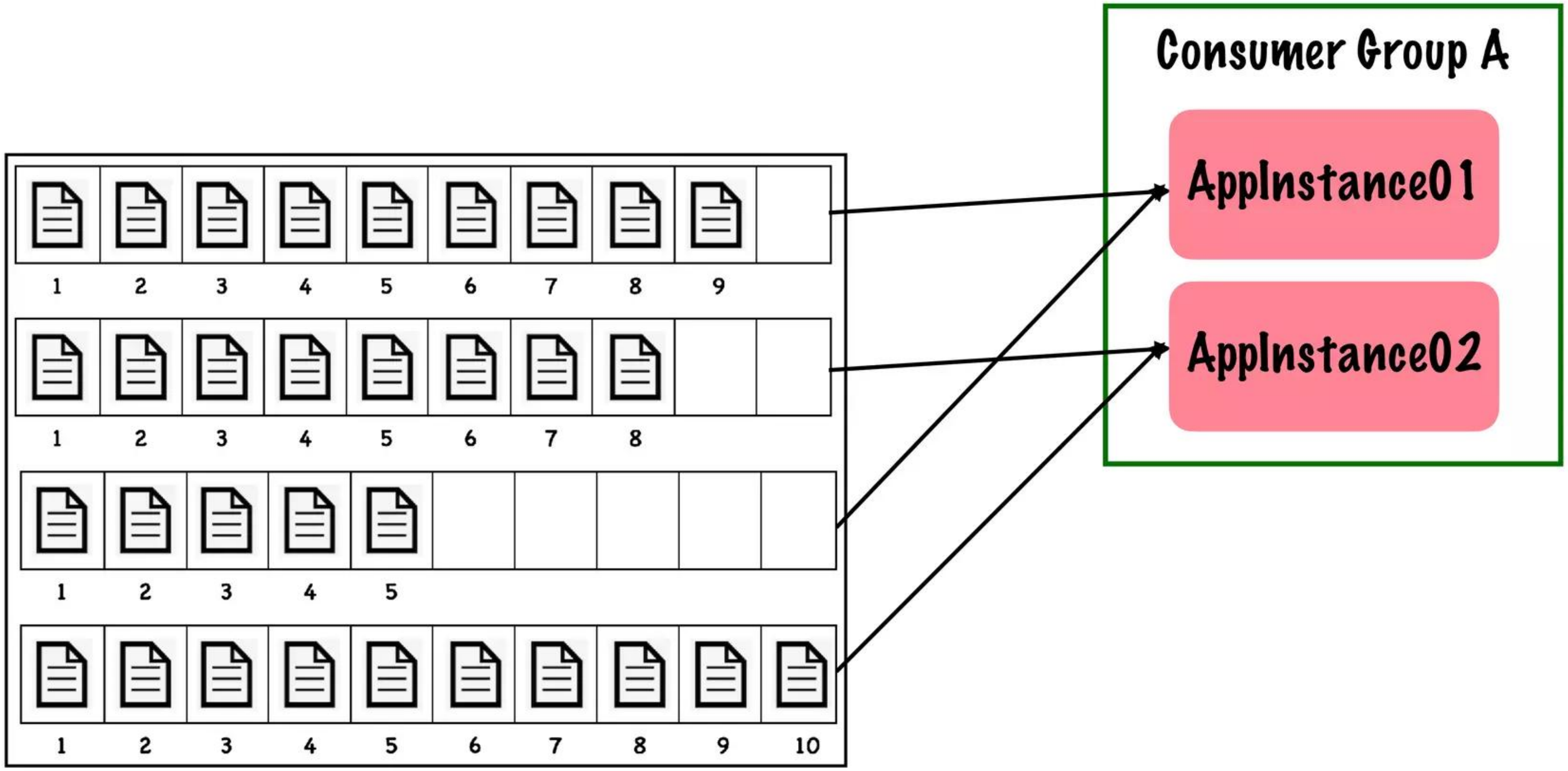
Sample Kafka Producer

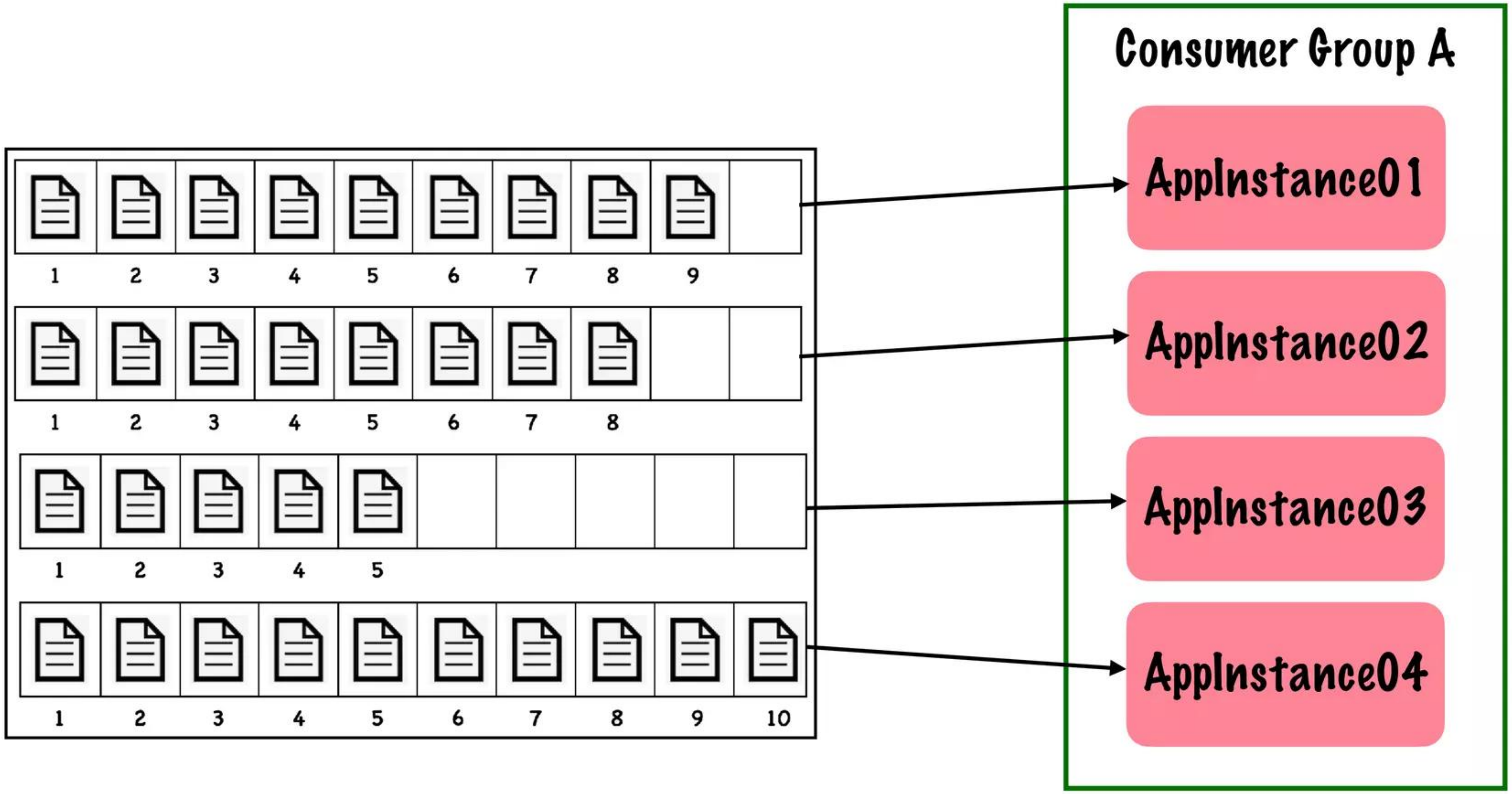
```
Producer<String, String> producer = new KafkaProducer<String, String>(props);

String key = "acme";
String value = "some info about acme, probably in JSON”;
ProducerRecord record = new ProducerRecord<String, String>(topic, key, value);

producer.send(record, new Callback(){
    @Override
    public void onCompletion(RecordMetadata m, Exception e) {
        if (e != null) {
            e.printStackTrace();
        } else {
            System.out.printf("Produced record to topic %s partition [%d]",
                m.topic(), m.partition());
        }
    }
});
```

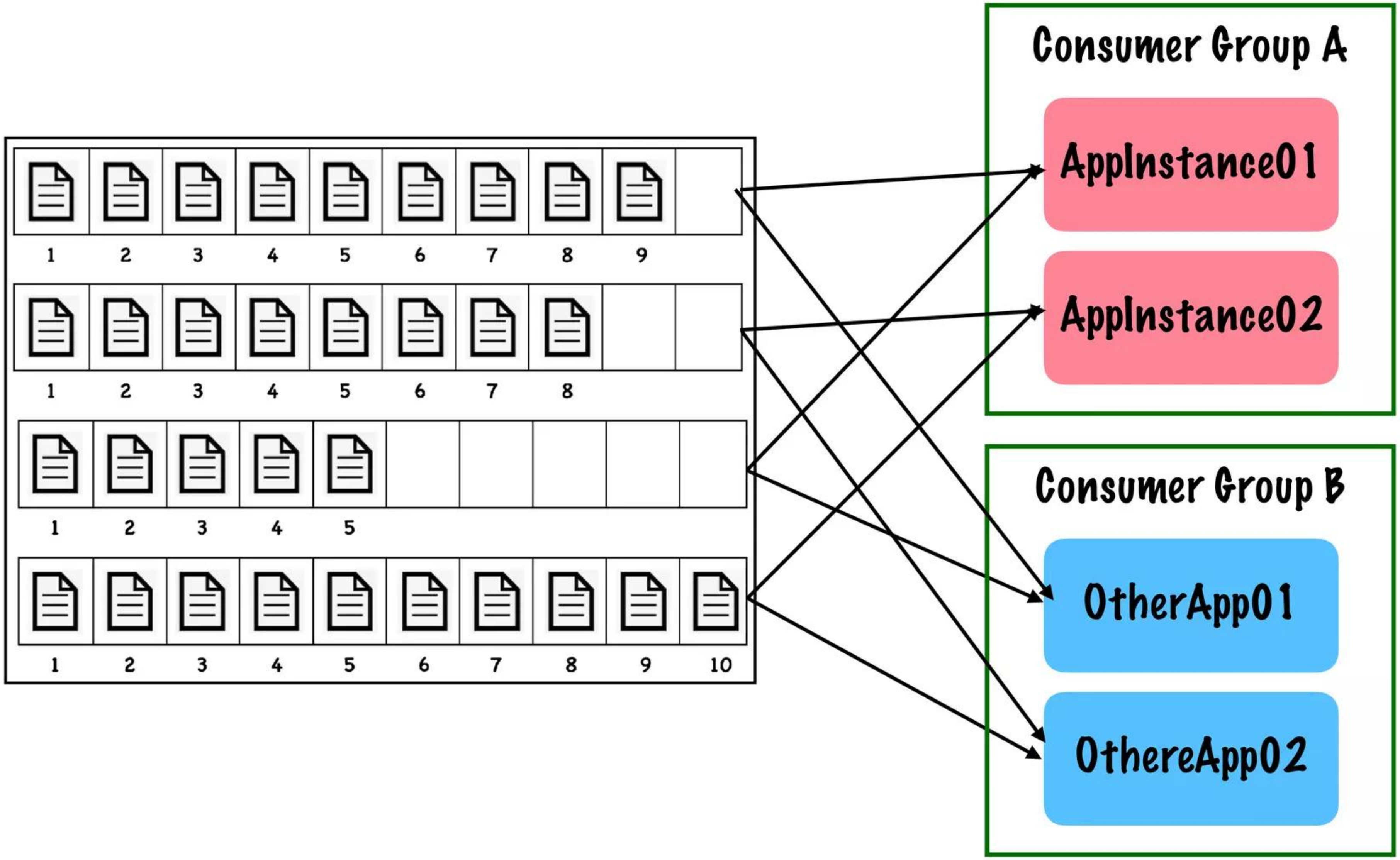
Sample Kafka Consumer

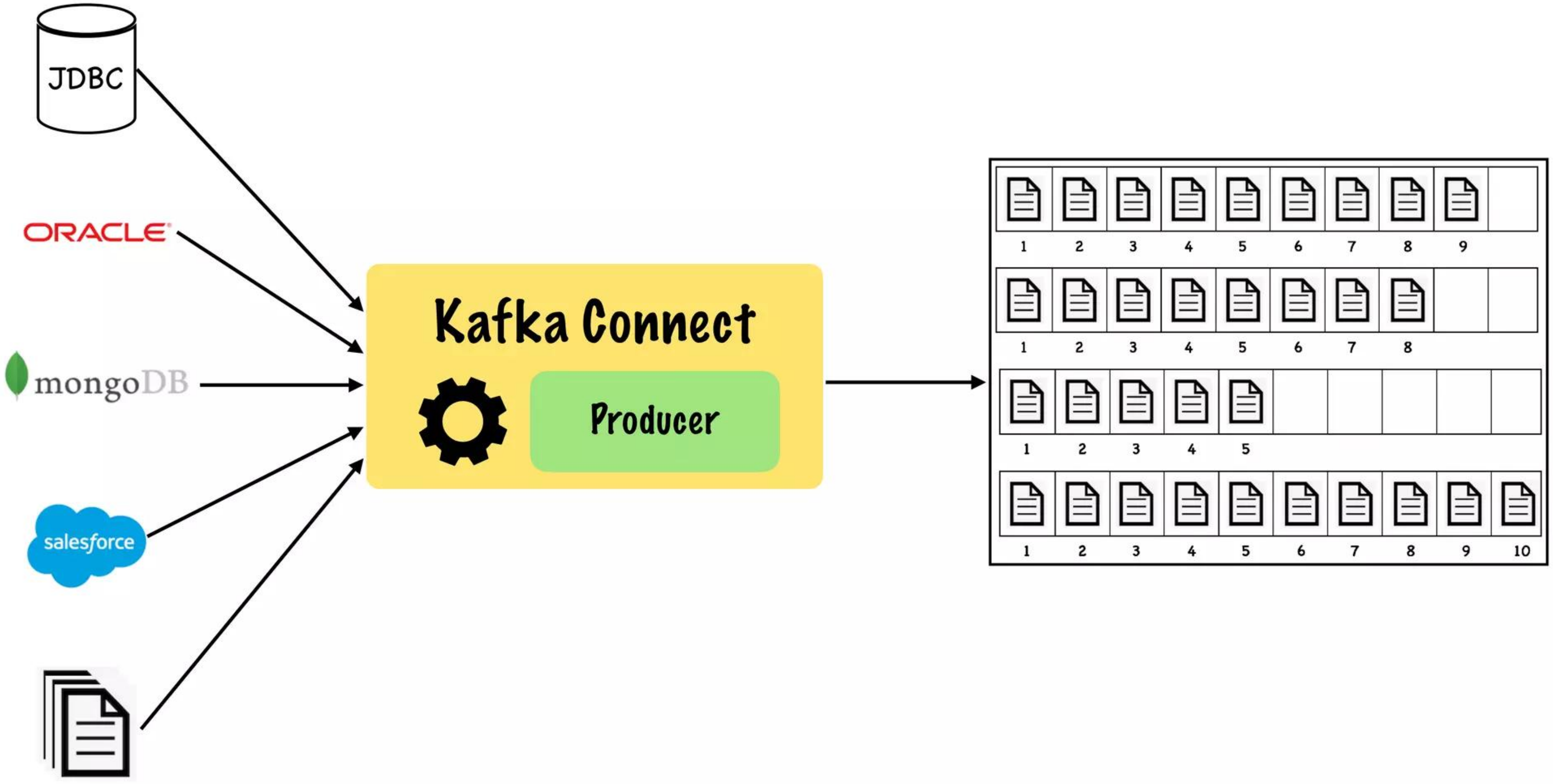
```java
Consumer<String, String> consumer = new KafkaConsumer<String, String>(props);
consumer.subscribe(Arrays.asList(topic));

try {
  while (true) {
    ConsumerRecords<String, String> records = consumer.poll(100);
    for (ConsumerRecord<String, String> record : records) {
      String key = record.key();
      String value = record.value();
      System.out.printf("Consumed record - key: %s and value: %s", key, value);
    }
  }
} finally {
  consumer.close();
}
```

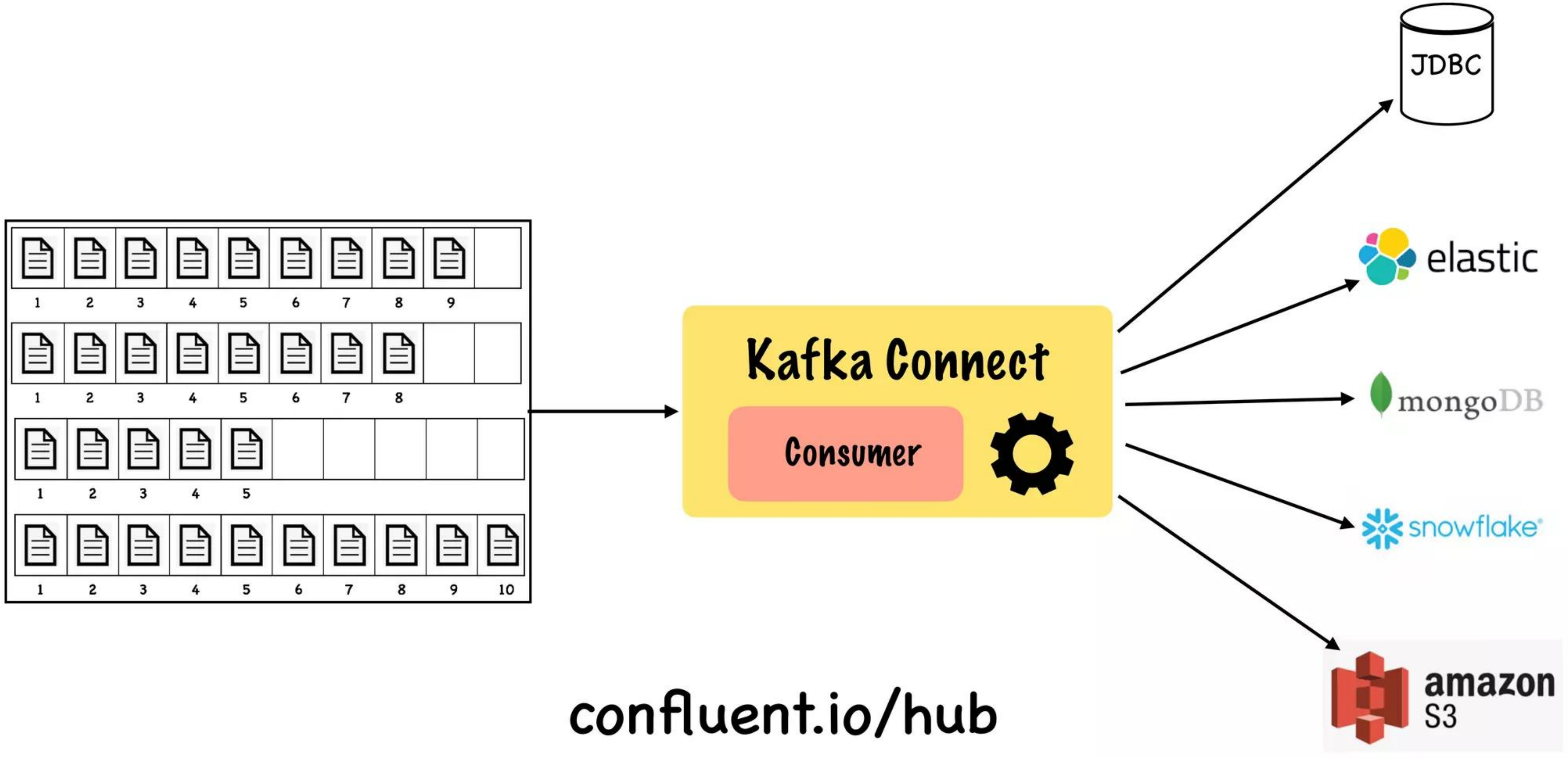

Producer
Schema Registry
1
2
3
4
5
6
7
8
9
10
Consumer


Consumer Group A
AppInstance01
AppInstance02
1 2 3 4 5 6 7 8 9
1 2 3 4 5 6 7 8
1 2 3 4 5
1 2 3 4 5 6 7 8 9 10

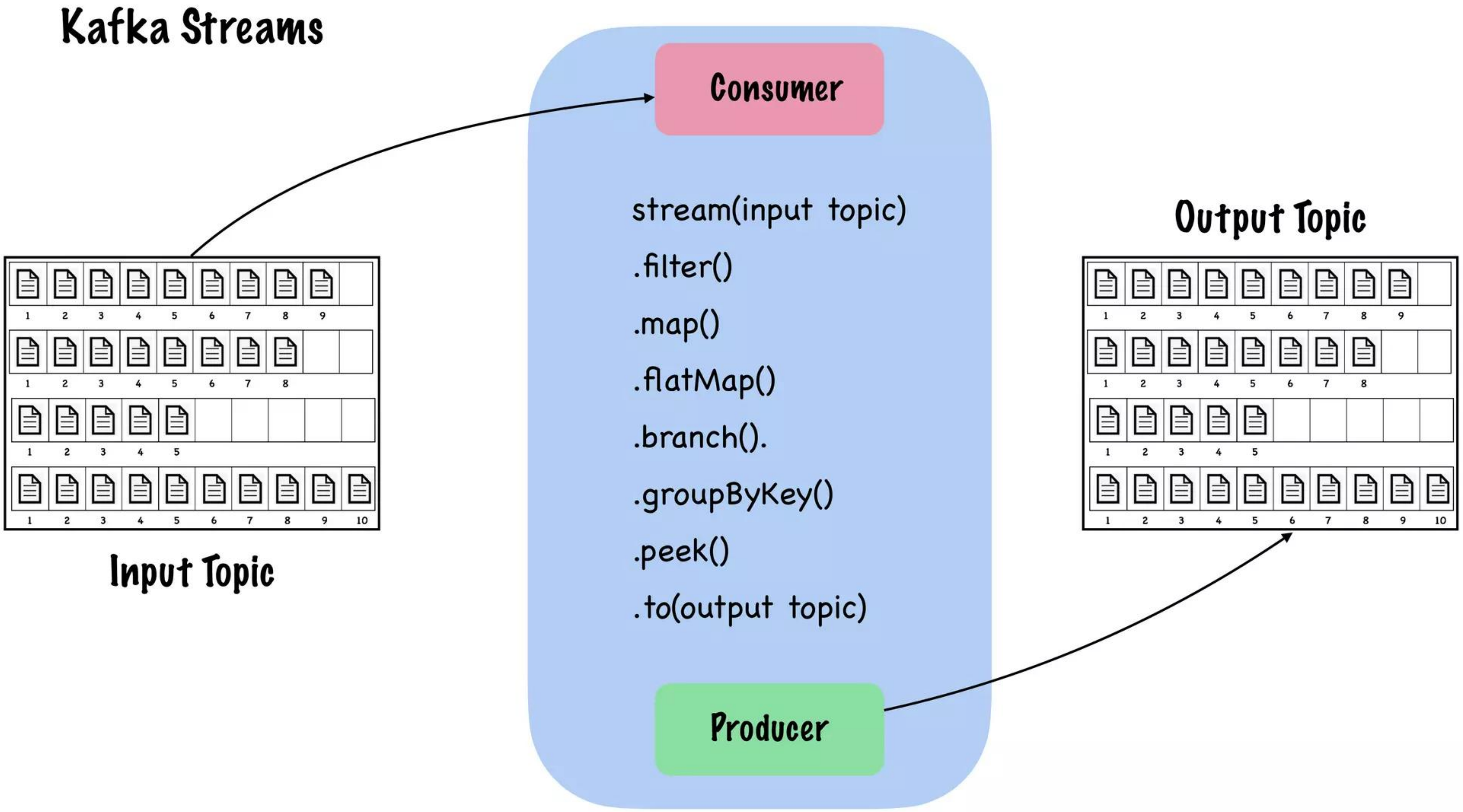

1 2 3 4 5 6 7 8 9
1 2 3 4 5 6 7 8
1 2 3 4 5
1 2 3 4 5 6 7 8 9 10
Consumer Group A
AppInstance01
AppInstance02
AppInstance03
AppInstance04


Consumer Group A
AppInstance01
AppInstance02
Consumer Group B
OtherApp01
OthereApp02
1 2 3 4 5 6 7 8 9
1 2 3 4 5 6 7 8
1 2 3 4 5
1 2 3 4 5 6 7 8 9 10

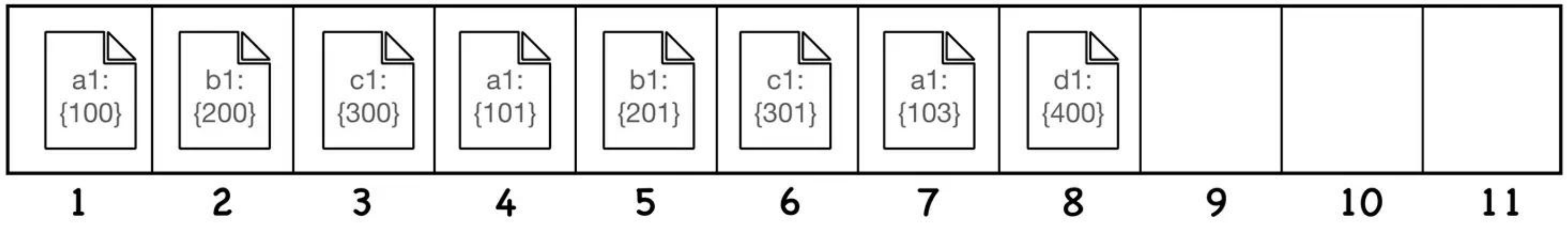

JDBC
ORACLE
mongoDB
salesforce
Kafka Connect
Producer
1 2 3 4 5 6 7 8 9
1 2 3 4 5 6 7 8
1 2 3 4 5
1 2 3 4 5 6 7 8 9 10



confluent.io/hub


Kafka Streams
Consumer
stream(input topic)
.filter()
.map()
.flatMap()
.branch().
.groupByKey()
.peek()
.to(output topic)
Producer
Input Topic
Output Topic
1 2 3 4 5 6 7 8 9
1 2 3 4 5 6 7 8
1 2 3 4 5
1 2 3 4 5 6 7 8 9 10

Stream



Table

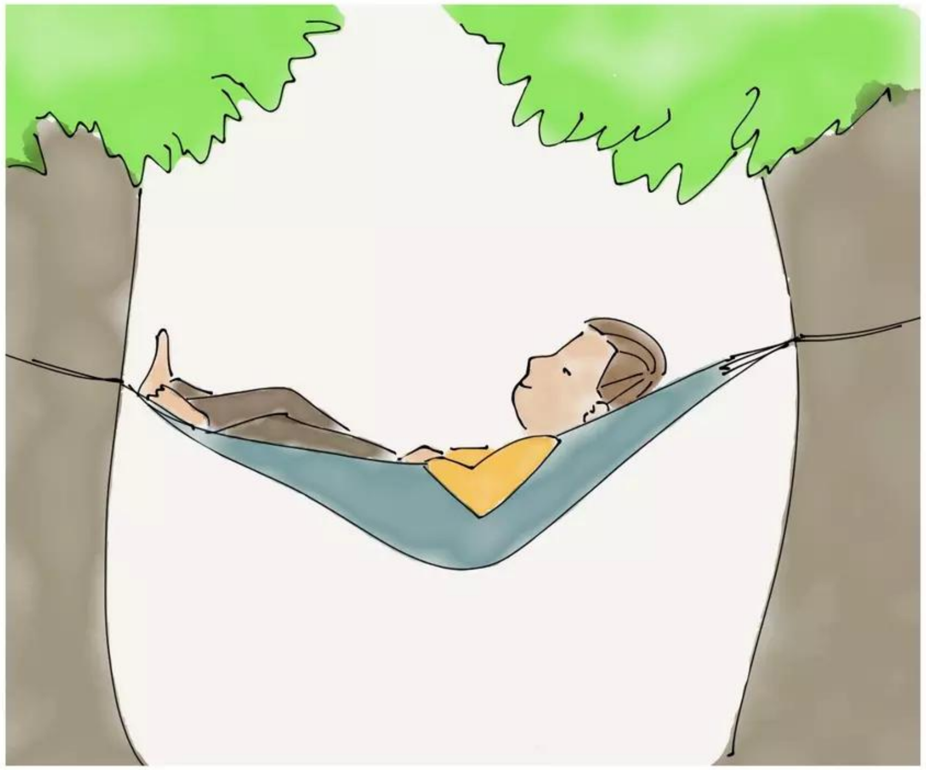
a1	{103}
b1	{201}
c1	{301}
d1	{400}

ksqlDB

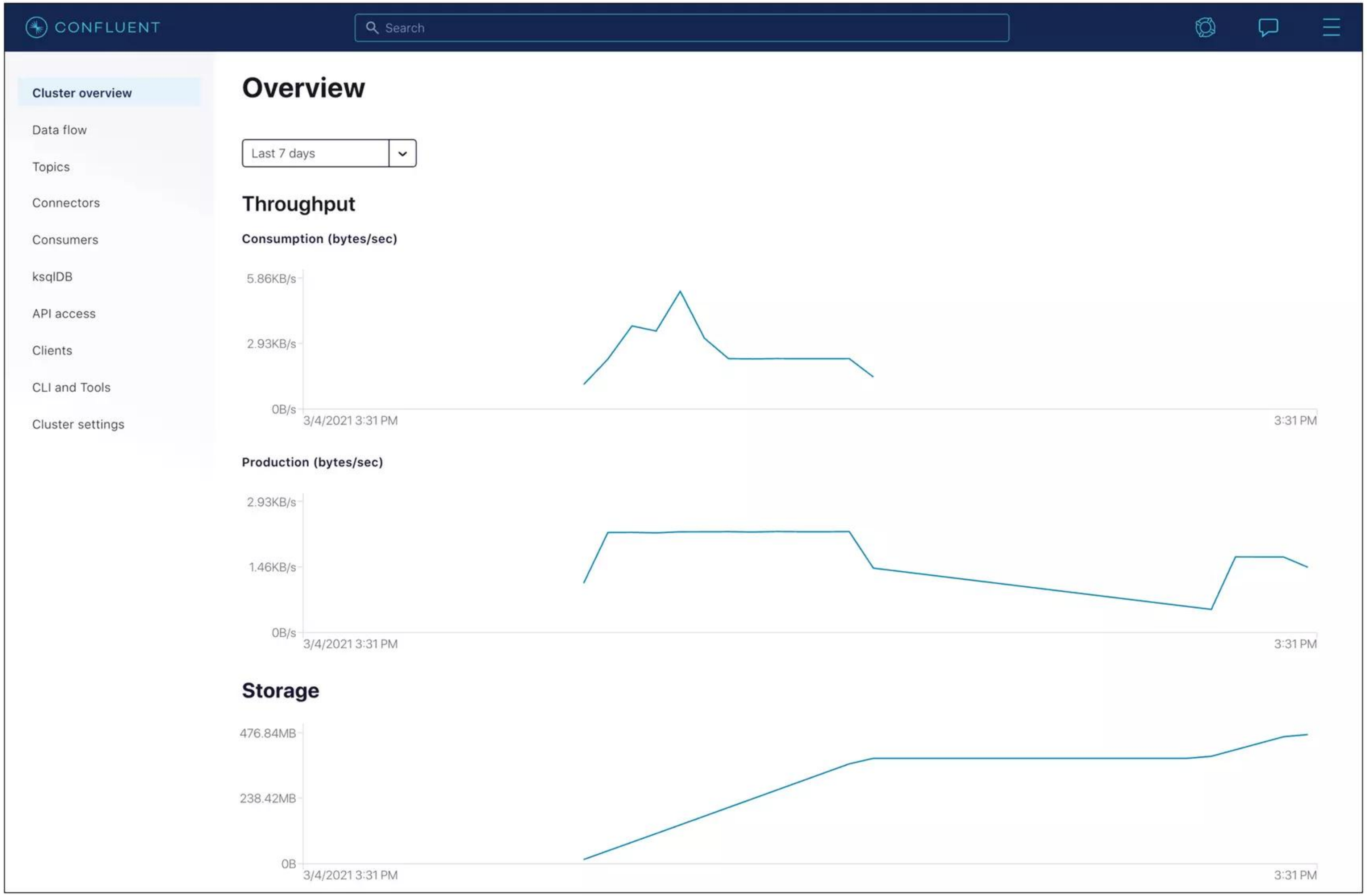
```
ksql> CREATE STREAM books (
    title VARCHAR, author VARCHAR, year_published
INT
 ) WITH (
    KAFKA_TOPIC=‘booklist’, VALUE_FORMAT=‘AVRO’
 );
```

```
ksql> SELECT * FROM books WHERE title LIKE ‘%kafka%’ EMIT CHANGES;
```

```
+---------------------------------+------------------------------+-------------+
|TITLE                            |AUTHOR                        |YEAR         |
+---------------------------------+------------------------------+-------------+
|Kafka: The Definitive Guide      |Shapira, Narkhede, Palino     |2017         |
|Kafka Streams in Action          |Bill Bejeck                   |2018         |
|Kafka in Action                  |Dylan                         |2020         |
|Apache Kafka Cookbook            |Raul Estrada                  |2017         |
```



```
curl -X POST -H "Content-Type: application/vnd.kafka.json.v2+json" \
     -H "Accept: application/vnd.kafka.v2+json" \
     --data '{"records":[{"value":{"foo":"bar"}}]}' \
     "http://localhost:8082/topics/jsontest"
```



Overview

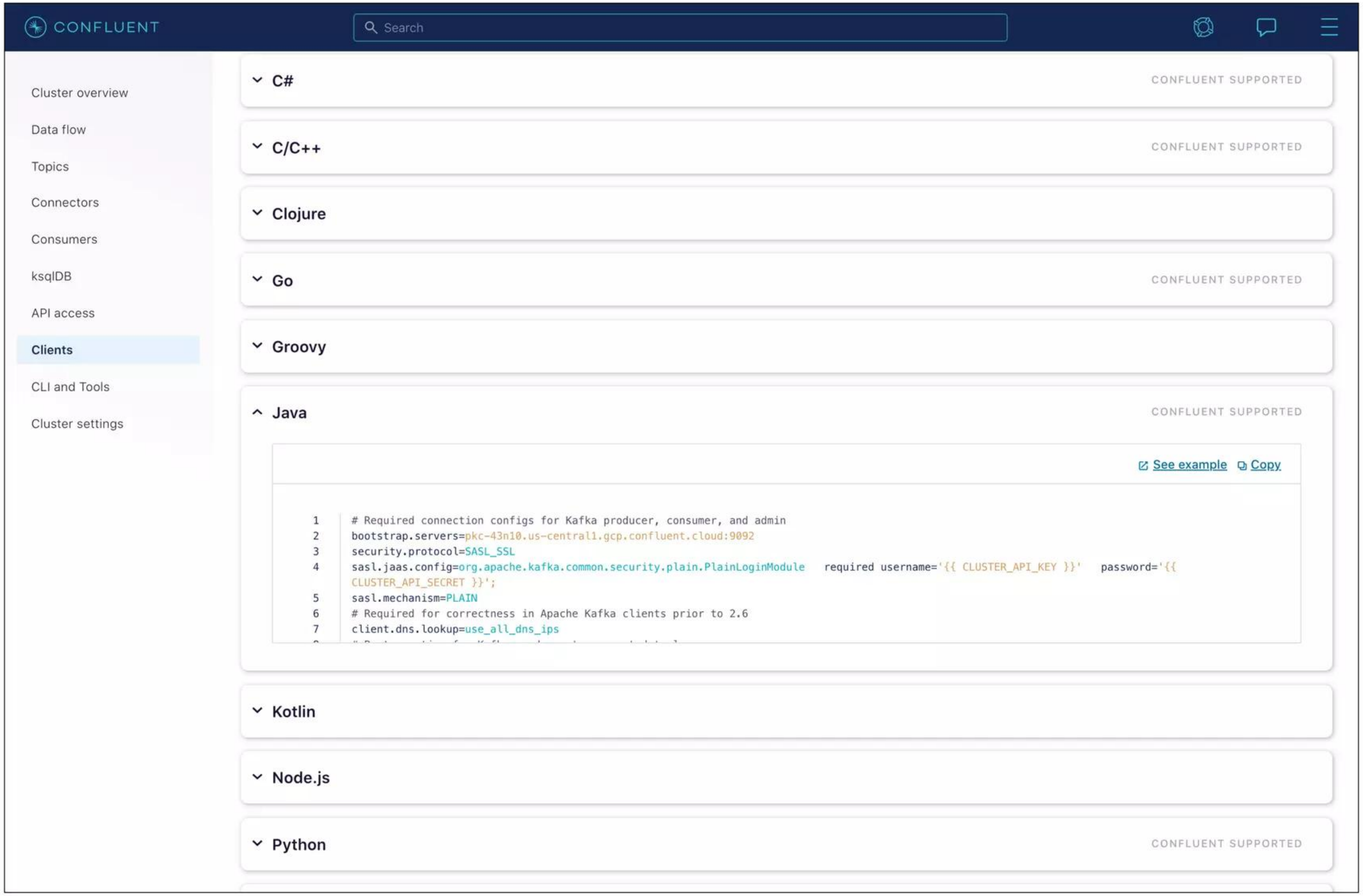
Last 7 days

Throughput

Consumption (bytes/sec)

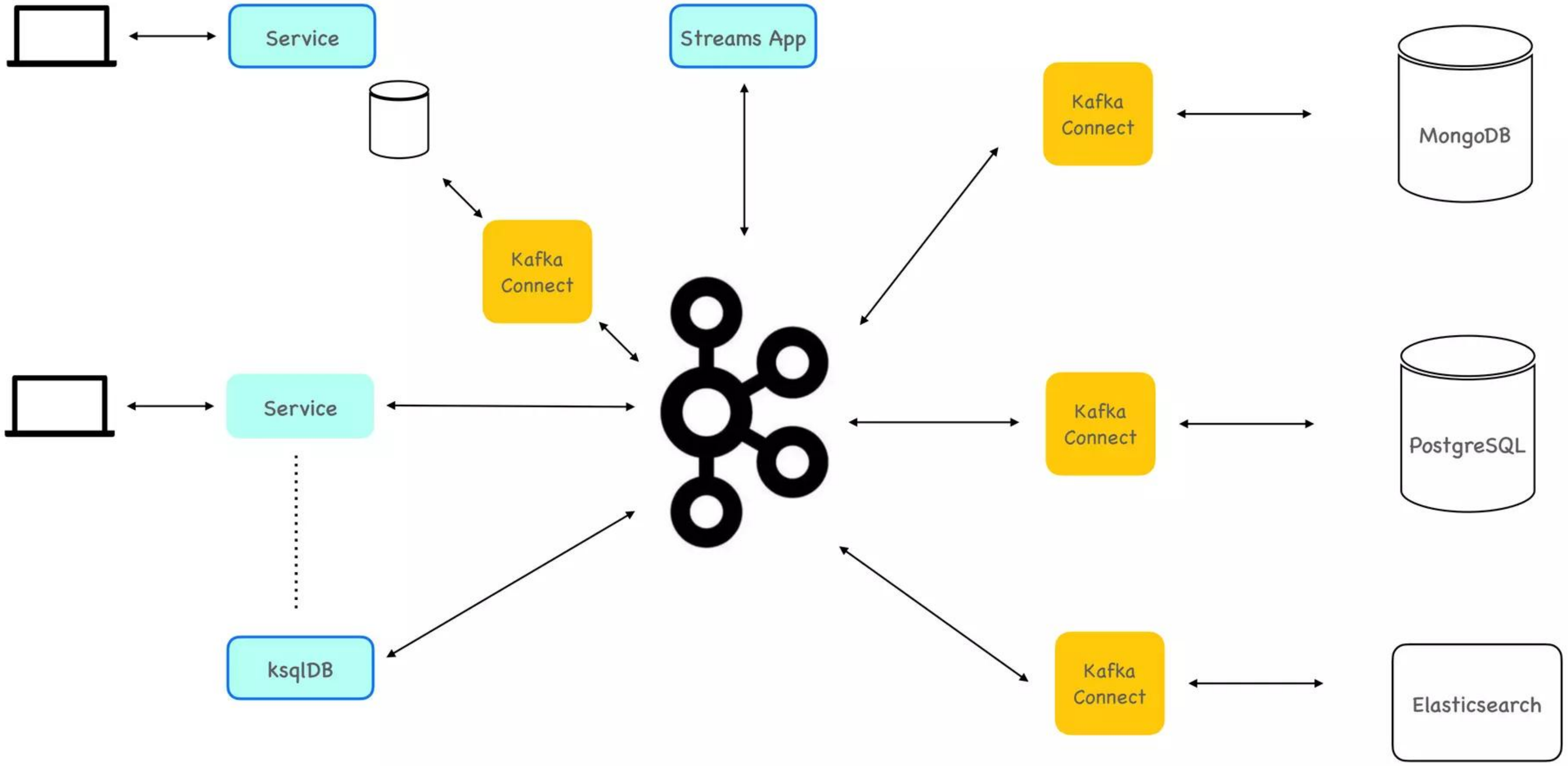
Production (bytes/sec)

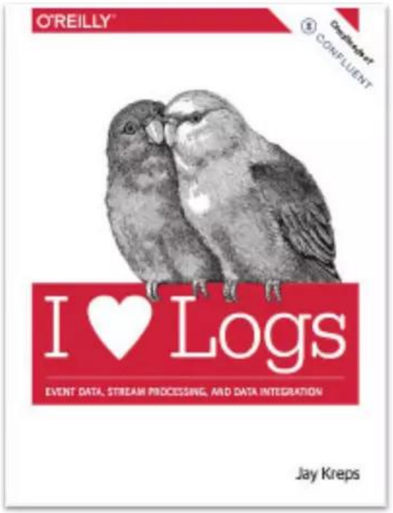
Storage



```
# Required connection configs for Kafka producer, consumer, and admin
bootstrap.servers=pkc-43n10.us-central1.gcp.confluent.cloud:9092
security.protocol=SASL_SSL
sasl.jaas.config=org.apache.kafka.common.security.plain.PlainLoginModule   required username='{{ CLUSTER_API_KEY }}'   password='{{
CLUSTER_API_SECRET }}';
sasl.mechanism=PLAIN
# Required for correctness in Apache Kafka clients prior to 2.6
client.dns.lookup=use_all_dns_ips
```

Event Streaming Platform



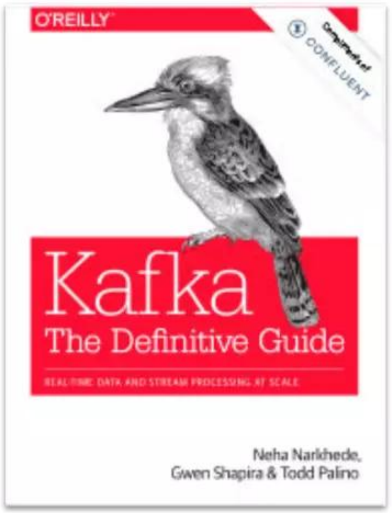

O'REILLY

Compliments of CONFLUENT

I ♥ Logs

EVENT DATA, STREAM PROCESSING, AND DATA INTEGRATION

Jay Kreps

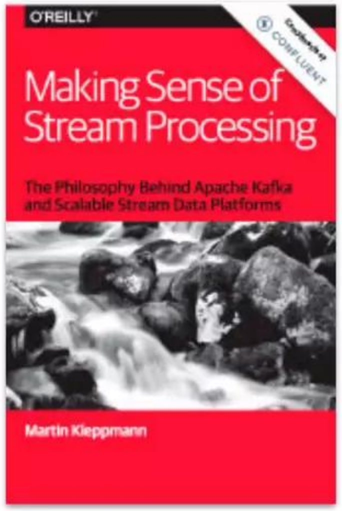

O'REILLY

Compliments of CONFLUENT

Kafka
The Definitive Guide

REAL-TIME DATA AND STREAM PROCESSING AT SCALE

Neha Narkhede,
Gwen Shapira & Todd Palino

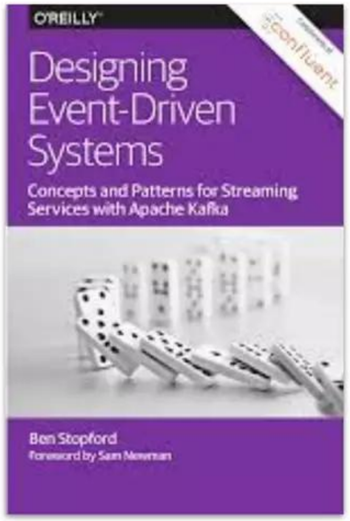

O'REILLY

Compliments of CONFLUENT

Making Sense of
Stream Processing

The Philosophy Behind Apache Kafka
and Scalable Stream Data Platforms

Martin Kleppmann

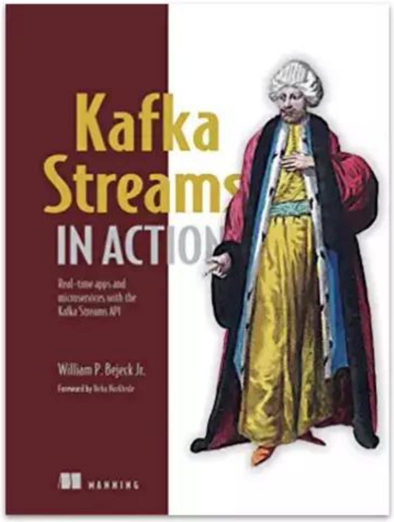

O'REILLY

Compliments of confluent

Designing
Event-Driven
Systems

Concepts and Patterns for Streaming
Services with Apache Kafka

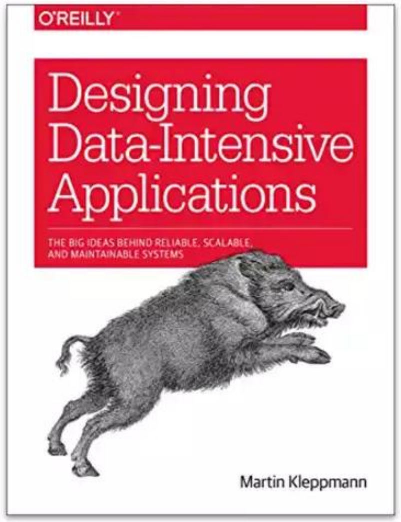
Ben Stopford
Foreword by Sam Newman


Kafka
Streams
IN ACTION

Real-time apps and
microservices with the
Kafka Streams API

William P. Bejeck Jr.
Foreword by Neha Narkhede

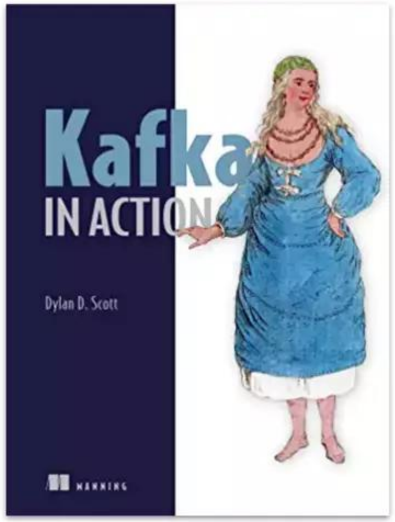
MANNING


O'REILLY

Designing
Data-Intensive
Applications

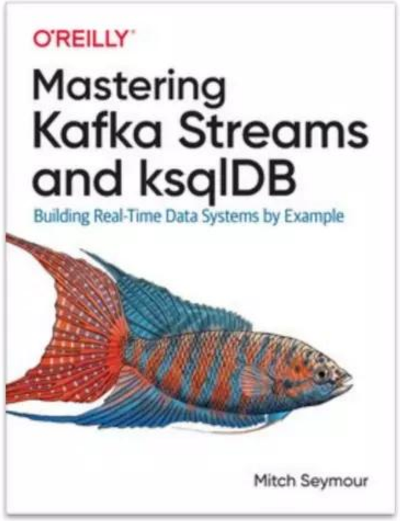
THE BIG IDEAS BEHIND RELIABLE, SCALABLE,
AND MAINTAINABLE SYSTEMS

Martin Kleppmann


Kafka
IN ACTION

Dylan D. Scott

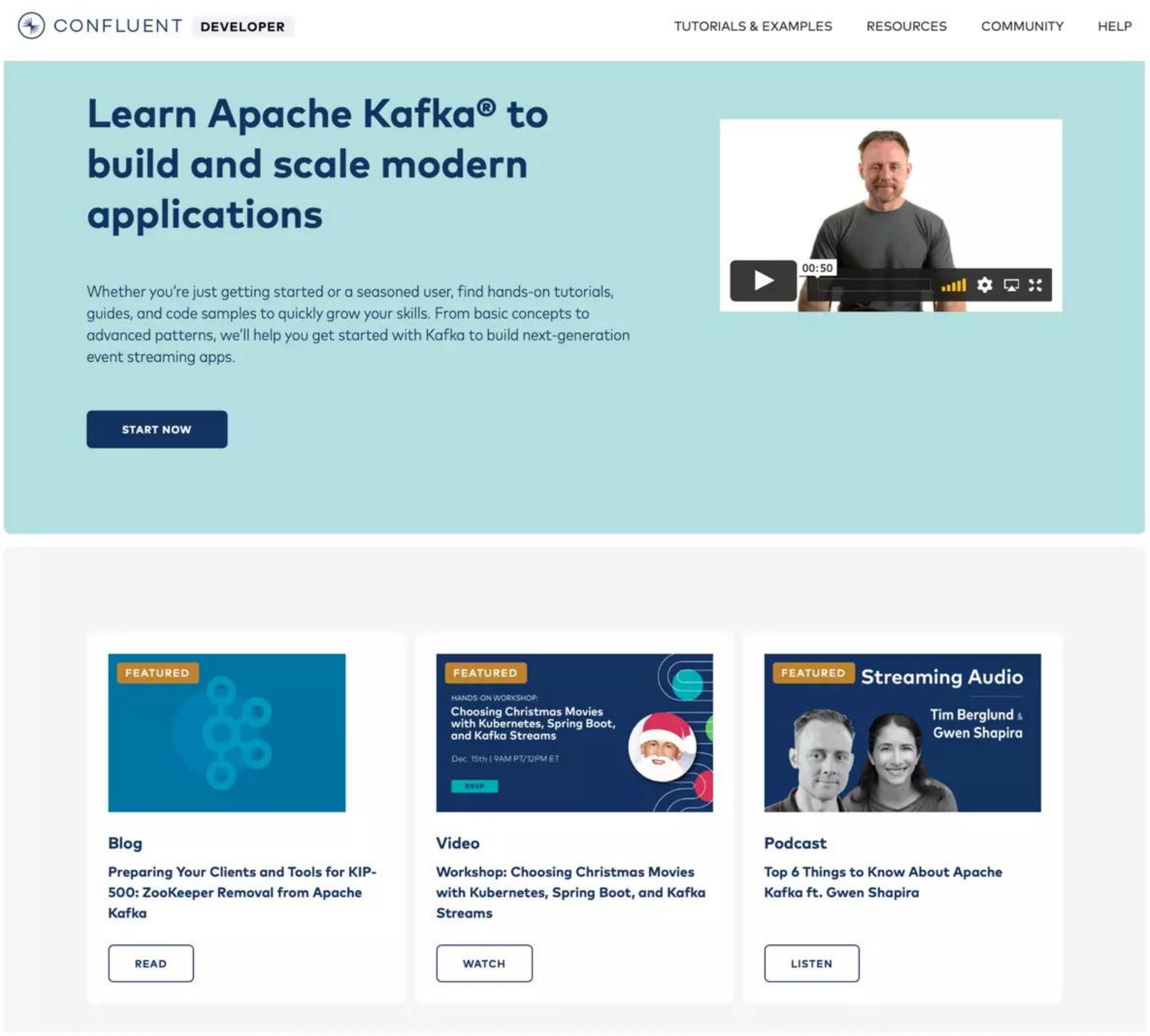
MANNING


O'REILLY

Mastering
Kafka Streams
and ksqlDB

Building Real-Time Data Systems by Example

Mitch Seymour


CONFLUENT DEVELOPER

TUTORIALS & EXAMPLES RESOURCES COMMUNITY HELP

Learn Apache Kafka® to build and scale modern applications

Whether you're just getting started or a seasoned user, find hands-on tutorials, guides, and code samples to quickly grow your skills. From basic concepts to advanced patterns, we'll help you get started with Kafka to build next-generation event streaming apps.

START NOW

Blog

Preparing Your Clients and Tools for KIP-500: ZooKeeper Removal from Apache Kafka

READ

Video

Workshop: Choosing Christmas Movies with Kubernetes, Spring Boot, and Kafka Streams

WATCH

Podcast

Top 6 Things to Know About Apache Kafka ft. Gwen Shapira

LISTEN

00:50
FEATURED
HANDS-ON WORKSHOP:
Choosing Christmas Movies with Kubernetes, Spring Boot, and Kafka Streams
Dec. 15th | 9AM PT/12PM ET
RSVP
Streaming Audio
Tim Berglund & Gwen Shapira

https://developer.confluent.io

Kafka The Definitive Guide
Making Sense of Stream Processing
I ♥ Logs
Designing Event-Driven Systems

https://cnfl.io/book-bundle

https://cnfl.io/meetup-hub

Community

https://cnfl.io/community

twitter.com/daveklein

dklein@confluent.io