

# Welcome to Kafka

## We're Glad You're Here

Dave Klein

# The Plan

- Overview of the Apache Kafka Ecosystem
- Strategies to go about learning it
- Resources for every learning style
- Friends to help you on your journey

# Event

## Notification

Order Placed

Temperature Read

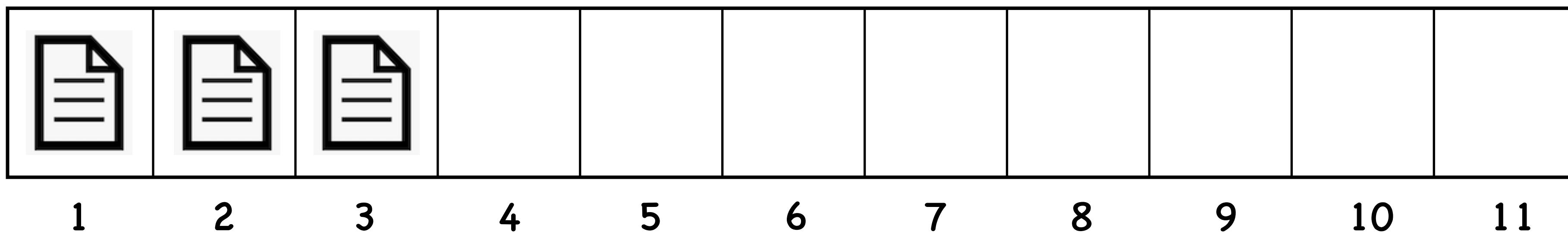
## State

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

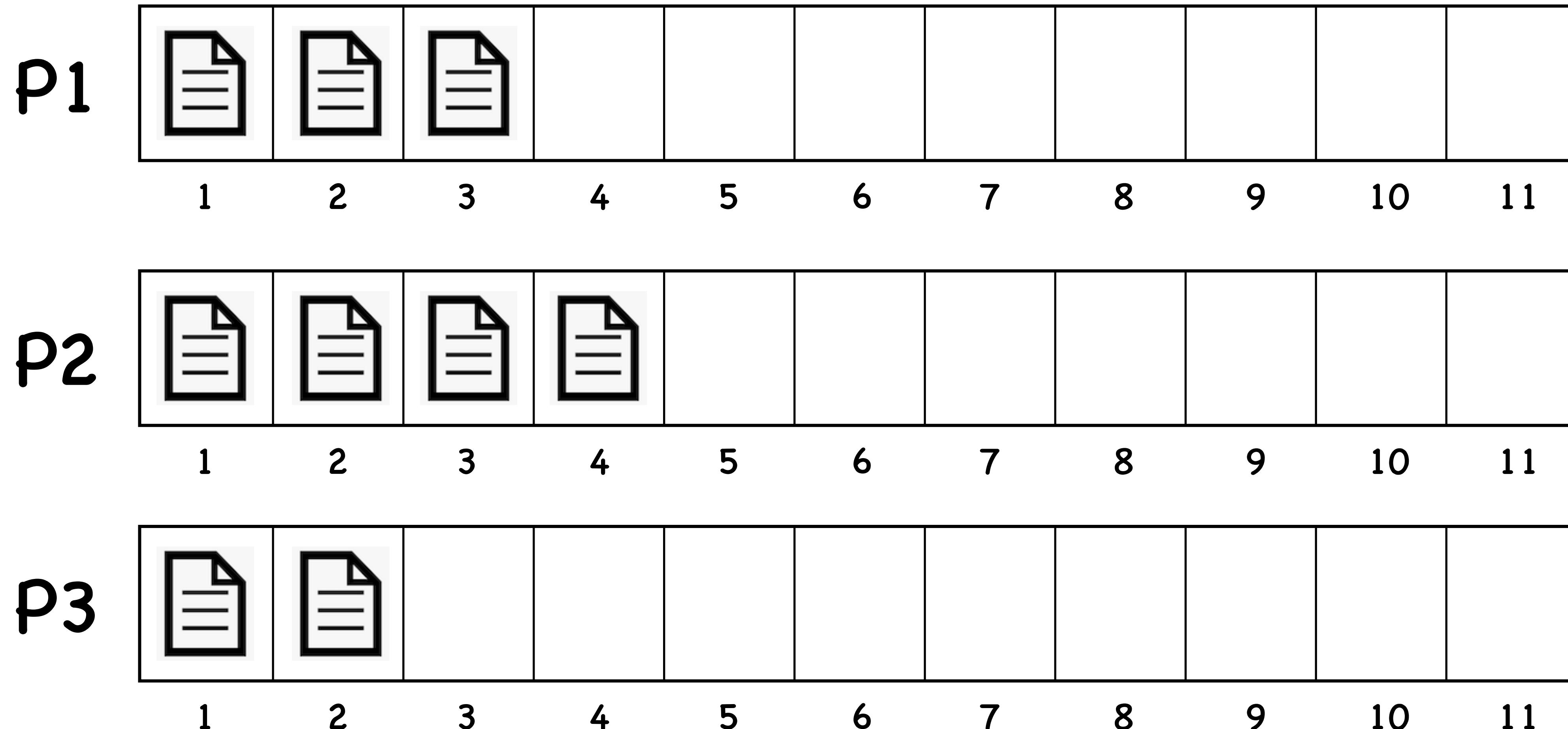
```
{temp:29, unit: F,  
time: 1606949836369}
```

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

# Topic (log)

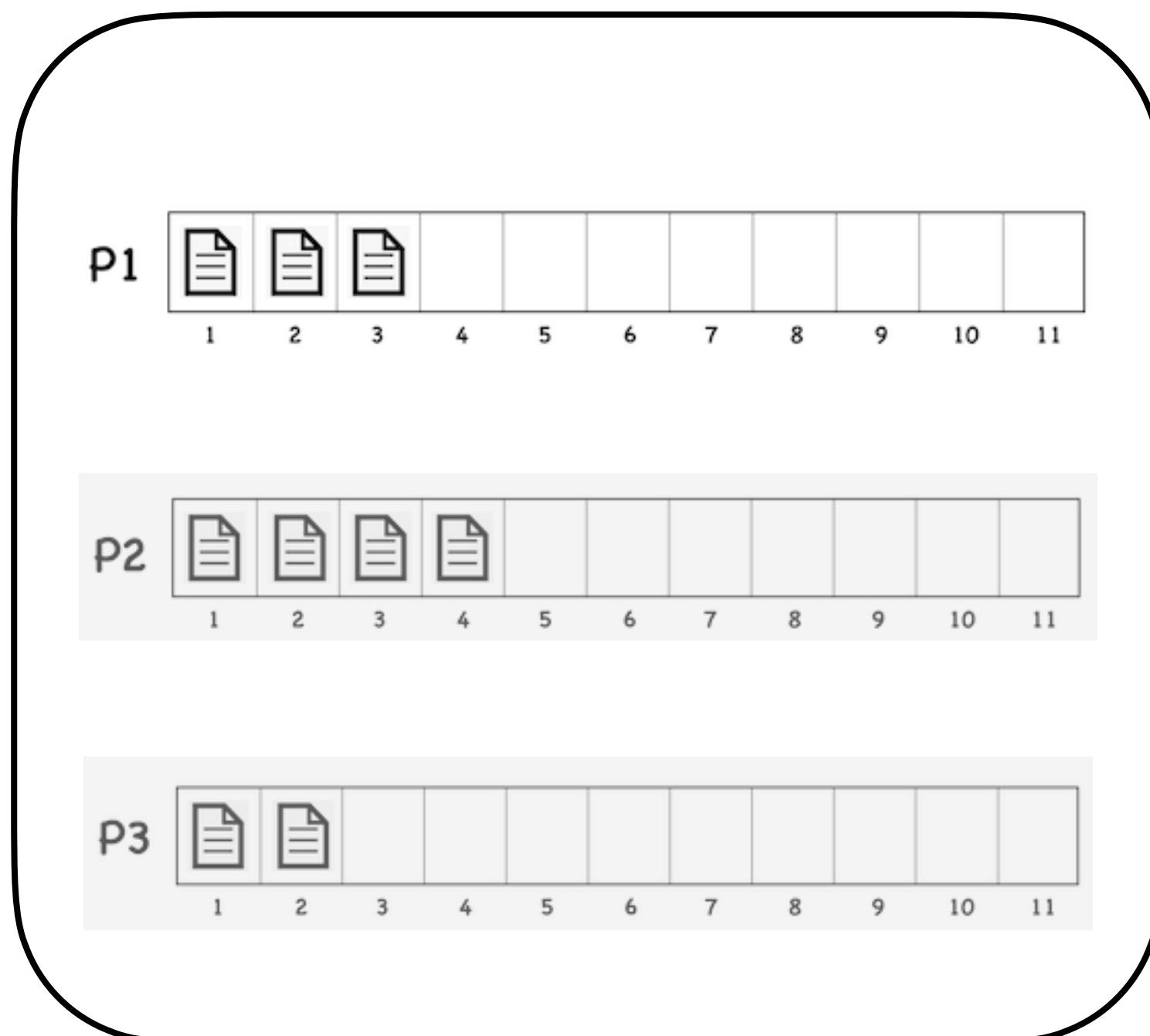


# Topic Partitions

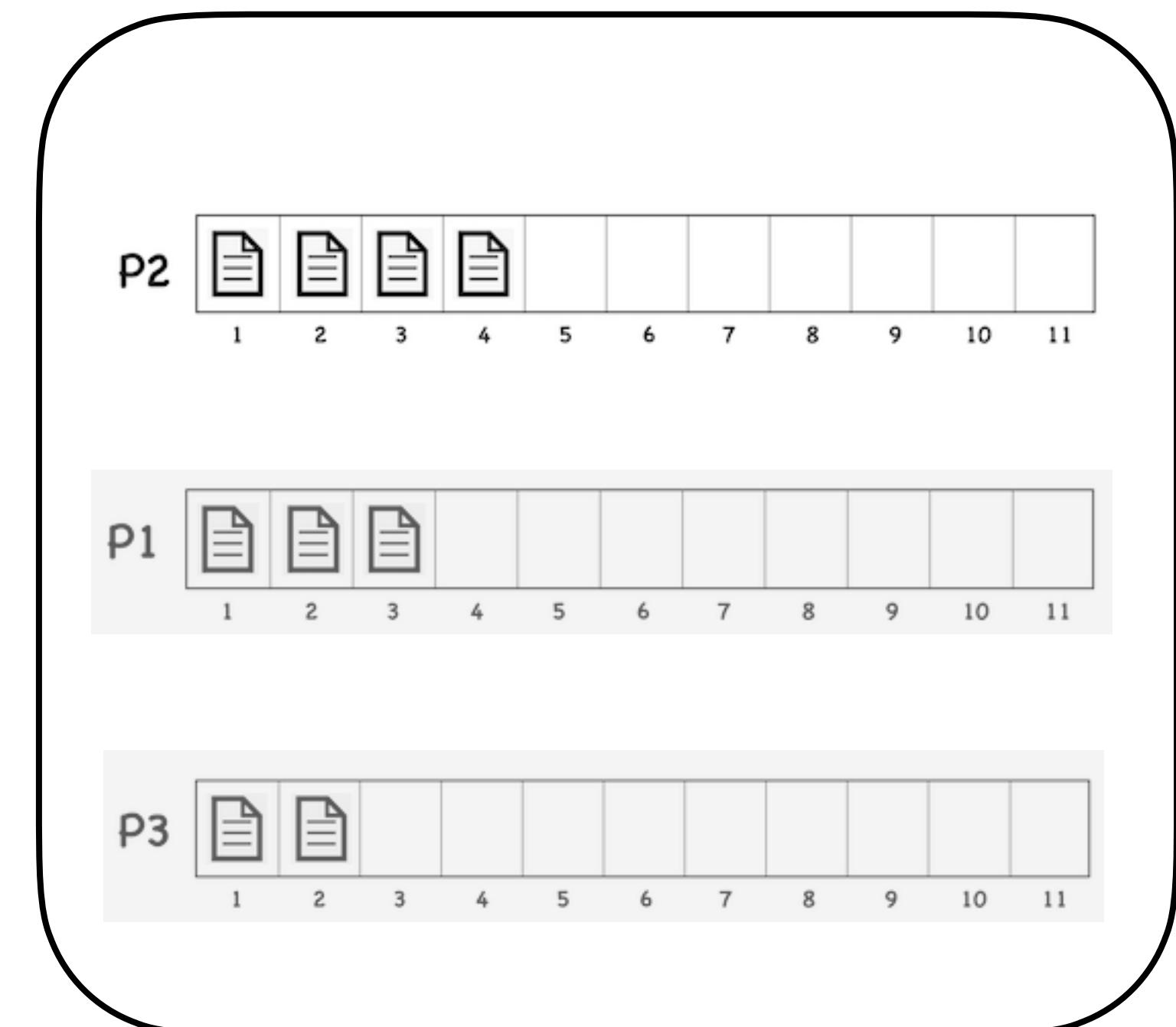


# Brokers and Replication

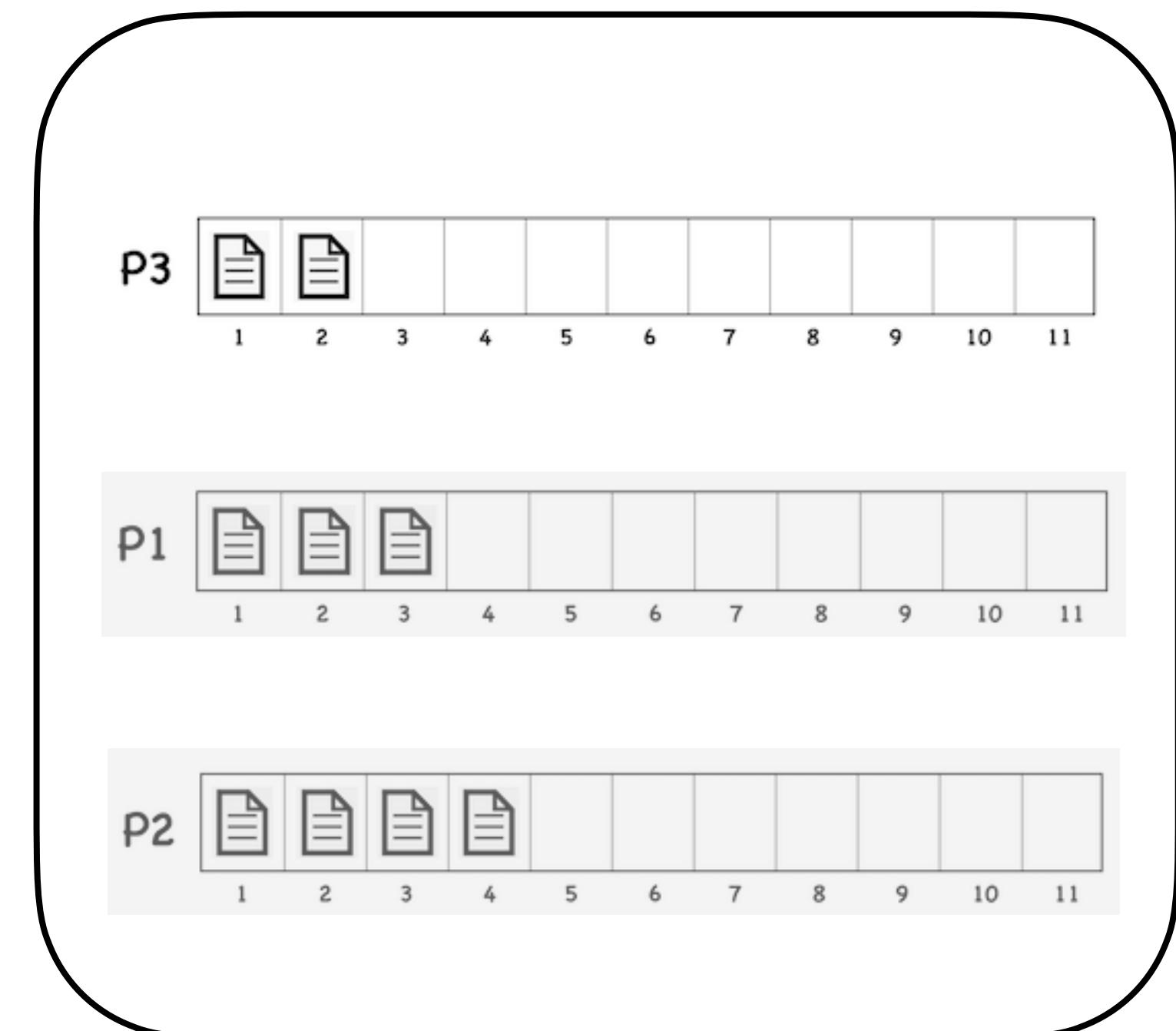
Broker 1



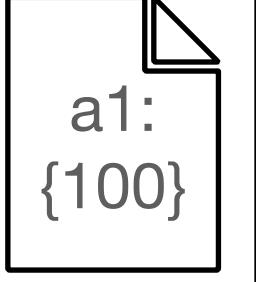
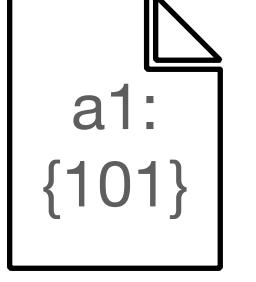
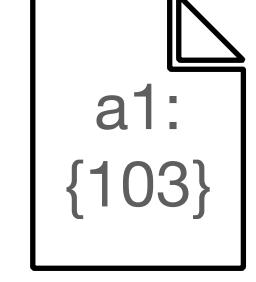
Broker 2



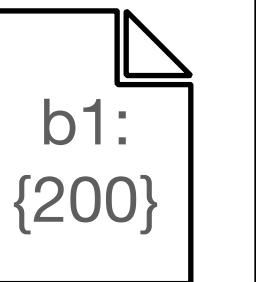
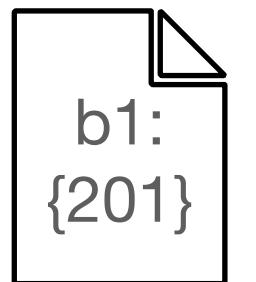
Broker 3



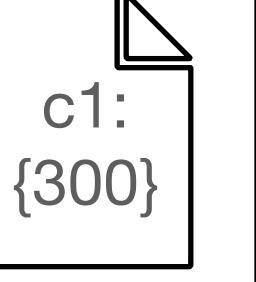
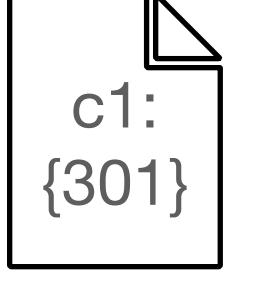
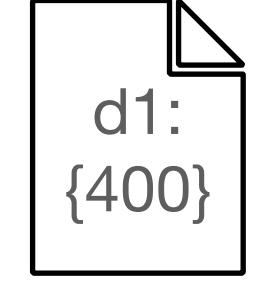
# Topic

 a1: {100}	 a1: {101}	 a1: {103}								
---	--	--	--	--	--	--	--	--	--	--

1 2 3 4 5 6 7 8 9 10 11

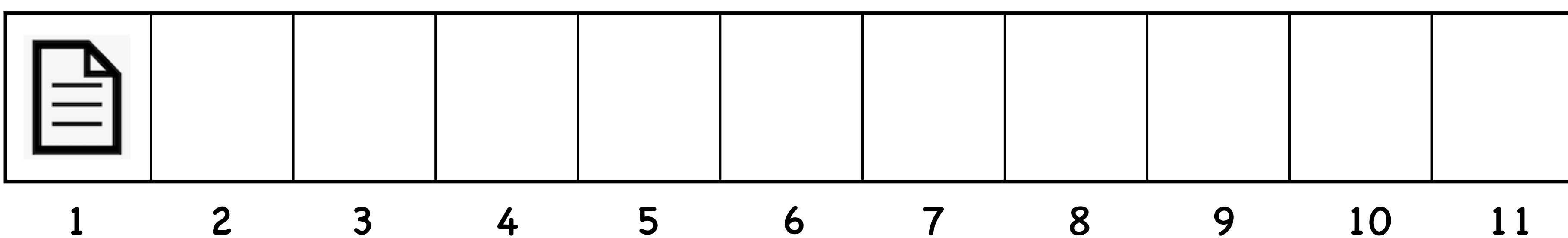
 b1: {200}	 b1: {201}									
---	--	--	--	--	--	--	--	--	--	--

1 2 3 4 5 6 7 8 9 10 11

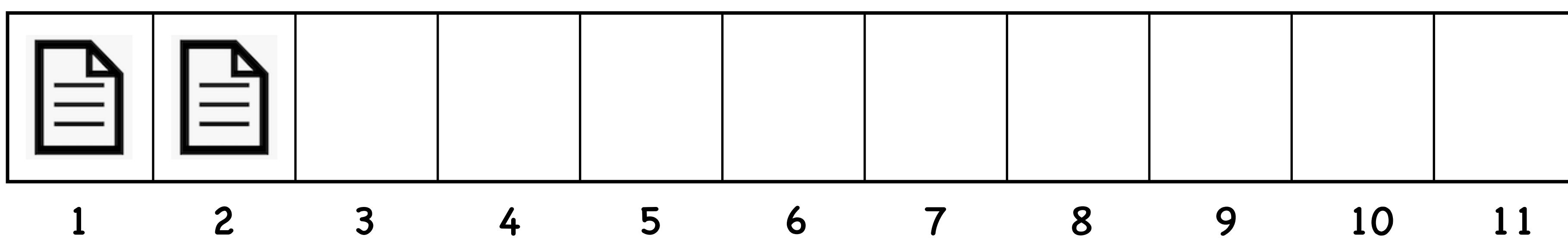
 c1: {300}	 c1: {301}	 d1: {400}	 d1: {401}							
---	--	--	--	--	--	--	--	--	--	--

1 2 3 4 5 6 7 8 9 10 11

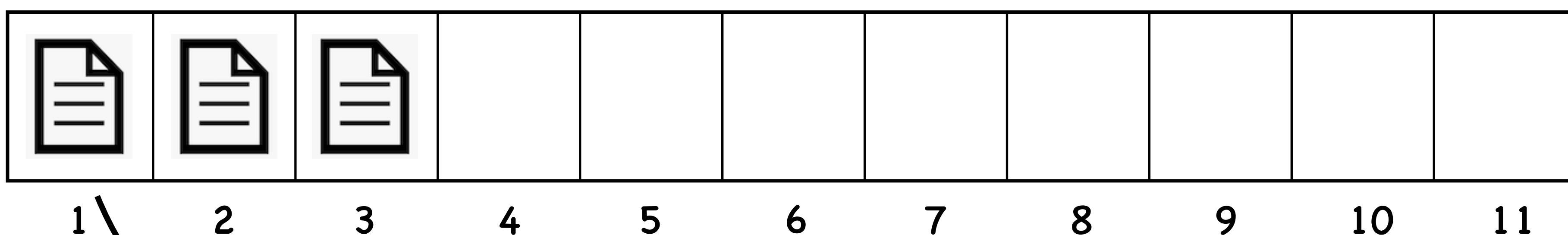
Producer



Producer



**Producer**



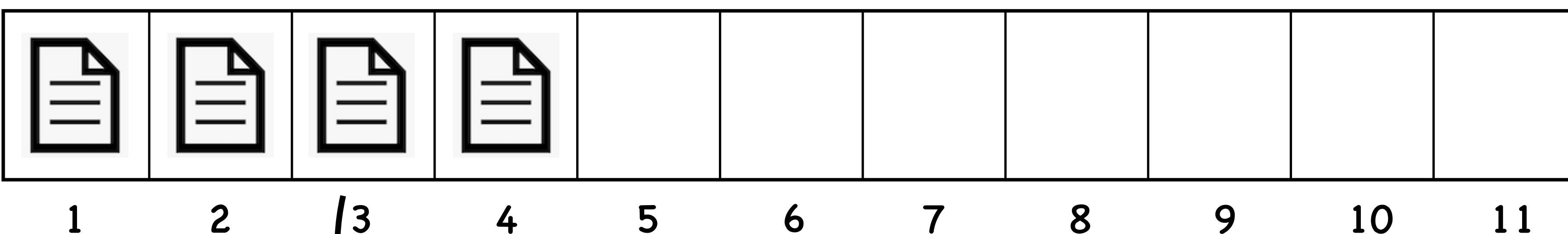
**Consumer**

Producer



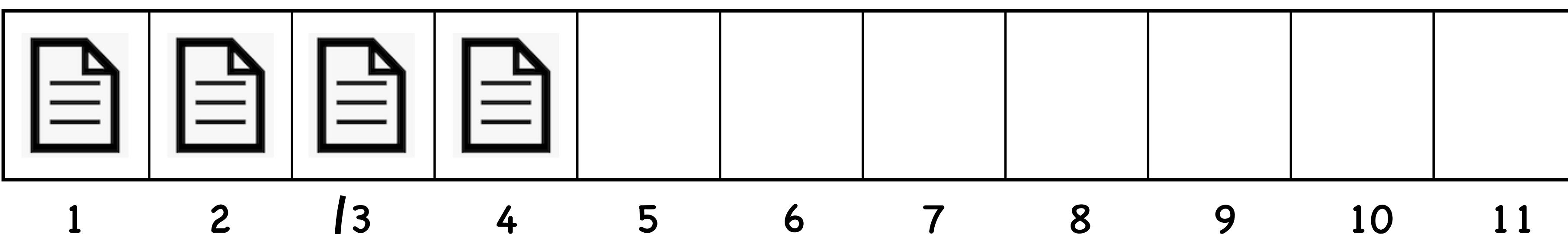
Consumer

Producer



Consumer

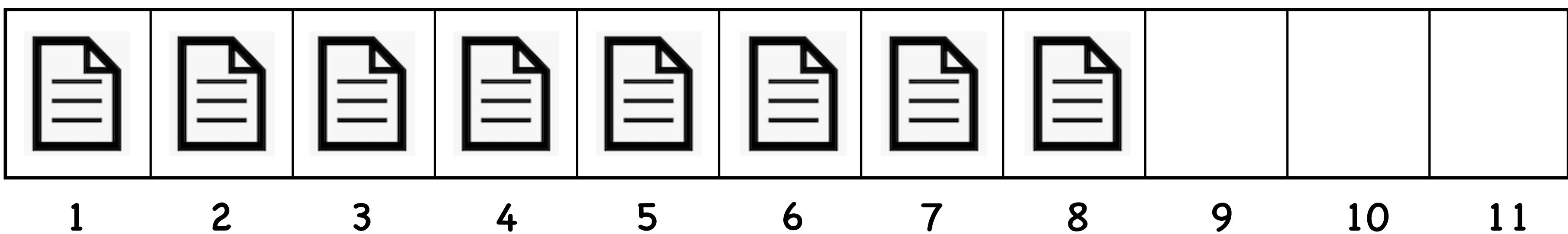
**Producer**



**Consumer**

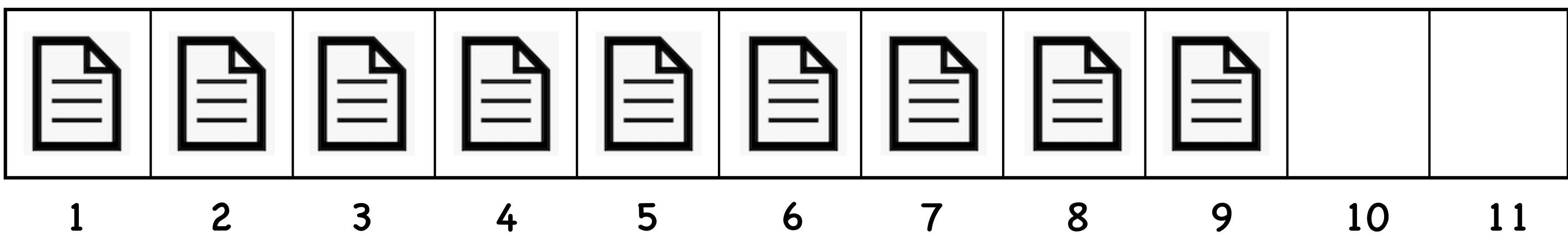
Committed Offset: 3

**Producer**



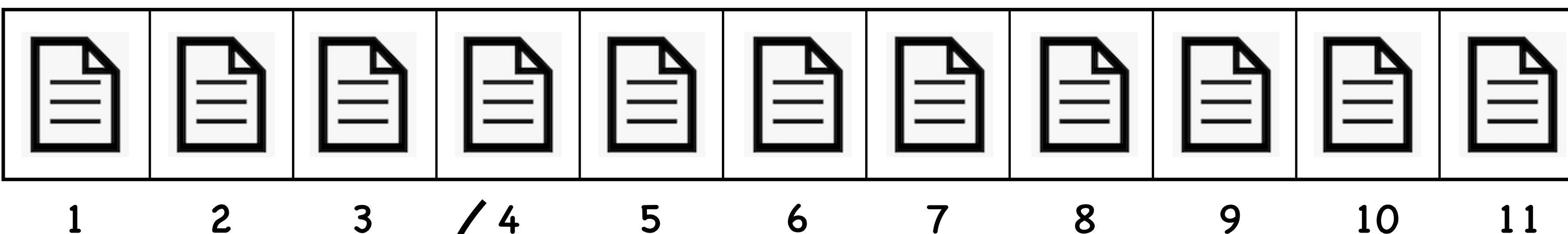
**Consumer**  
Committed Offset: 3

**Producer**



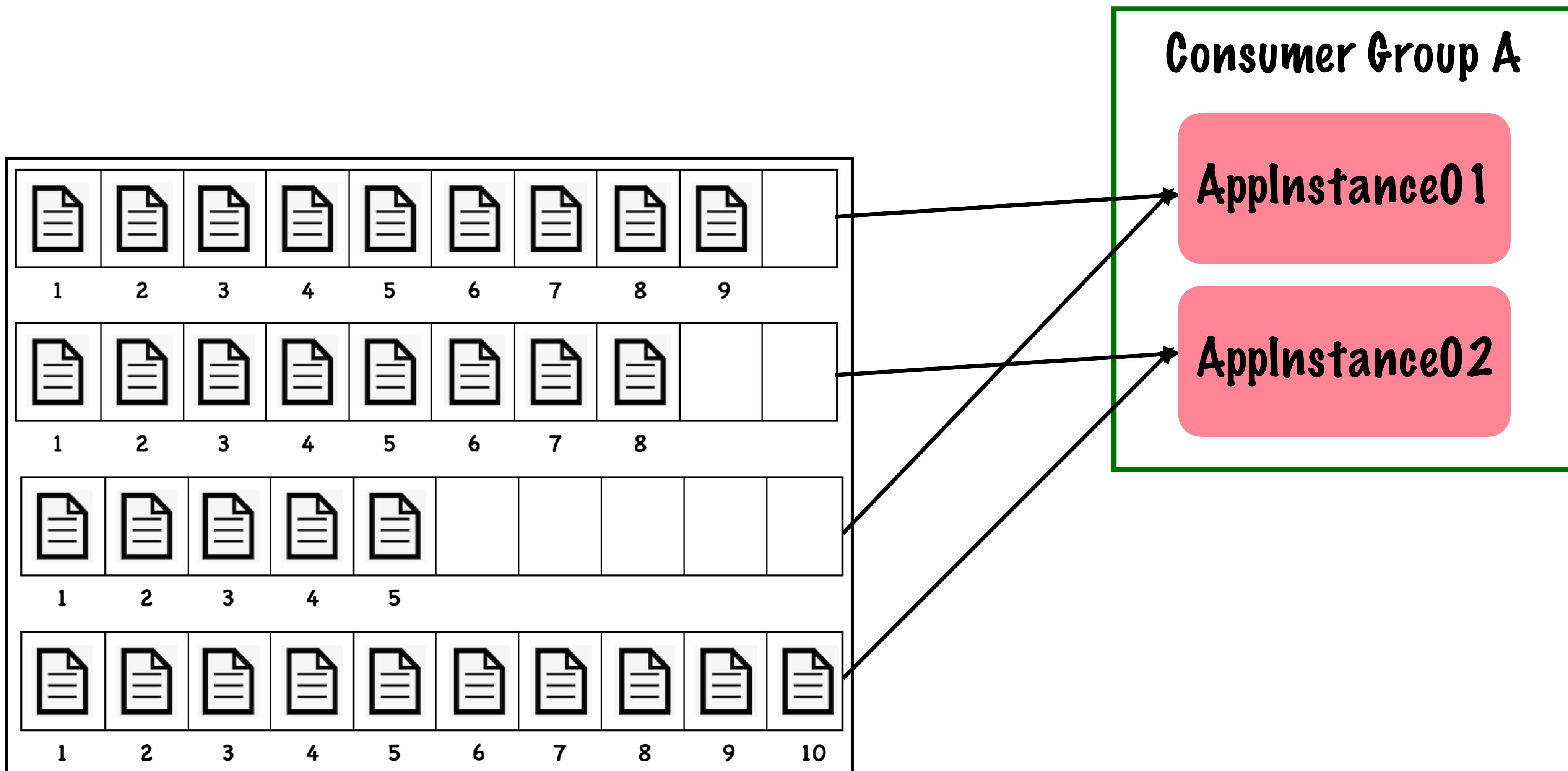
**Consumer**  
Committed Offset: 3

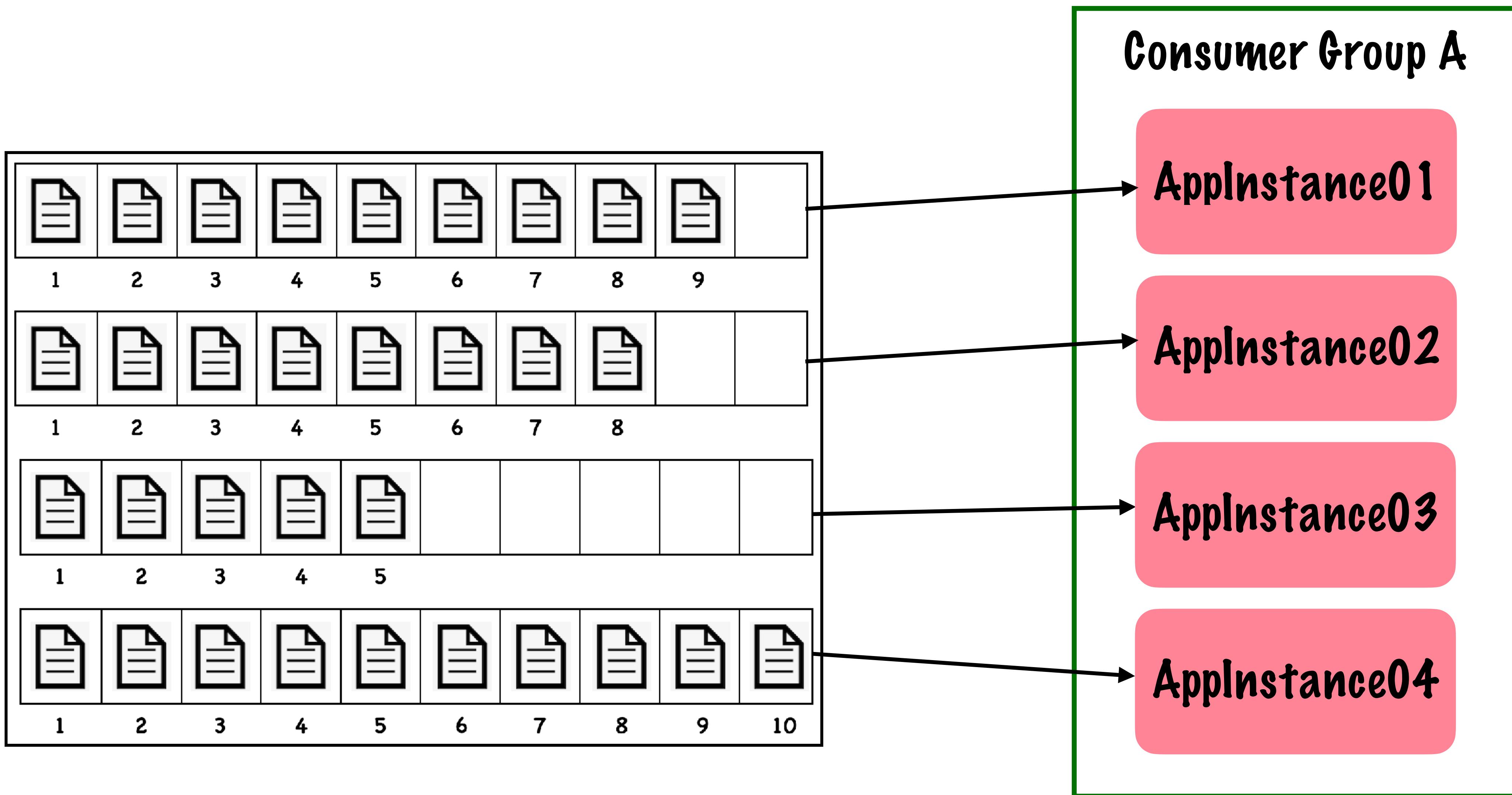
**Producer**

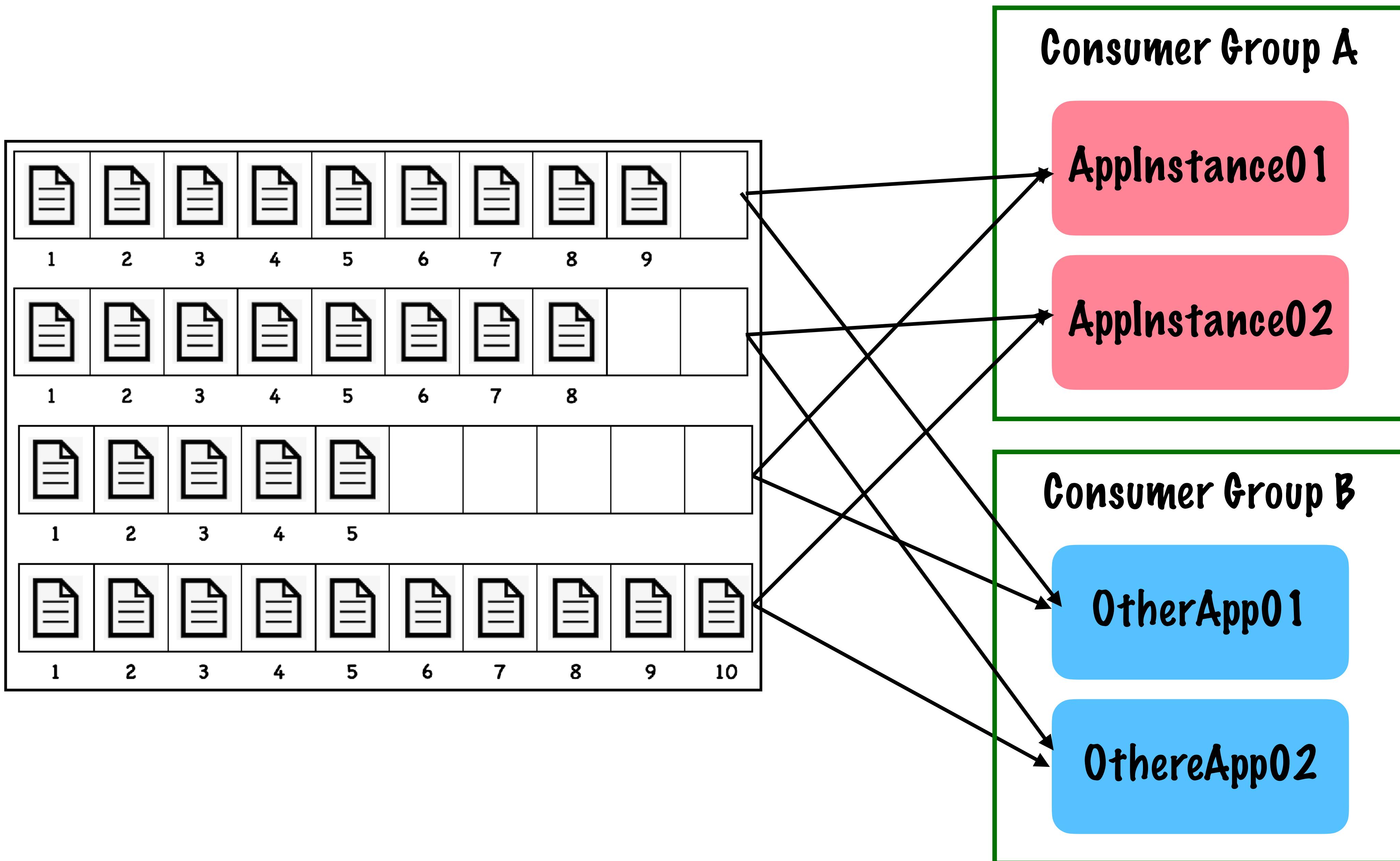


**Consumer**

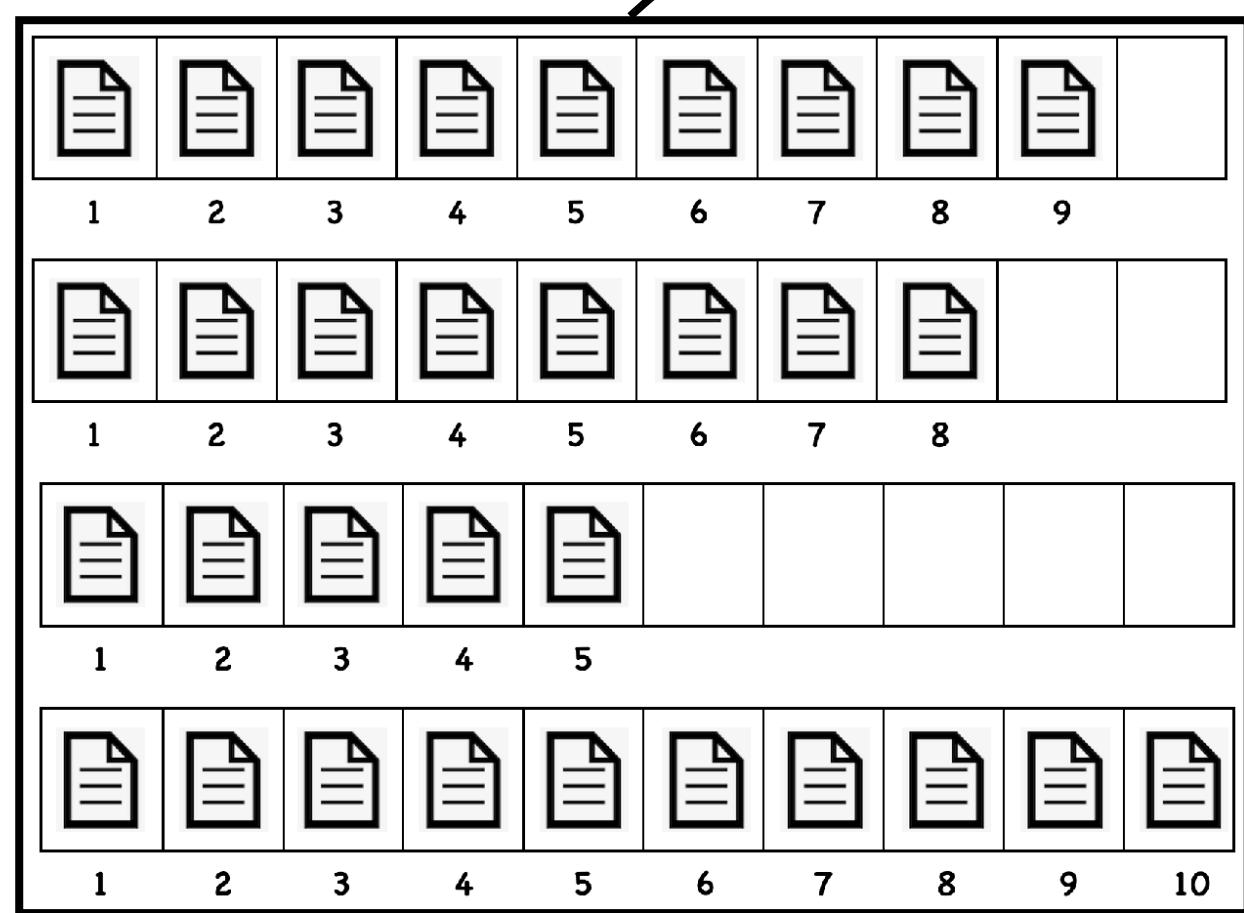
Committed Offset: 3



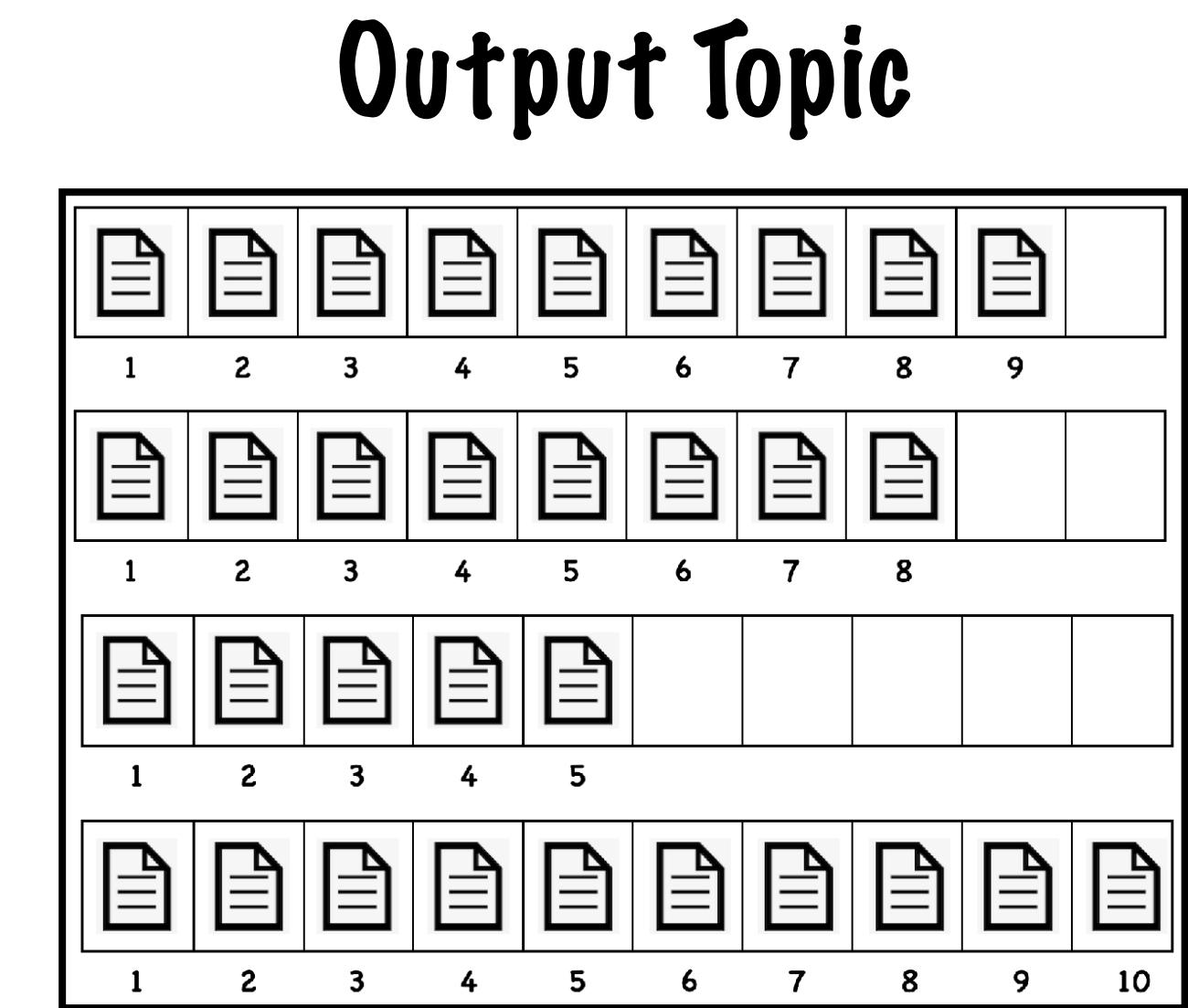
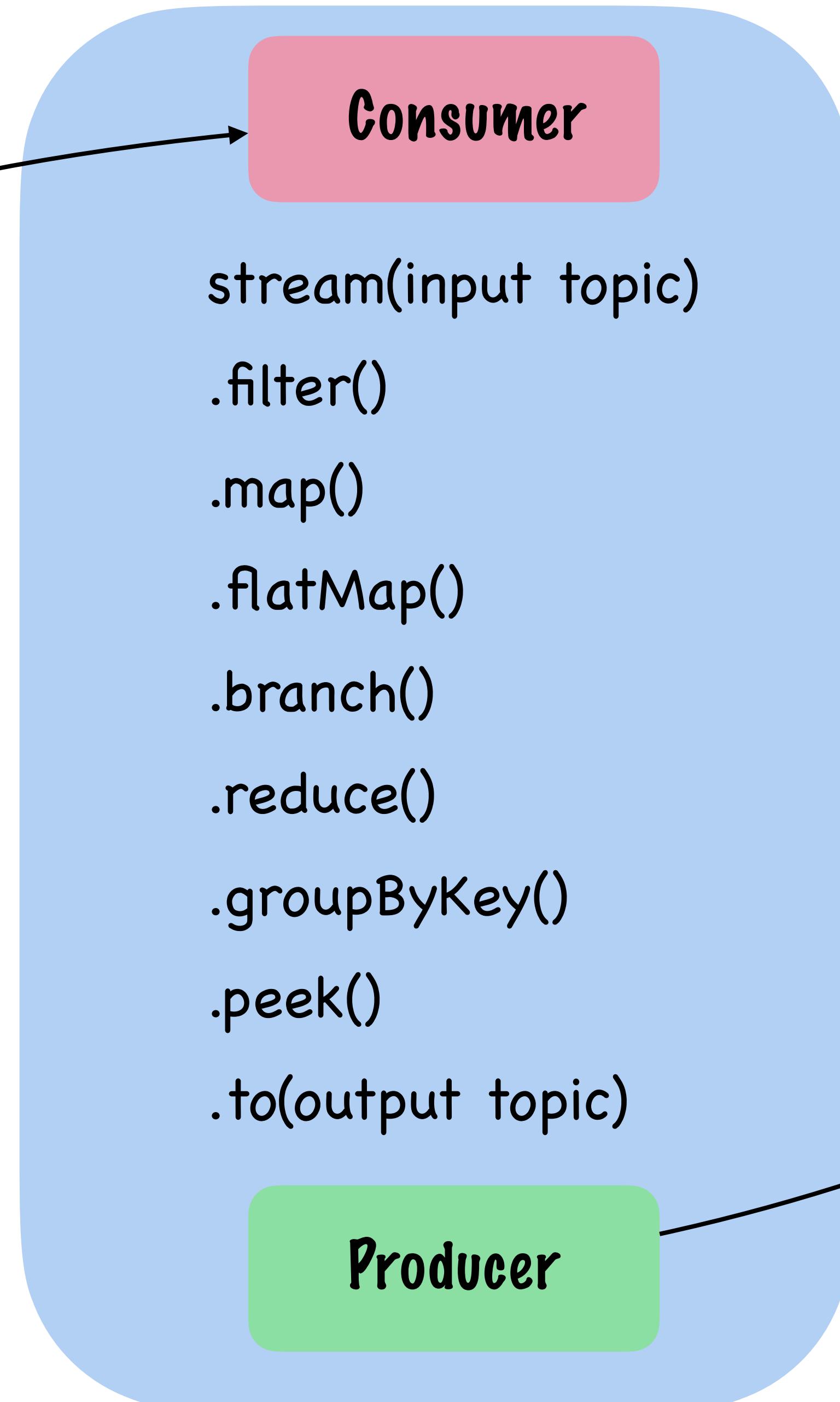




# Kafka Streams

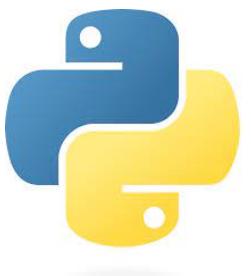


Input Topic



Output Topic

# Event Streaming Libraries



<https://github.com/quixio/quix-streams>

<https://github.com/faust-streaming/faust>



<https://github.com/lovoo/goka>

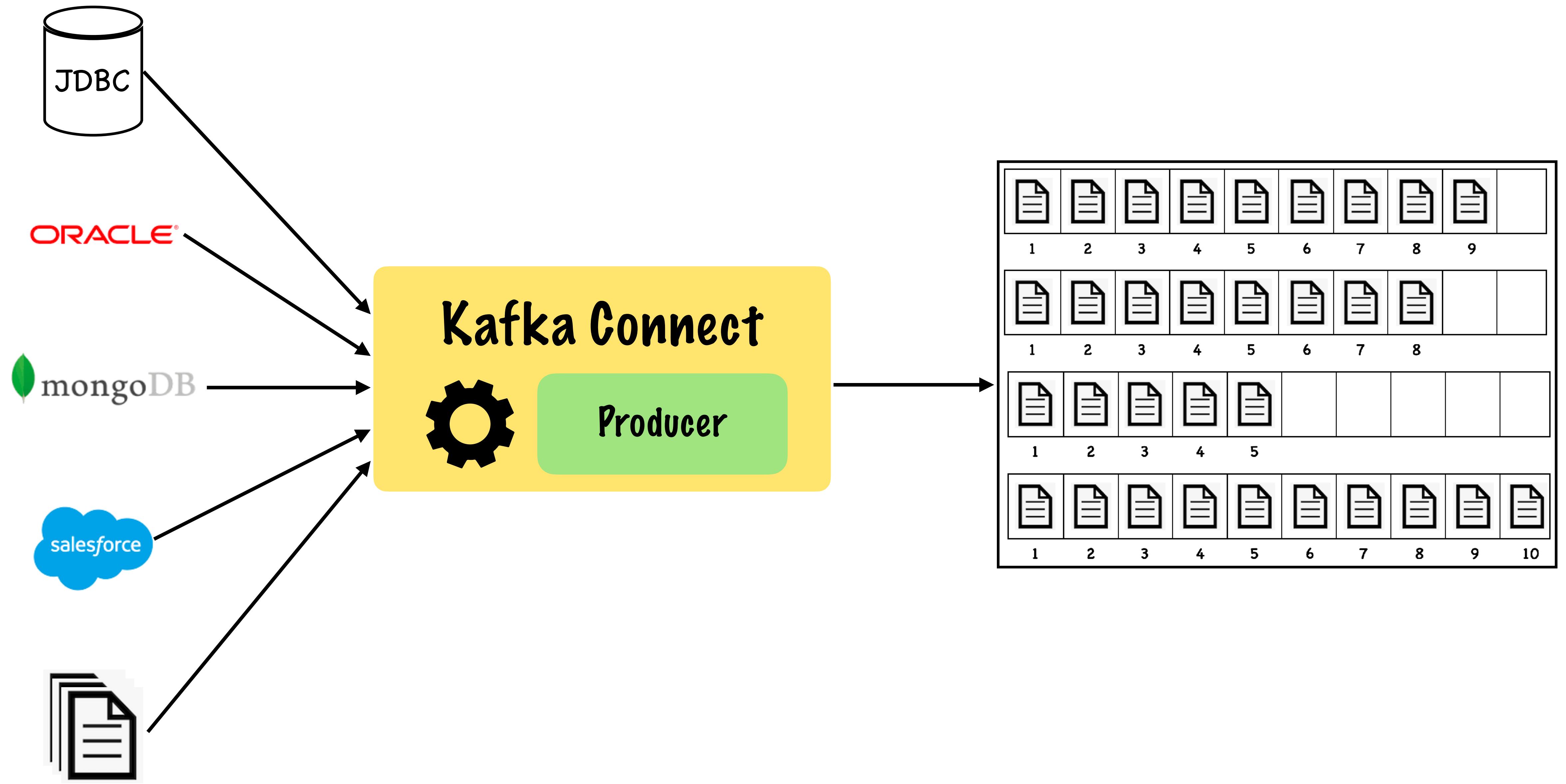


<https://github.com/quixio/quix-streams>

<https://github.com/LGouellec/kafka-streams-dotnet>

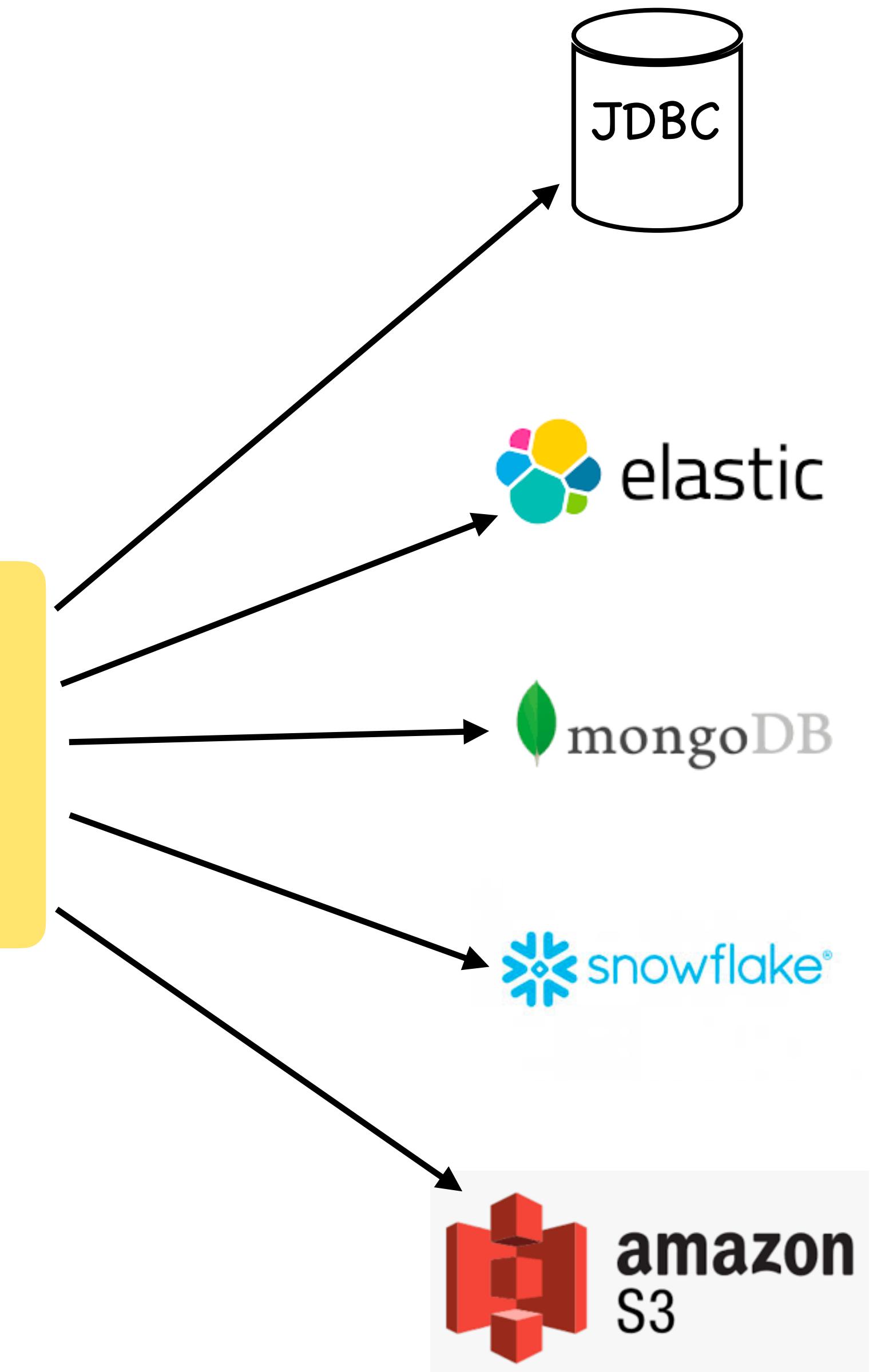
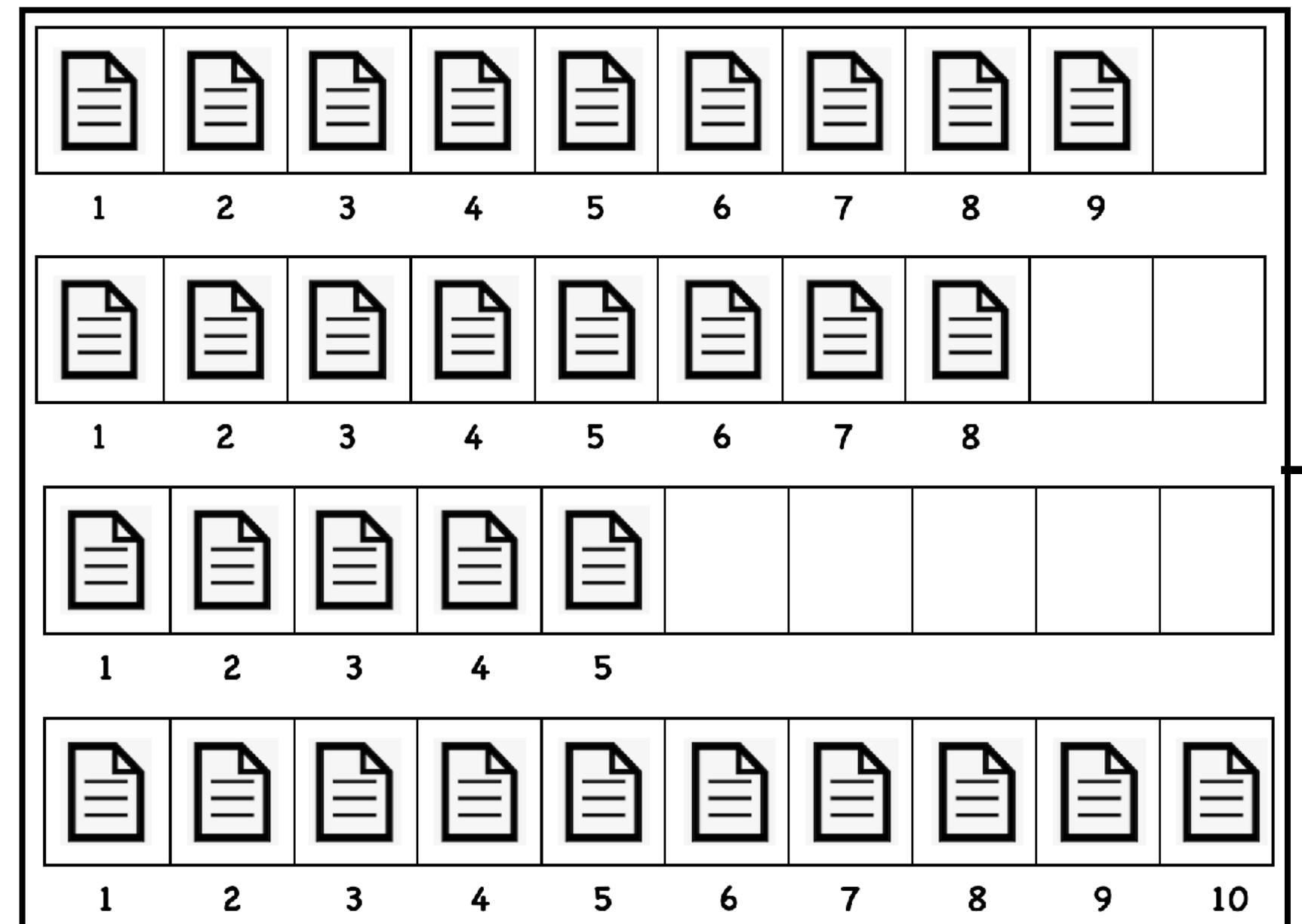


<https://github.com/ArroyoSystems/arroyo>



[dave@quix.io](mailto:dave@quix.io)

@daveklein





```
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"
```

# Command Line Tools

## Kafka Shell Scripts

- Start and stop brokers
- Create / manage topics
- Produce / consume events

Confluent CLI - <https://docs.confluent.io/confluent-cli/current/overview.html>

- Start / stop services in Confluent Platform / Cloud
- Manage access control and secrets



kcat (formerly Kafkacat) - <https://github.com/edenhill/kcat>

- Produce / consume events in style

kcctl (Casey Cuddle) - <https://github.com/kcctl/kcctl>

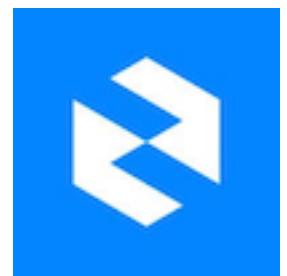
- Command-line client for Kafka Connect

# Graphical User Interfaces



## Confluent Control Center

<https://docs.confluent.io/platform/current/control-center/overview.html>



## Conduktor

<https://www.conduktor.io>



## Kafdrop

<https://github.com/obsidiandynamics/kafdrop>



## akHQ

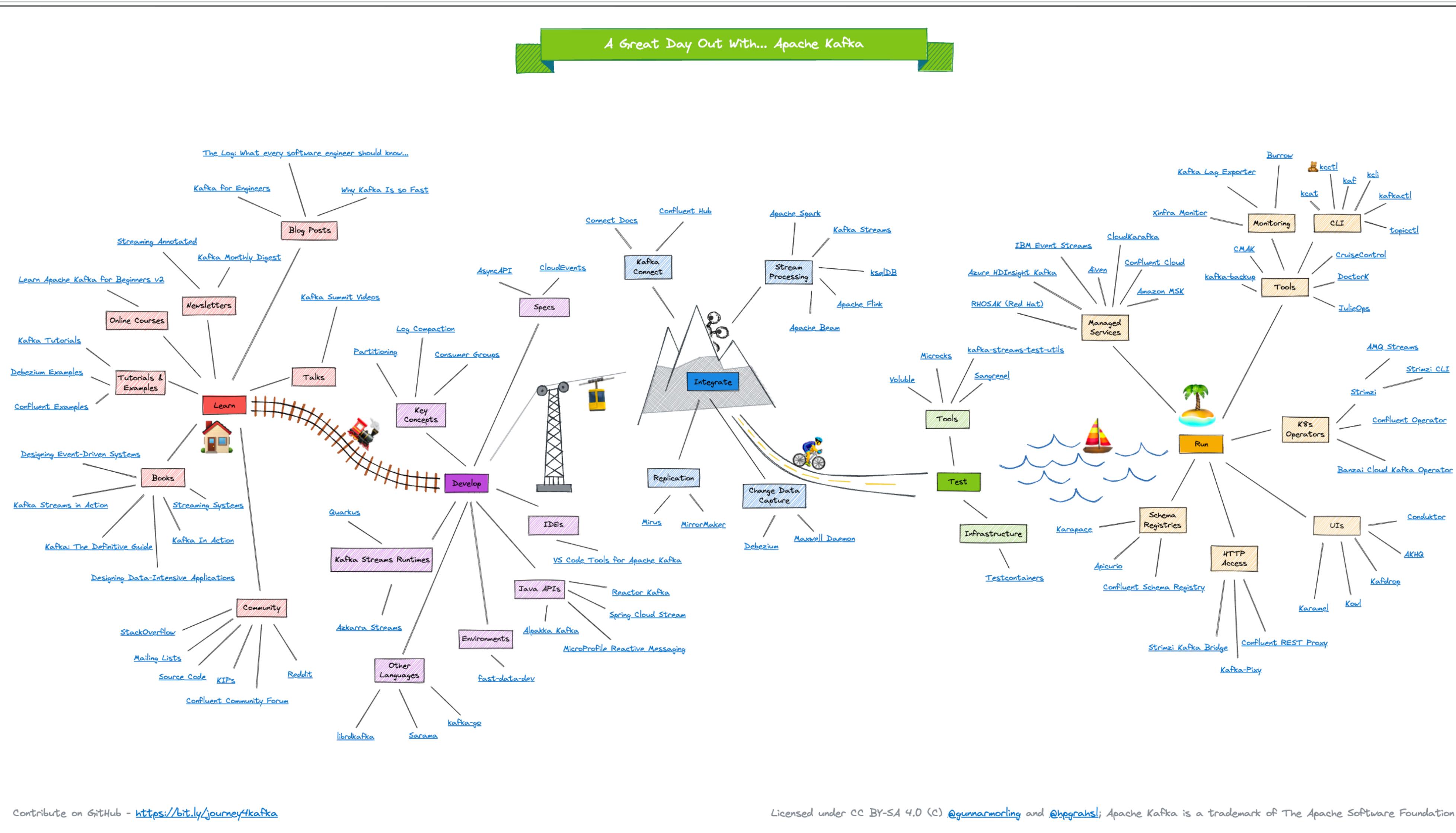
<https://github.com/tchiotludo/akhq>



## UI For Apache Kafka

<https://github.com/provectus/kafka-ui>

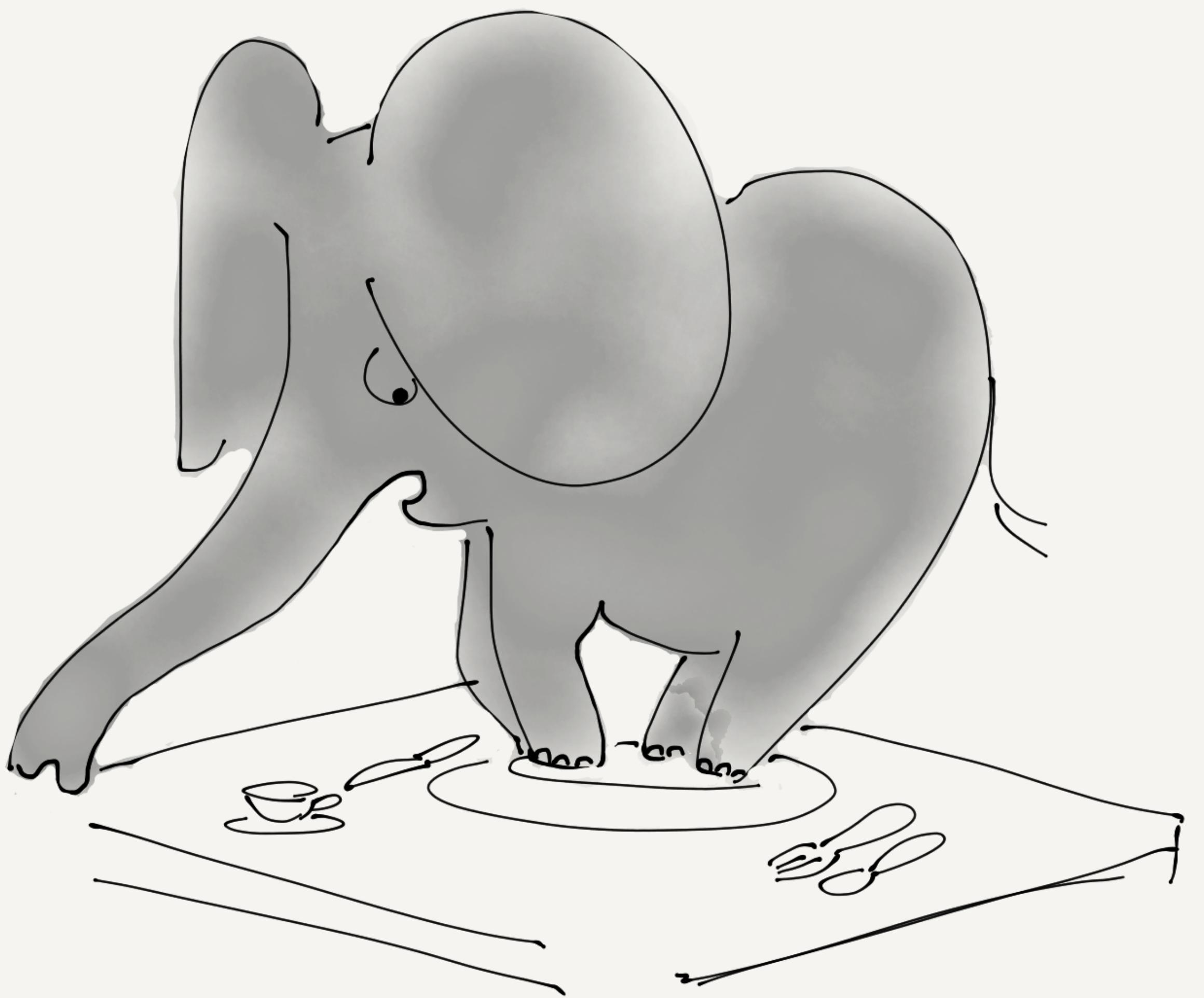
# A Great Day Out With Kafka



dave@quix.io

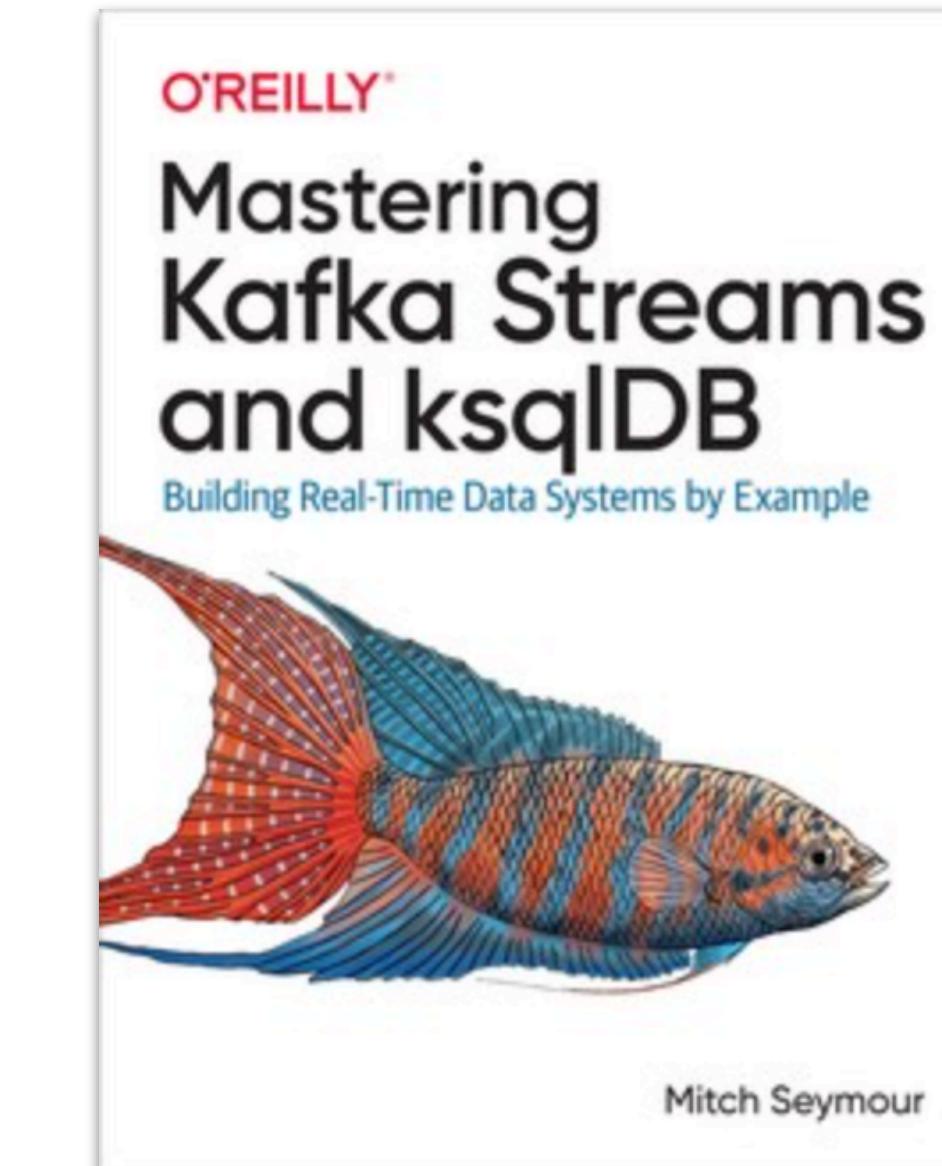
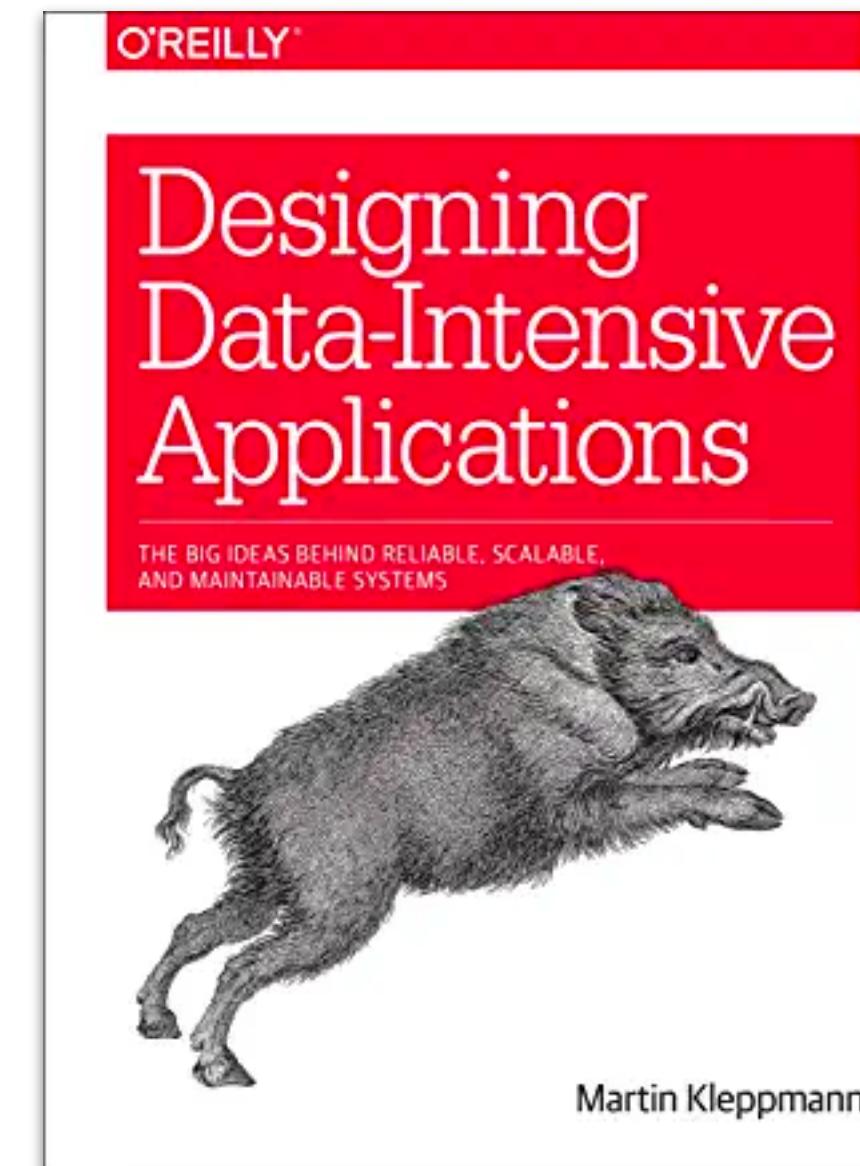
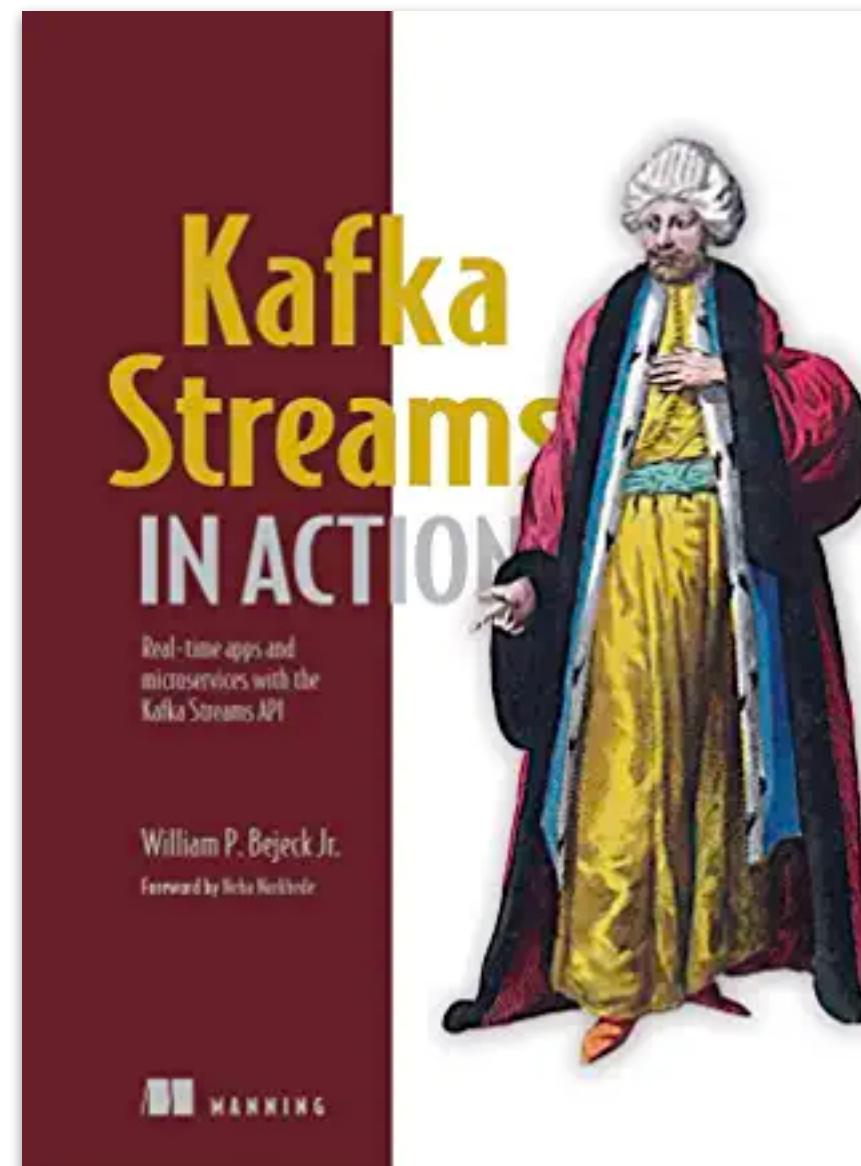
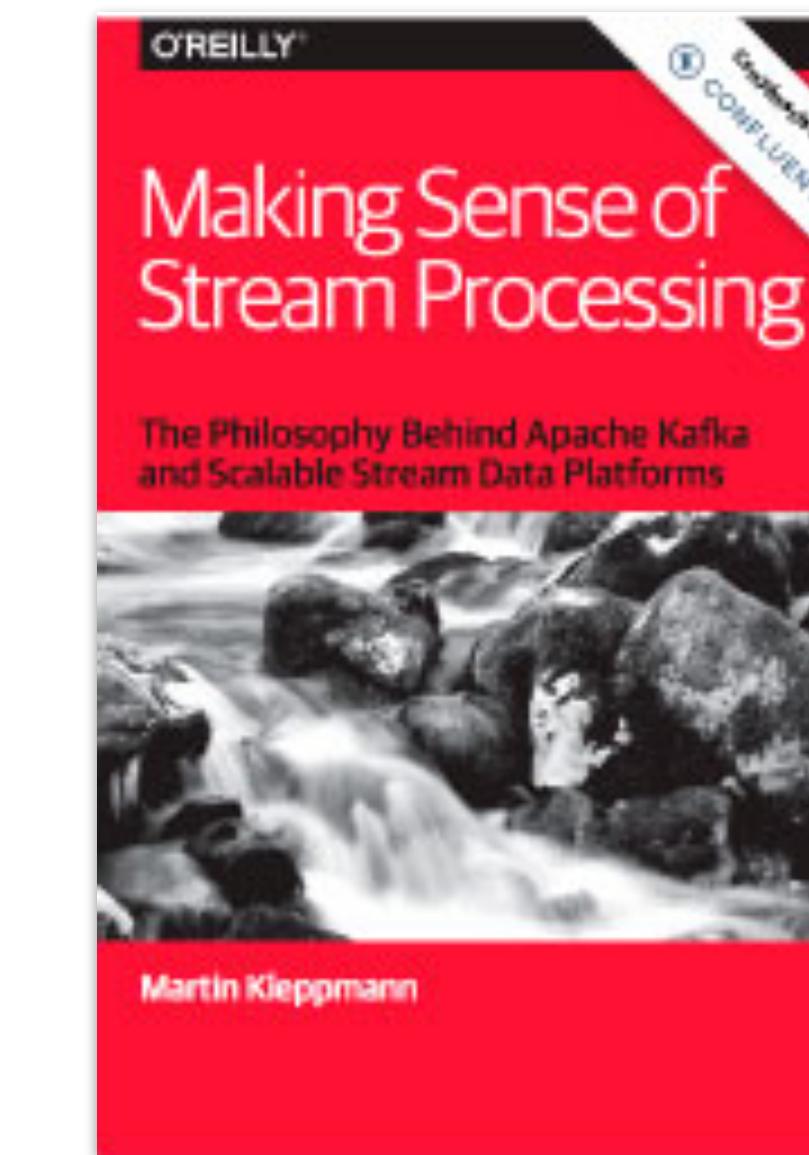
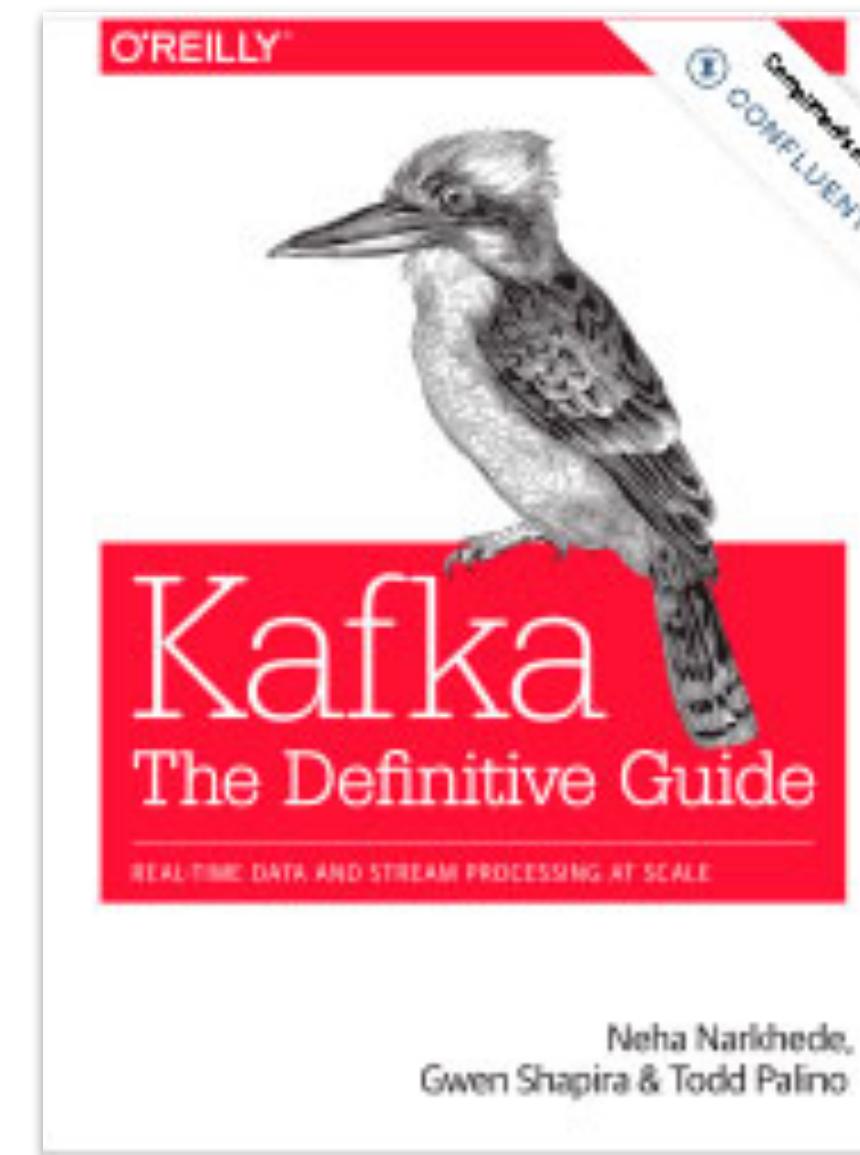
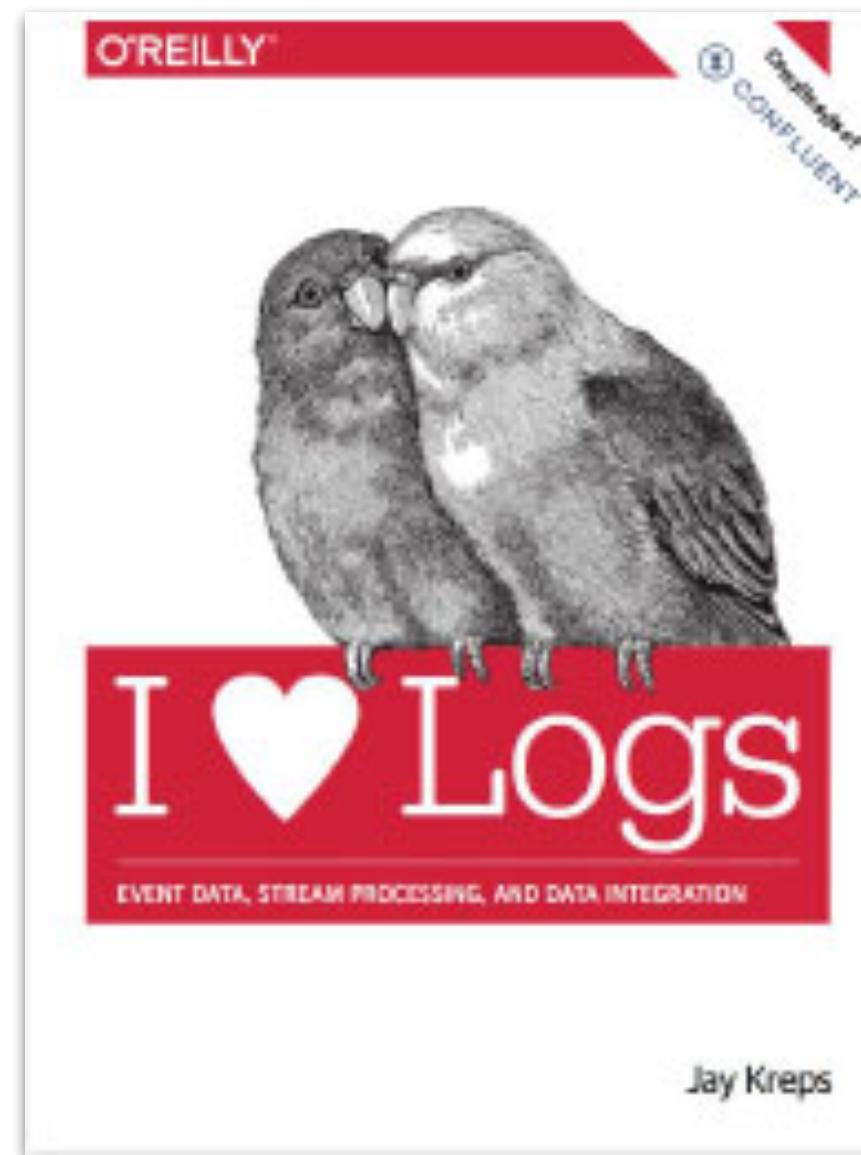
<https://a-great-day-out-with.github.io/kafka.html>

@daveklein



dave@quix.io

@daveklein



# Blogs and Articles

Kafka articles on dev.to - <https://dev.to/t/kafka>

Kafka on Medium - <https://medium.com/search?q=kafka>

Confluent's blog - <https://www.confluent.io/blog>

Robin Moffatt's blog - <https://rmoff.net>

Kafka on Aiven's blog - <https://aiven.io/blog/category/kafka>

Quix Blog - <https://aiven.io/blog/category/kafka>

[Pages](#)[Blog](#)

## SPACE SHORTCUTS

[Retrospectives](#)

## CHILD PAGES

[Kafka Improvement Proposals](#)[KIP-500: Replace ZooKeeper ...](#)

# KIP-500: Replace ZooKeeper with a Self-Managed Metadata Quorum

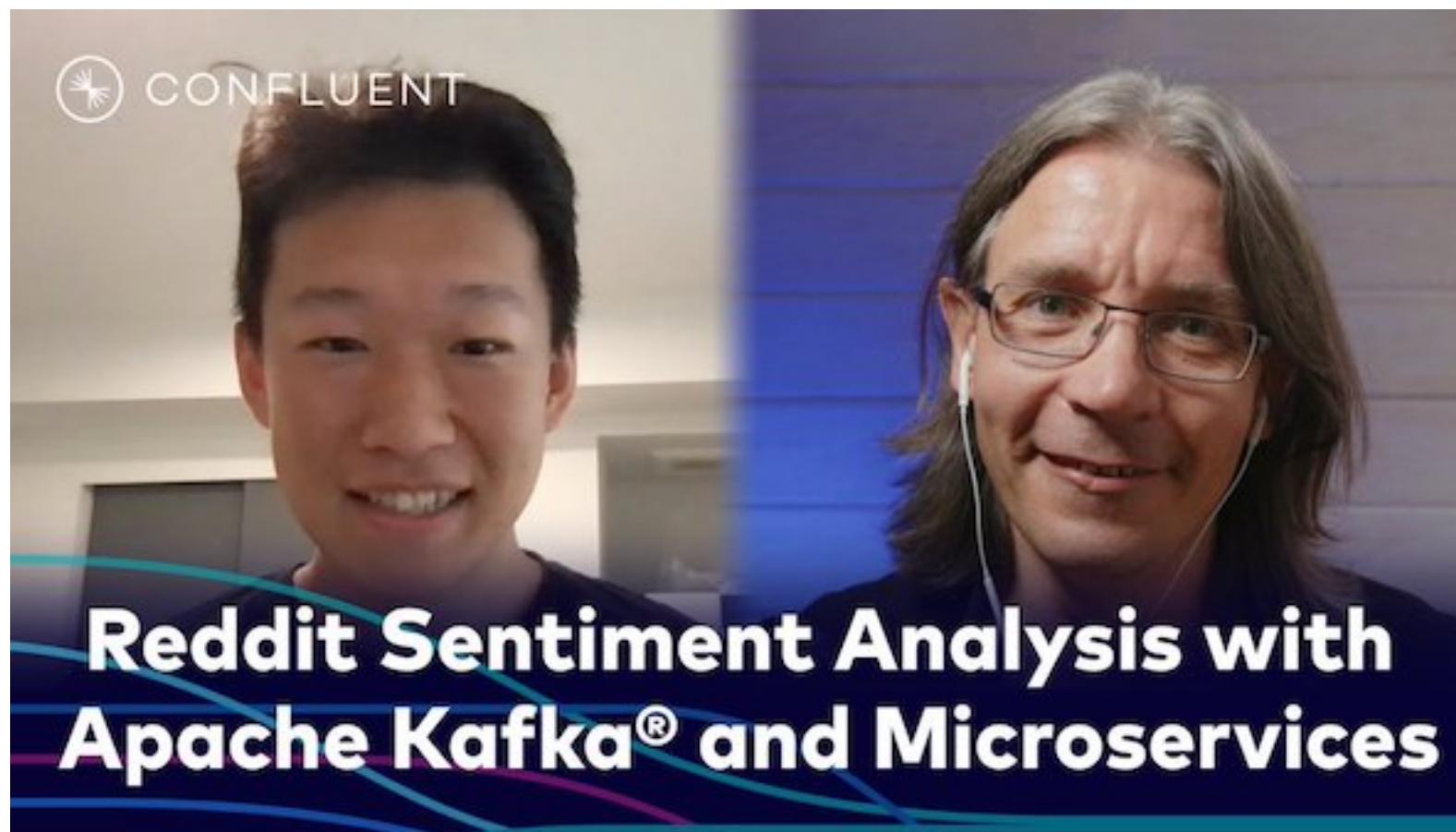
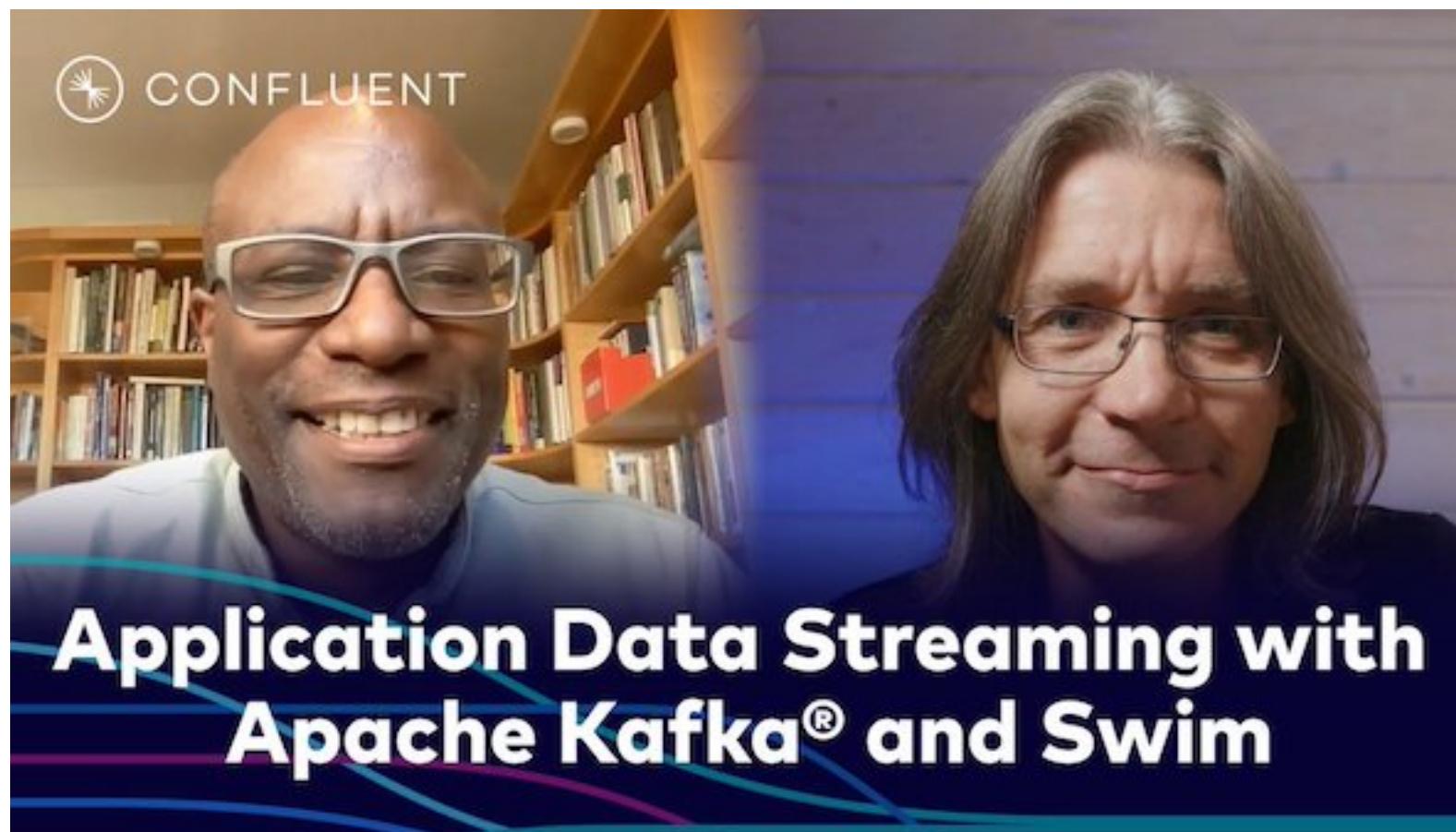
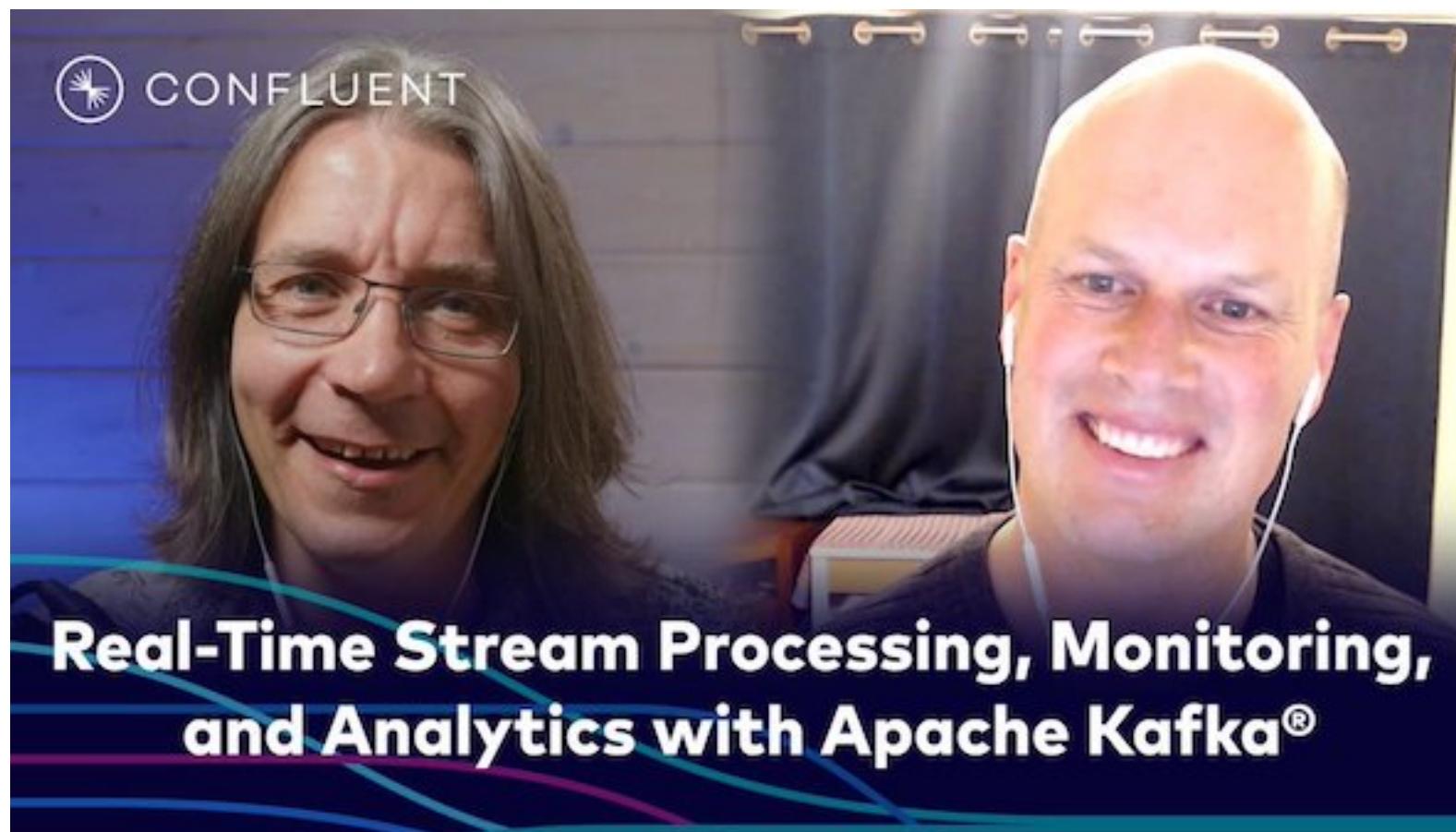
Created by Colin McCabe, last modified on Jul 09, 2020

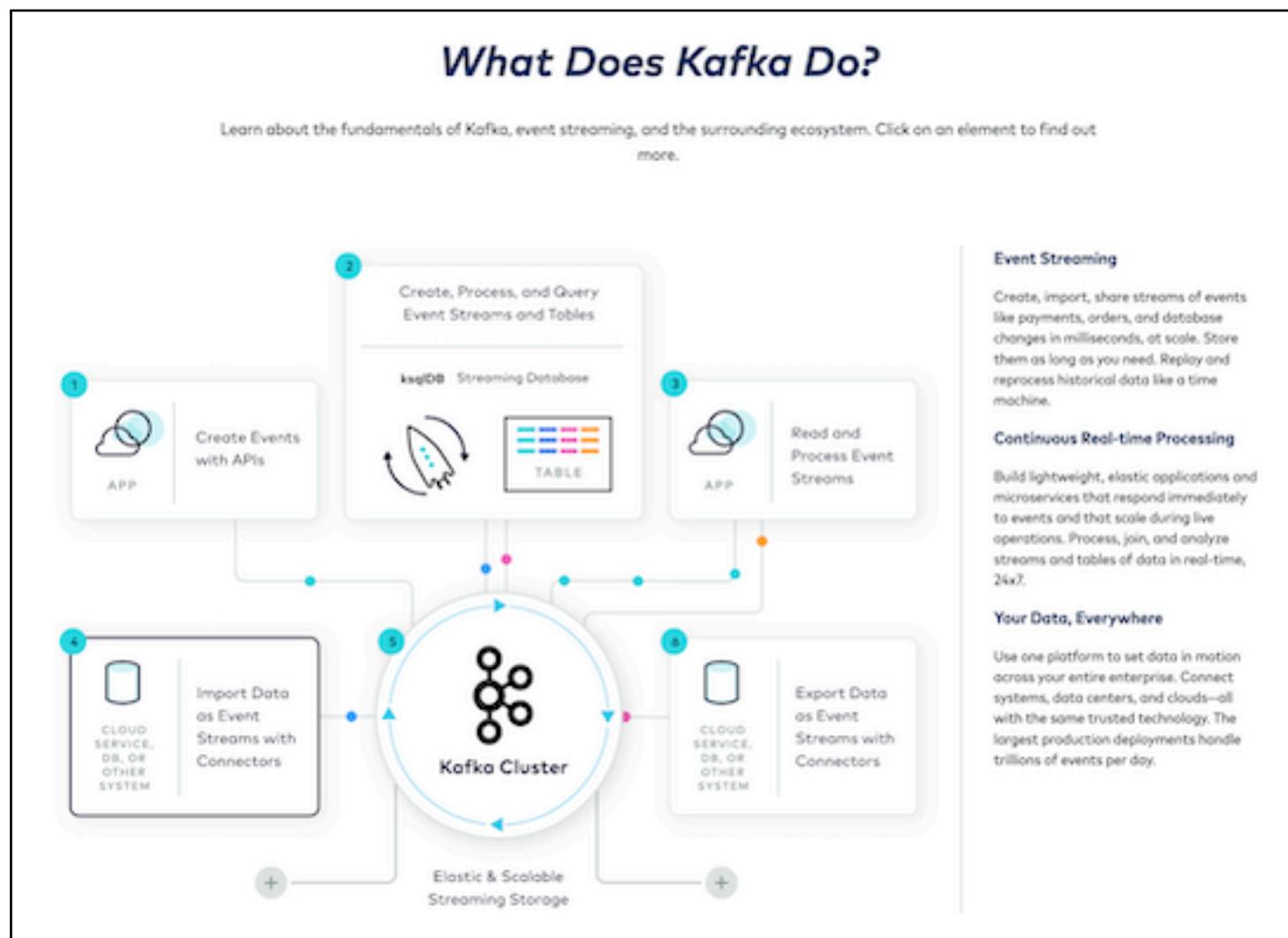
- Status
- Motivation
  - Metadata as an Event Log
  - Simpler Deployment and Configuration
- Architecture
  - Introduction
  - Overview
  - The Controller Quorum
  - Broker Metadata Management
  - The Broker State Machine
  - Broker States
    - Offline
    - Fenced
    - Online
    - Stopping
  - Transitioning some existing APIs to Controller-Only
  - New Controller APIs
  - Removing Direct ZooKeeper Access from Tools
- Compatibility, Deprecation, and Migration Plan
  - Client Compatibility
  - Bridge Release
  - Rolling Upgrade
    - Upgrade to the Bridge Release
    - Start the Controller Quorum Nodes
    - Roll the Broker Nodes
    - Roll the Controller Quorum
- Rejected Alternatives
  - Pluggable Consensus
- Follow-on Work
- References

## Status

Current state: Accepted

# Streaming Audio Podcast with Kris Jenkins





### Apache Kafka® - Articles

Learn more about Kafka with these deep-dive articles.

 <b>Create, Process, and Query Event Streams and Tables</b>	 <b>What Is Apache Kafka?</b>	 <b>Event Streaming vs Related Trends</b>	 <b>KRaft: Apache Kafka Without ZooKeeper</b>
 <b>ksqlDB - Streaming Database</b>	 <b>Event Streaming</b>	 <b>Building Systems Using Transactions in Apache Kafka®</b>	 <b>Continuous Real-time Processing</b>
 <b>Read and Process Event Streams</b>	 <b>Event Streaming</b>	 <b>Event Source</b>	 <b>Event Sink</b>
 <b>Import Data as Event Streams with Connectors</b>	 <b>Export Data as Event Streams with Connectors</b>	 <b>Event Storage</b>	 <b>Event Processing</b>
 <b>APP</b>	 <b>CLOUD SERVICE, DB, OR OTHER SYSTEM</b>	 <b>Stream Processing</b>	 <b>Table</b>
 <b>APP</b>	 <b>CLOUD SERVICE, DB, OR OTHER SYSTEM</b>	 <b>Compositional Patterns</b>	

### Event Streaming Patterns

Event Streaming Platforms provide an architecture that enables software to react and operate as events occur. These platforms allow for software components to work together in a real-time, decoupled, and scalable fashion. When software is modeled as streams of events, new capabilities surface along with new unique technical challenges.

This catalog contains simple and reusable architectural patterns that can be applied over event streaming systems. When composed together, these patterns can help meet the design demands of modern real-time distributed systems. The following patterns are categorized by their function in the event streaming system, including sourcing data, processing events as streams, to integrations with external systems.

#### Compositional Patterns

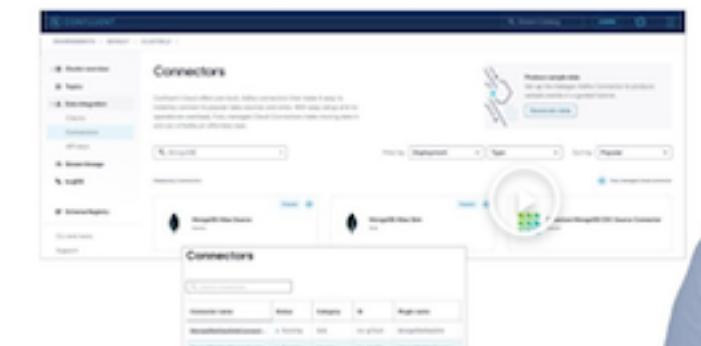
Design patterns help you plan, implement, and communicate about software architectures. The main groups of patterns for event streaming are shown in this diagram, including compositional patterns that apply across several areas.

Click on each group to find out more.

## Kafka Deep Dives



### Introduction to Kafka Connect



### KAFKA CONNECT



### ksqlDB Introduction



## In-depth Video Courses

## CONFLUENT Developer



**KAFKA TUTORIALS**

**Kafka Tutorials**

Dozens of step-by-step tutorials that show you the basics of developing streaming applications based on Apache Kafka and Confluent.

**PAC-MAN**

**Streaming Pac-Man**

Streaming Pac-Man is the funniest application that you will ever see while applying principles of streaming analytics using Apache Kafka. Built around the famous Pac-Man game, this application ingests and stores events from the game into Kafka topics and allows you to process them in near real-time using ksqlDB.

**ETL pipelines**

Demo showing a complete ETL (extract, transform, load) pipeline in the cloud, using fully-managed services on Confluent Cloud.

**MQTT Tracker**

This demo shows how to stream data from your phone through MQTT and process it in Kafka.

**GitHub** **flux** **kubernetes** **DevOps for Apache Kafka**

## Live Coding

## Welcome to Event Streaming Patterns

Event Streaming Platforms provide an architecture that enables software to react and operate as events occur. These platforms allow for software components to work together in a real-time, decoupled, and scalable fashion. When software is modeled as streams of events, new capabilities surface along with new unique technical challenges.

This catalog contains simple and reusable architectural patterns that can be applied over event streaming systems. When composed together, these patterns can help meet the design demands of modern real-time distributed systems. The following patterns are categorized by their function in the event streaming system, including sourcing data, processing events as streams, to integrations with external systems.

Design patterns help you plan, implement, and communicate about software architectures. The main groups of patterns for event streaming are shown in this diagram, including compositional patterns that apply across several areas.

Click on each group to find out more.

## Tutorials and Demos

## Introduction

- Short learning curve
- High adoption (# on Stack Overflow Q3/2022)
  - Web dev
  - Automation
  - Machine learning
  - Data engineering
  - Microservices
  - Web applications
- Vibrant Community



APACHE KAFKA® FOR PYTHON DEVELOPERS

[https://www.youtube.com/playlist?list=PLa7VYi0yPIH1odVnZC430071CVD\\_4Sx1e](https://www.youtube.com/playlist?list=PLa7VYi0yPIH1odVnZC430071CVD_4Sx1e)

# Kafka Education Portals

Conduktor Kafkademy

<https://www.conduktor.io/kafka/>

Apache Kafka Getting Started Guide

<https://kafka.apache.org/documentation/#gettingStarted>

Aiven Developer Center - Kafka

<https://aiven.io/developer/kafka>

Tutorials Point Kafka Series

[https://www.tutorialspoint.com/apache\\_kafka/index.htm](https://www.tutorialspoint.com/apache_kafka/index.htm)

# The Joy of (Kafka) Community



<https://www.youtube.com/watch?v=MchgdIWH7MY>

# Community Forums and Slack

Confluent Community Slack

<https://www.confluent.io/en-gb/community/ask-the-community>

The Stream - Stream processing community

<https://quix.io/community>

Quix Forum

<https://forum.quix.io/>

Aiven Forum

<https://quix.io/community>

Confluent Forum

<https://forum.confluent.io/>

# Formatted Messages Are Easier To Read



For a single line of code, select and click `</>`  
or wrap it in a pair of single back-ticks: `foo`

For code blocks (or logs) select all and click `</>`  
or wrap it all in triple back-ticks

```
```for (n : lotsOn) {  
    System.out.println("foo");  
}
```

<https://slack.com/help/articles/202288908-Format-your-messages>

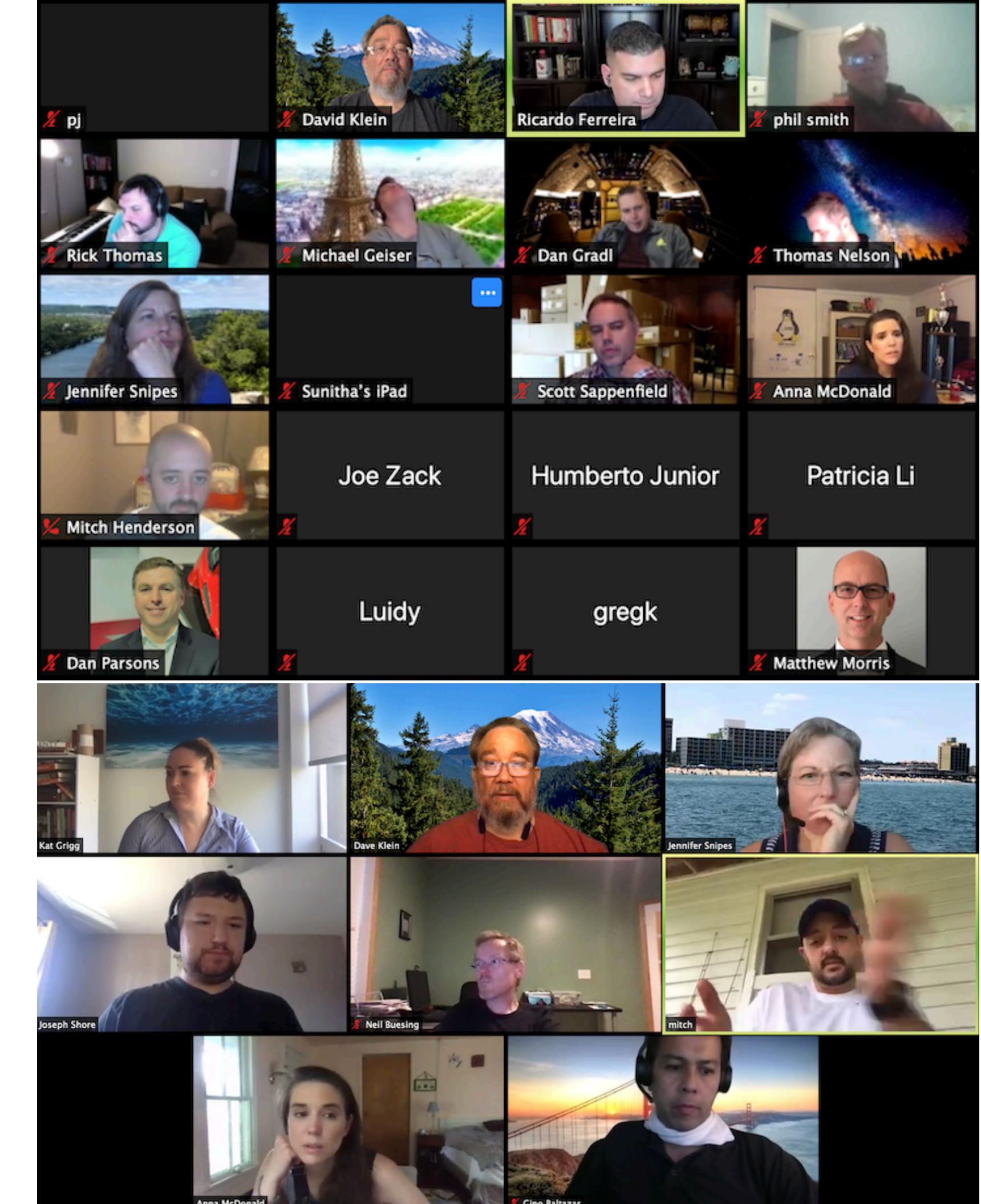


# Meetup Hub

Here you can find out about upcoming in-person and virtual meetups, as well as content from past meetup events.

[Click here for updates on new meetups](#)

[events.confluent.io/meetups](https://events.confluent.io/meetups)





# Confluent Community Catalysts

## Class of 2022/2023 Part 1

	<p>Anton Rodriguez  @antonmry</p>		<p>Bobby Calderwood  @bobbycalderwood</p>		<p>Elad Leev  @eladleev</p>		<p>Florian Hussonnois  @fhussonnois</p>		<p>Francesco Tisiot  @ftisiot</p>
	<p>Gerard Klijns  @GKlijns</p>		<p>Gunnar Morling  @gunnarmorling</p>		<p>Hans-Peter Grahsl  @hpgrahsl</p>		<p>Israel Ekpo  @IzzyAcademy</p>		<p>João Bosco Seixas  @joaboscoseixas</p>
	<p>Jordan Moore  @OneCricketeer</p>		<p>Josep Casals  @jcasals</p>		<p>Kate Stanley  @katestanley91</p>		<p>Liam Clarke-Hutchinson  @loicmdivad</p>		<p>Loic DIVAD  @loicmdivad</p>
	<p>Luan Moreno Medeiros Maciel  @luansql</p>		<p>Marcelo Costa  @marcelojscosta</p>		<p>Mateus Oliveira  @mateusoliveira_</p>		<p>Matteo Ferroni  @mattferroni</p>		<p>Mickael Maison  @MickaelMaison</p>



# Confluent Community Catalysts

Class of  
2022/2023

Part 2



Mitch  
Henderson [@Mr\\_mitchellh](#)



Neil  
Buesing [@nbuesing](#)



Nikki  
Thean [@NikkiThean](#)



Ofir  
Sharony



Olena  
Kutsenko [@OlenaKutsenko](#)



Peter  
Ko



Raúl  
Estrada [@HerrRul](#)



Ryanne  
Dolan [@DolanRyanne](#)



Saumitra  
Srivastav  [@\\_saumitra\\_](#)



Simon  
Aubury [@SimonAubury](#)



Vanitha  
Vellingiri



Wonyoung  
Choi

# Thank you!

Wednesday, 17 May

Real-time Event Processing with Python

Dave Klein

9:00 Breakout Room 4

Building Real-Time Applications at Scale: A Case Study in Cyclist Crash Detection

Tomas Neubauer

2:00 Breakout Room 3

Slide deck available at:

<https://github.com/daveklein/welcome>