Uncorking Analytics with Apache Pulsar, Apache Flink, and Apache Pinot

Viktor Gamov, Confluent | @gamussa

San Francisco, CA, October 2024

Before We Proceed...

https://gamov.dev/uncorking-analytics

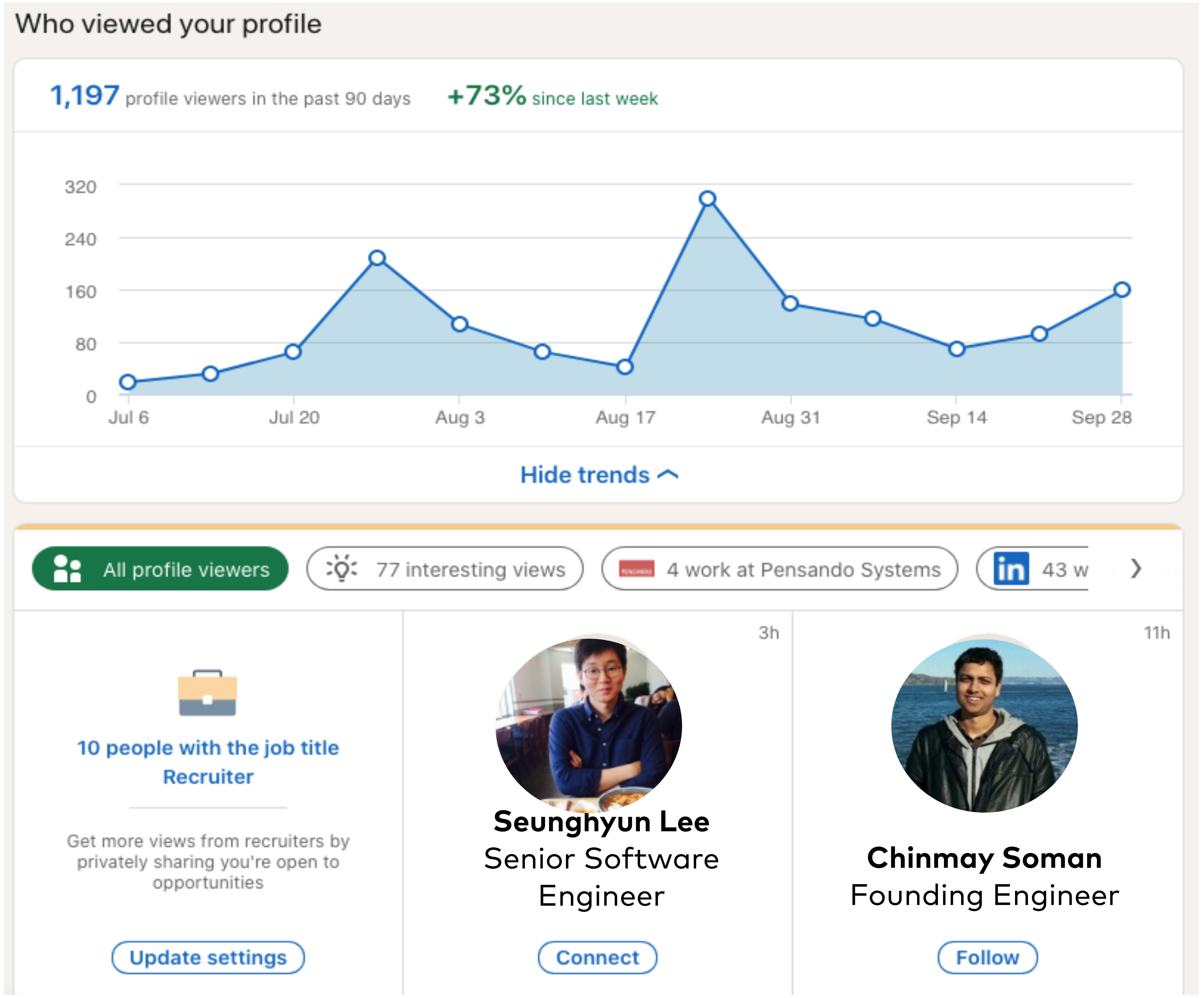
A Taxonomy of Analytics

REAL-TIME
BATCH
INTERNAL
EXTERNAL
OBSERVABILITY/ MONITORING
USER-FACING ANALYTICS
DASHBOARDS
REPORTING
REPORTING FEATURES

REAL-TIME
BATCH
INTERNAL
EXTERNAL
DATADOG
PINOT
CLICKHOUSE
ROCKSET
DRUID
TRINO,
PRESTO,
BIGQUERY
SNOWFLAKE
HADOOP
LEGACY
DWH
SEE ALSO
(BUT WITH CACHING)

Who Does Real-Time Analytics?

Who Viewed My Profile?


Who viewed your profile

1,197 profile viewers in the past 90 days +73% since last week

10 people with the job title Recruiter

Get more views from recruiters by privately sharing you're open to opportunities

Seunghyun Lee
Senior Software Engineer

Chinmay Soman
Founding Engineer

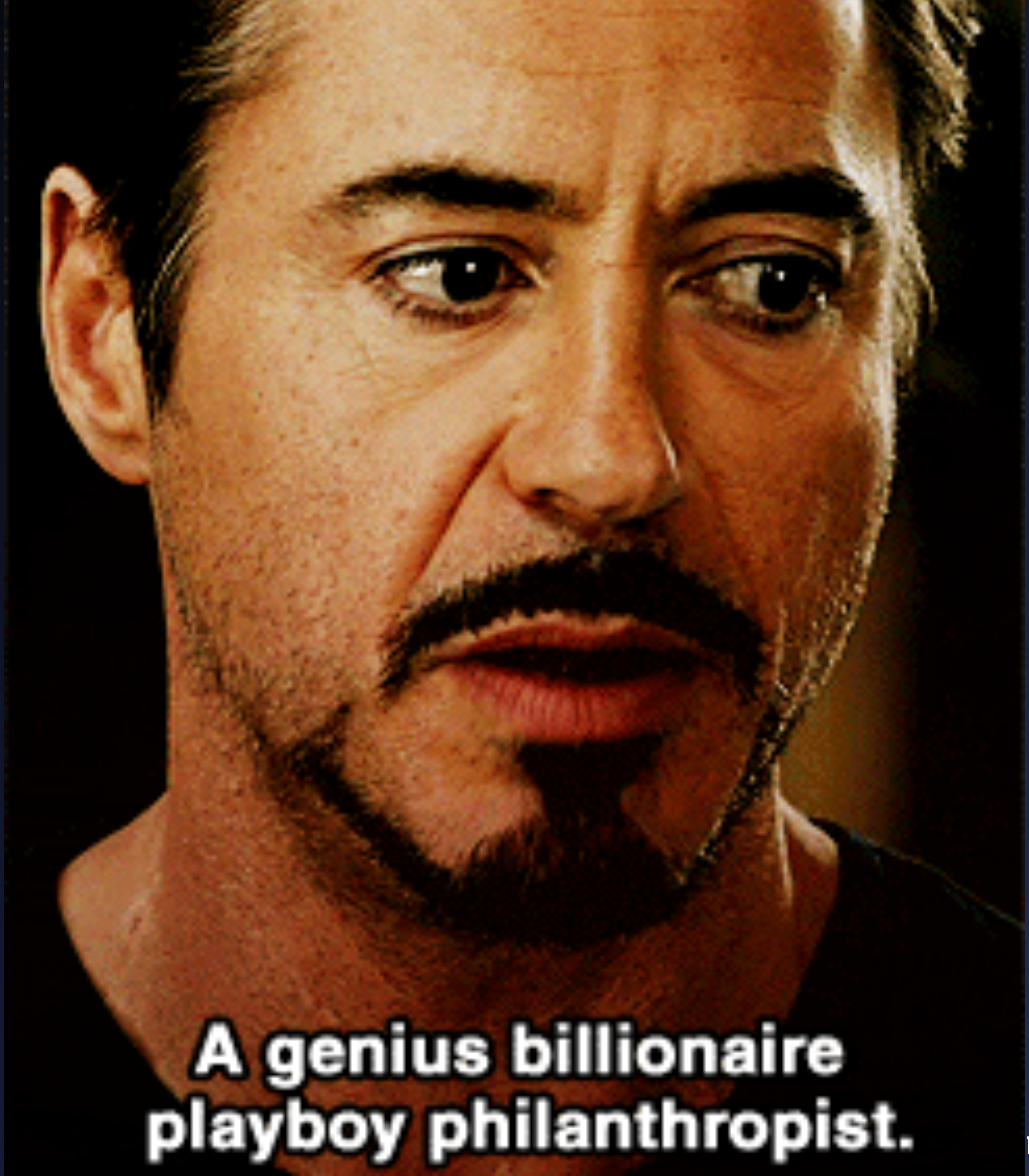
Total users	700 Million+
QPS	100,000s
Latency SLA	< 100 ms p99th
Freshness	Seconds

Viktor

GAMOV

- Principal Developer Advocate | Confluent
- Java Champion
- O'Reilly and Manning Author

~~Twitter~~ X: @gamussa



mit

What is Apache Pinot™?

“Apache Pinot is a real-time distributed OLAP database, designed to serve OLAP workloads on streaming data with extreme low latency and high concurrency.”

The essence of real-time analytics

LATENCY

The amount of time it takes to execute a query

CONCURRENCY

The ability of a system to handle multiple queries simultaneously

FRESHNESS

The up-to-date nature of data in the system

The essence of real-time analytics

LATENCY

As low as 10ms

CONCURRENCY

As many as 100,000 queries per second

FRESHNESS

Seconds from event time till queryable in Pinot

OLTP

OLTP

- Transaction focused
- Write-heavy workloads
- Often involves a single record per operation

OLAP

- Aggregation-focused
- Read-heavy workloads
- Often involves many records in one operation

Data Model

- Pinot uses the completely familiar **tabular data model**
- **Table** creation and **schema** definition **expressed in JSON**
- **Queries** expressed in **SQL**

Architecture:
Tables and Segments

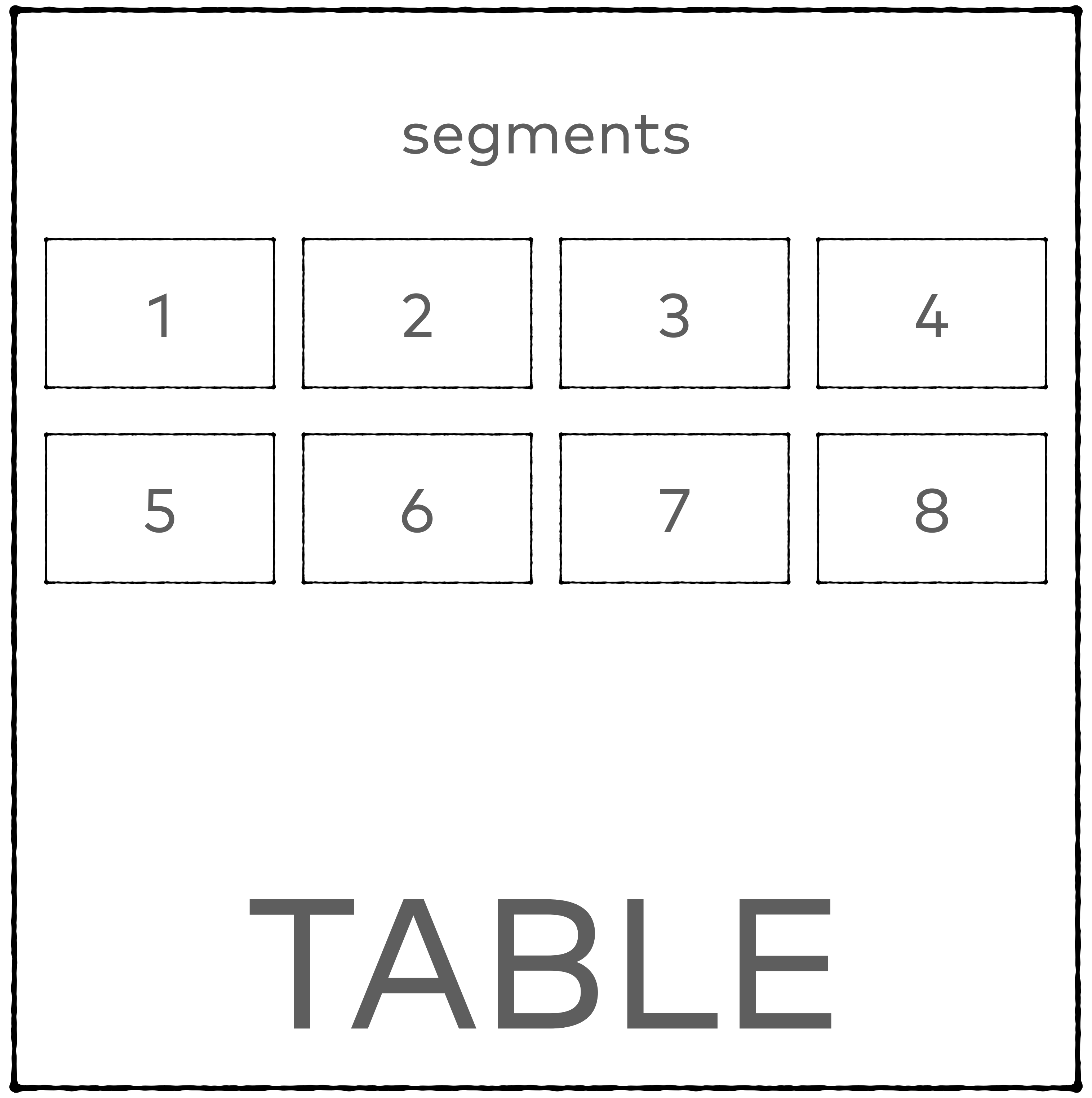
Tables

- The basic **unit of data storage** in Pinot
- Composed of rows and columns
- Expected to **scale to arbitrarily large row counts**
- Defined using a **schema** and `tableConfig` JSON file
- Three varieties: **offline**, **real-time**, and hybrid
- Every **column** is either a metric, dimension, or date/time

Segments

- Tables are **split into units of storage** called segments
- Similar to **shards** or partitions but transparent to you, the user
- For offline tables, segments are created outside of Pinot and pushed into the cluster using a REST API
- For real-time tables, segments are created automatically from events sourced by the event streaming system (e.g., Pulsar, Kafka)
- Standard utilities support batch ingest from standard file types (AVRO, JSON, CSV)
- APIs are available to create segments from Spark, Flink, and Hadoop

Segments

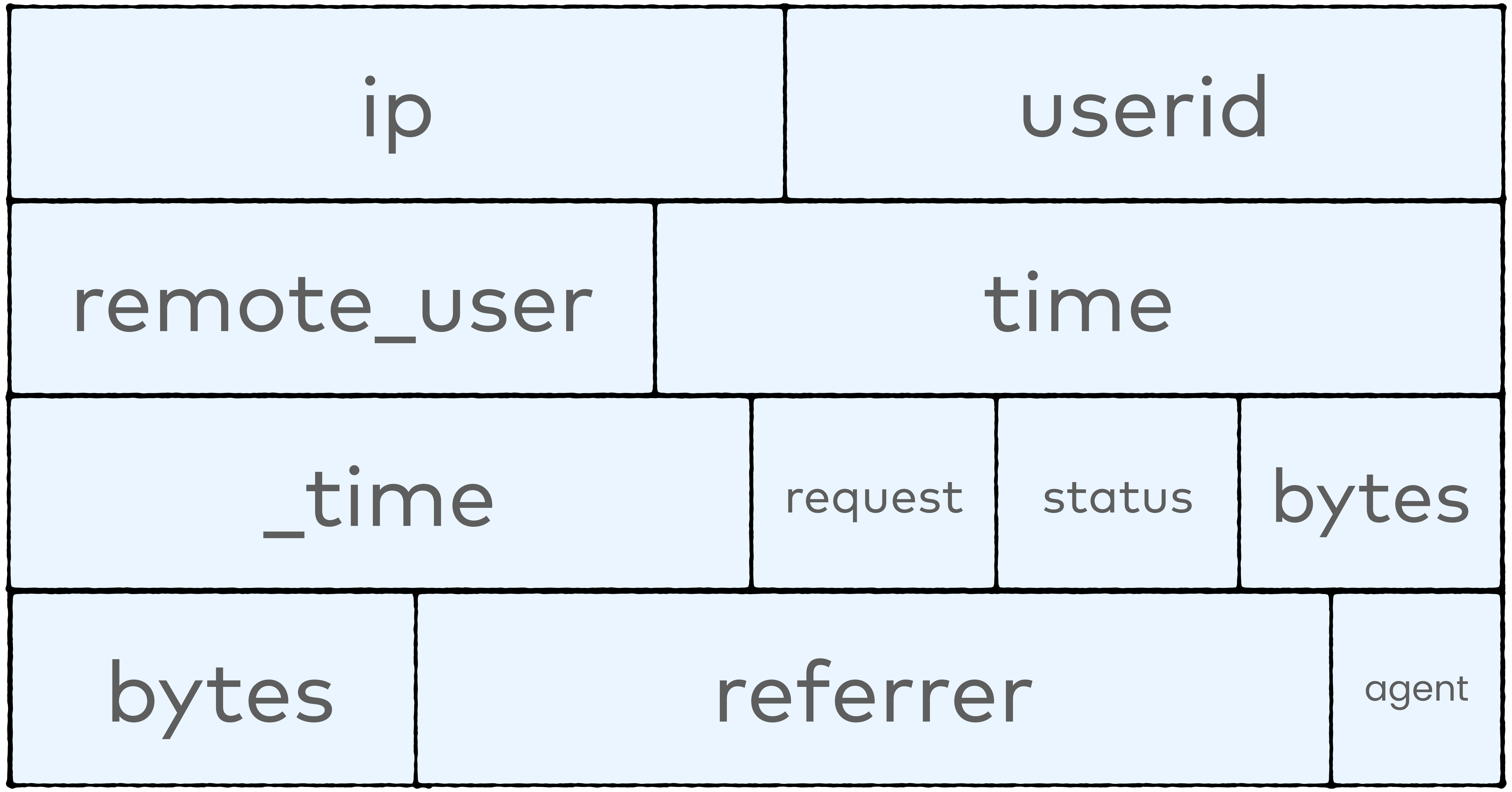


Segment Structure

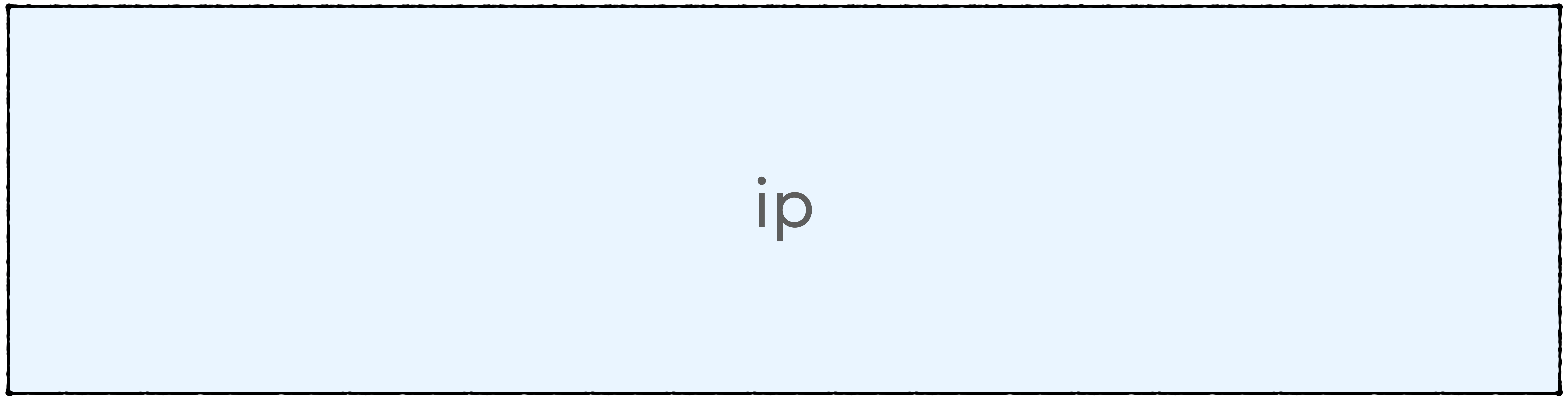
- Pinot is a **columnar** database
- All of a segment's **values** for a **single column are stored contiguously**
- Dimension columns are typically **dictionary-encoded**
- **Indexes** are stored as a part of the **segment**
- **Segments** are **immutable** once written
- Segments have a **configurable retention period**

Segment Structure

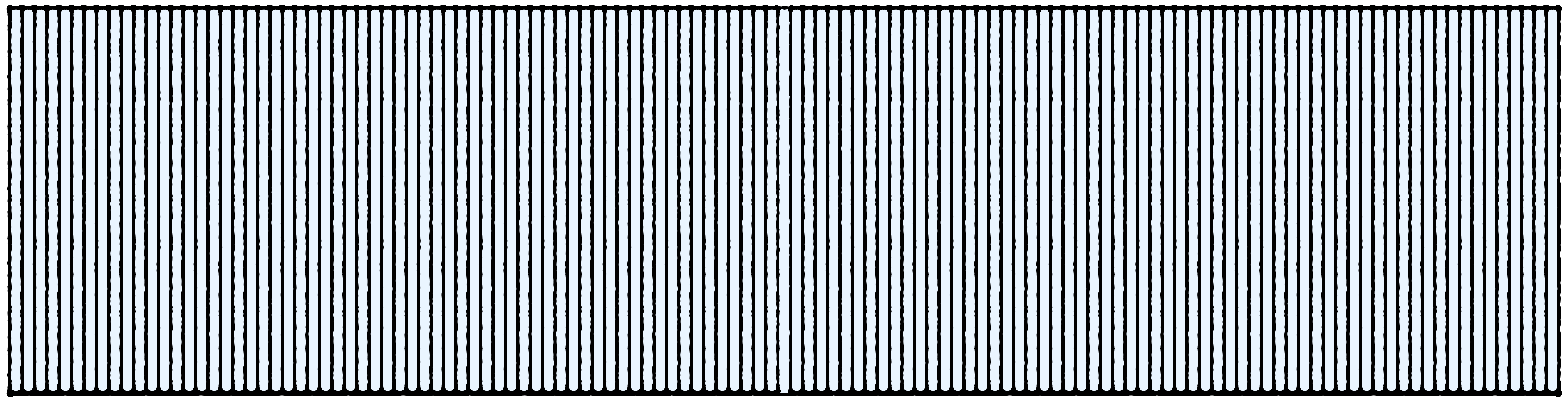
```
...,
{
   "ip": "111.173.165.103",
   "userid": 10,
   "remote_user": "-",
   "time": "3271",
   "_time": 3271,
   "request": "GET",
   "status": "406",
   "bytes": "1289",
   “referrer": "-",
   "agent": "Mozilla/5.0"
},
...,
```



Segment Structure



Segment Structure



166.27.69.94
202.43.225.122
250.192.178.235
165.193.151.176
211.235.25.163
182.45.66.204
13.213.178.183
199.191.187.233
239.36.131.30
132.116.134.205

Part 1

Batch Ingestion in Pinot

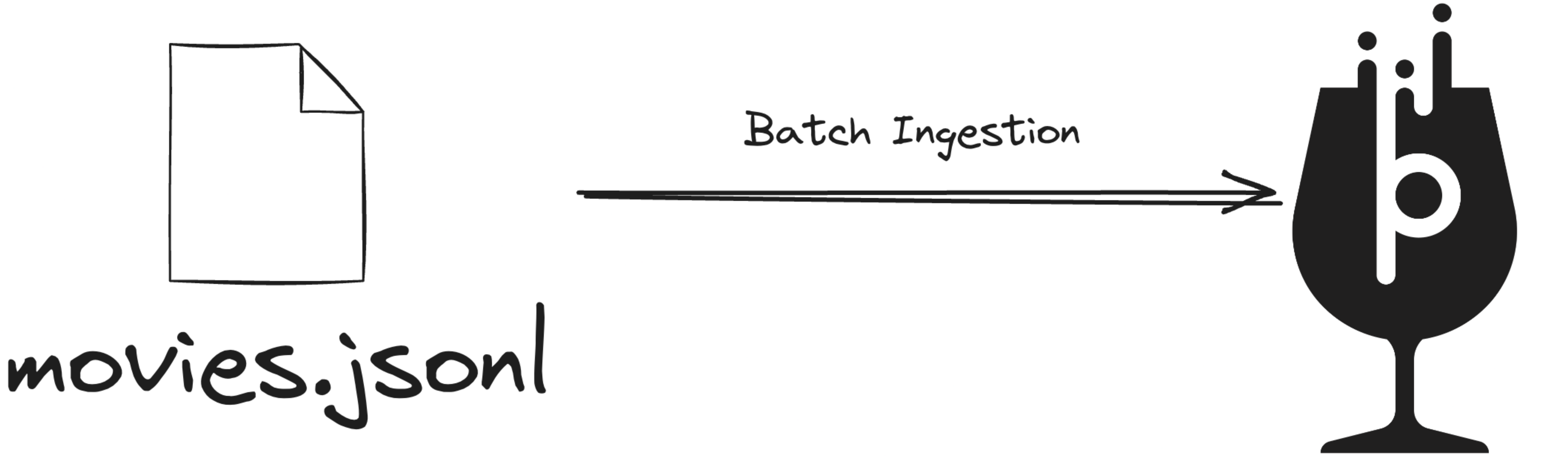
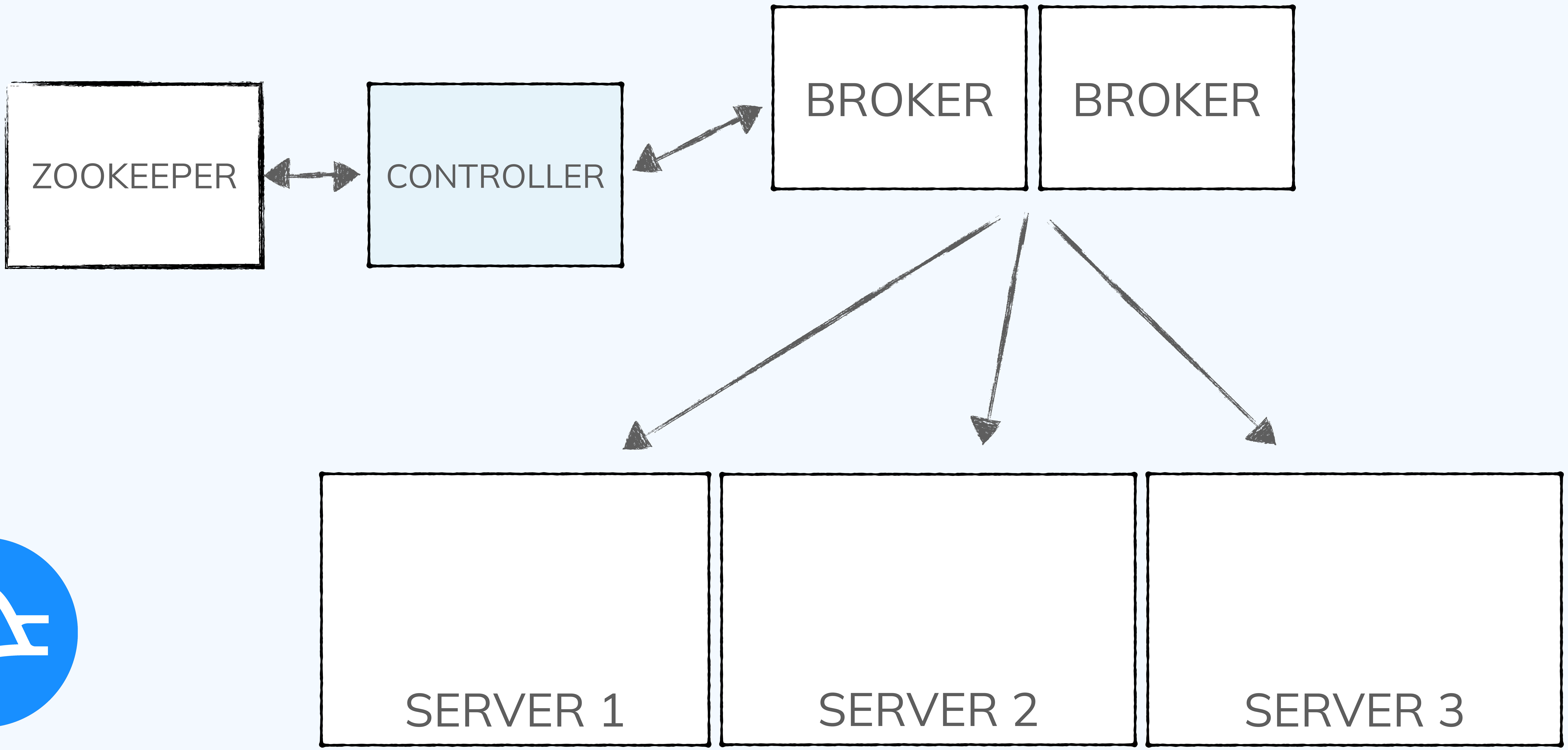


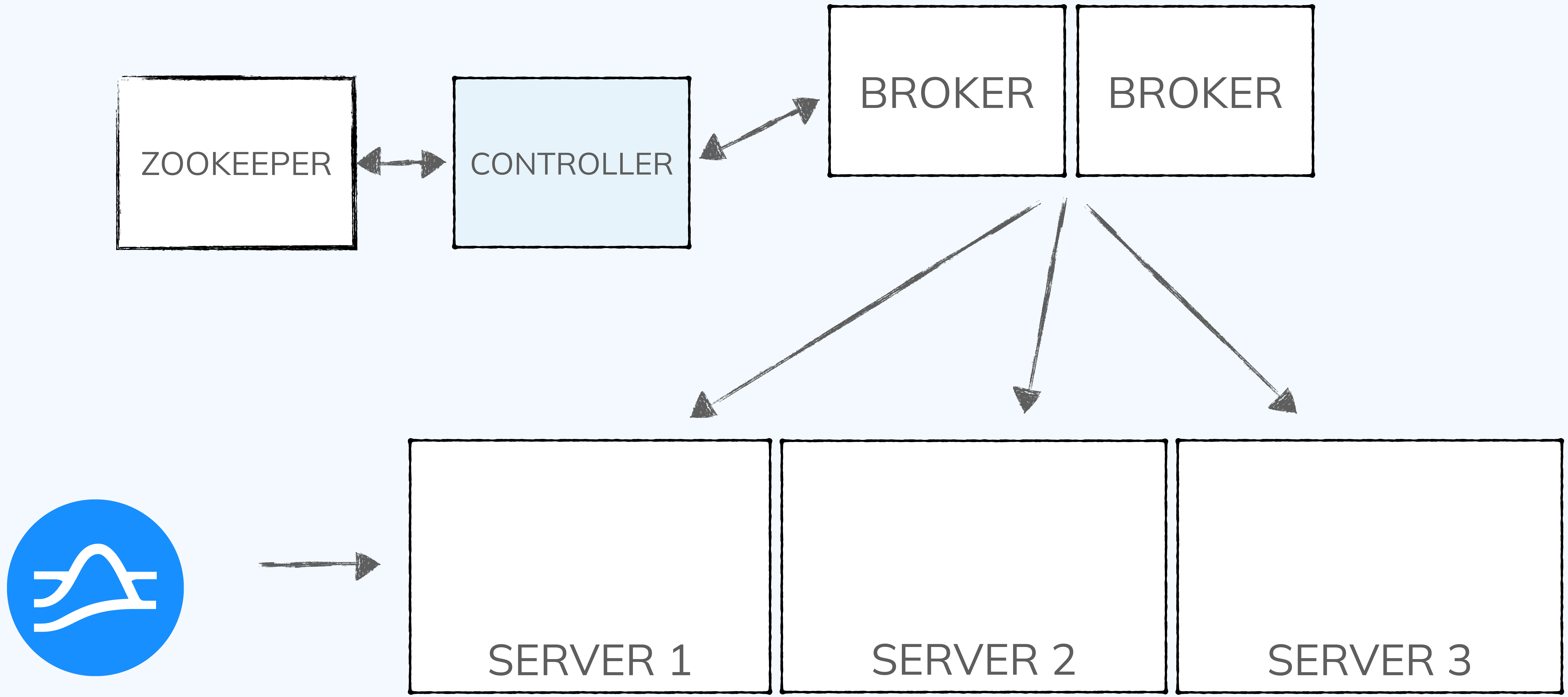
Exhibit 1

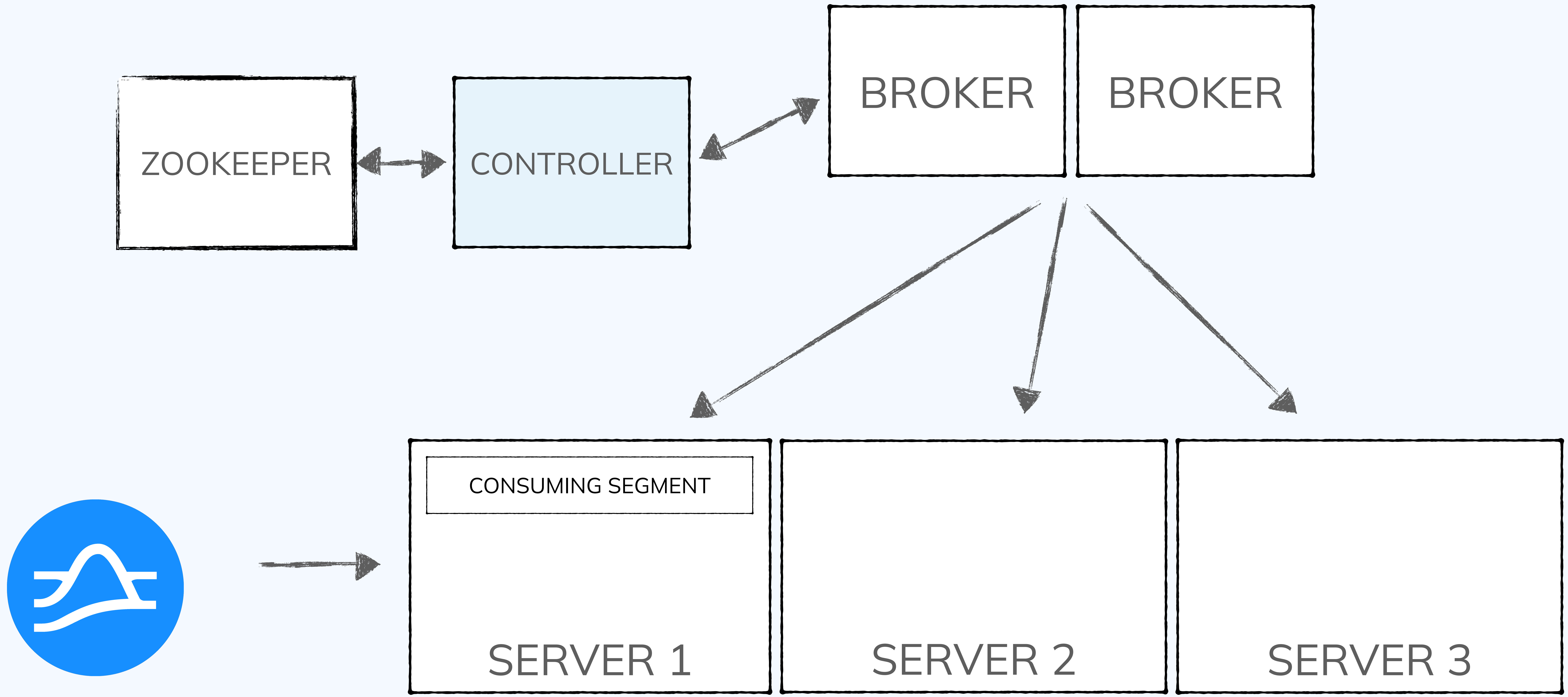
Part 2

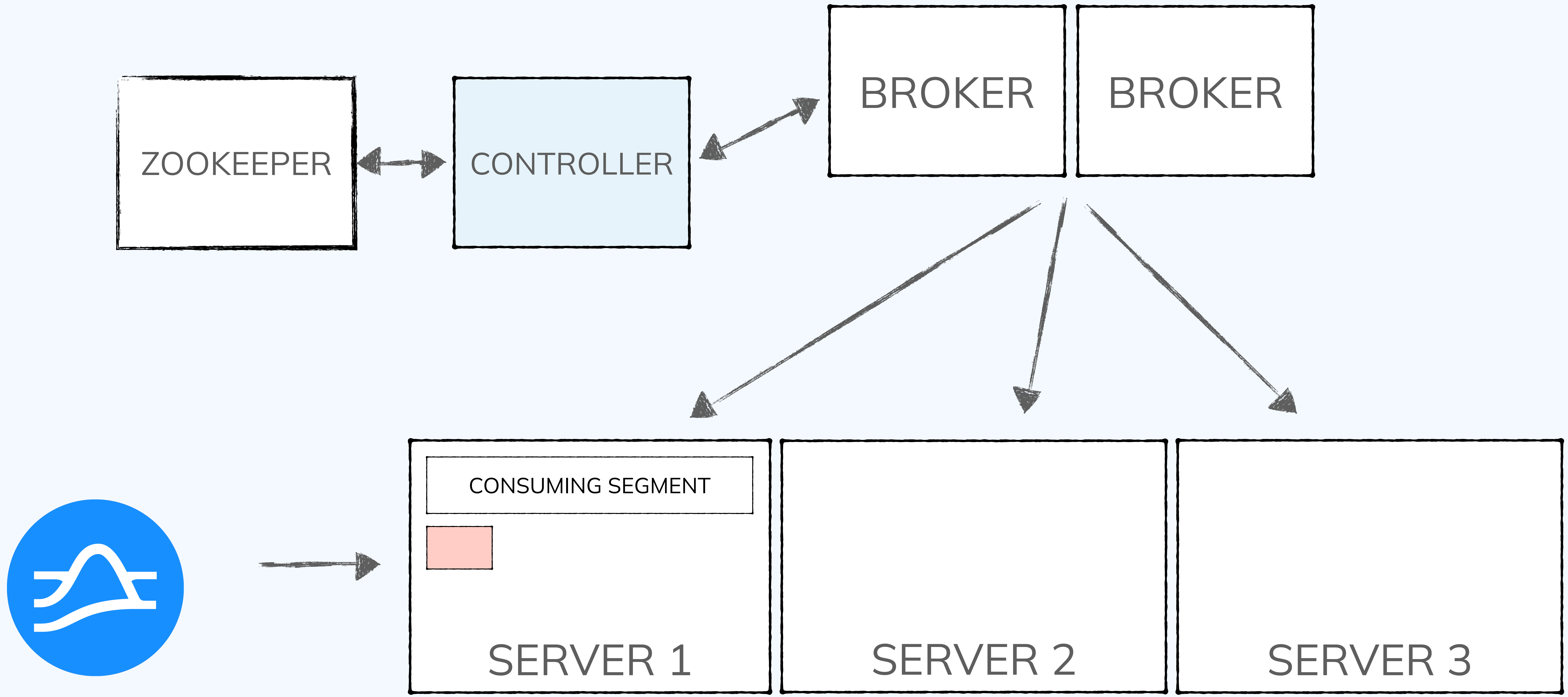
Streaming Ingestion with Kafka

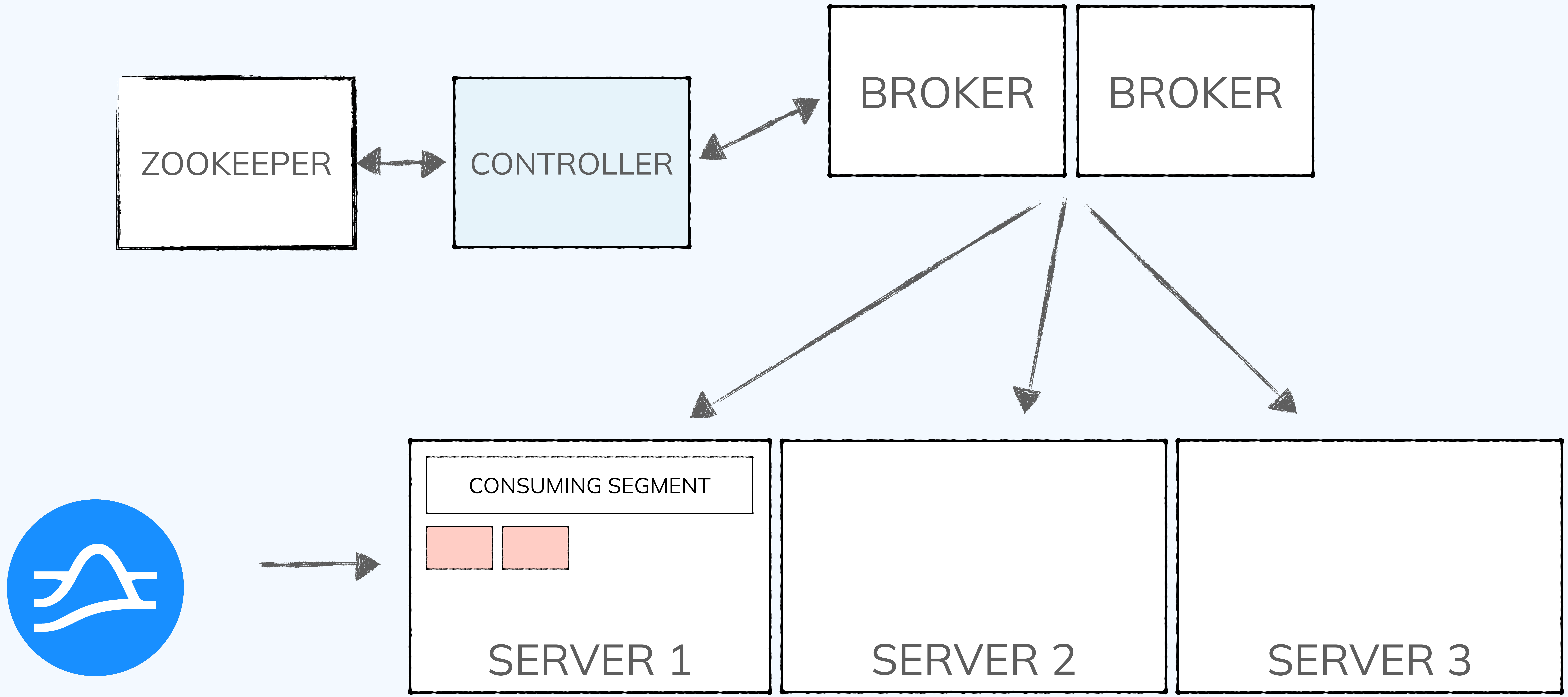
movies.jsonl
Offline Table Ingestion
ratings
Realitime Table Ingestion
Exhibit 2

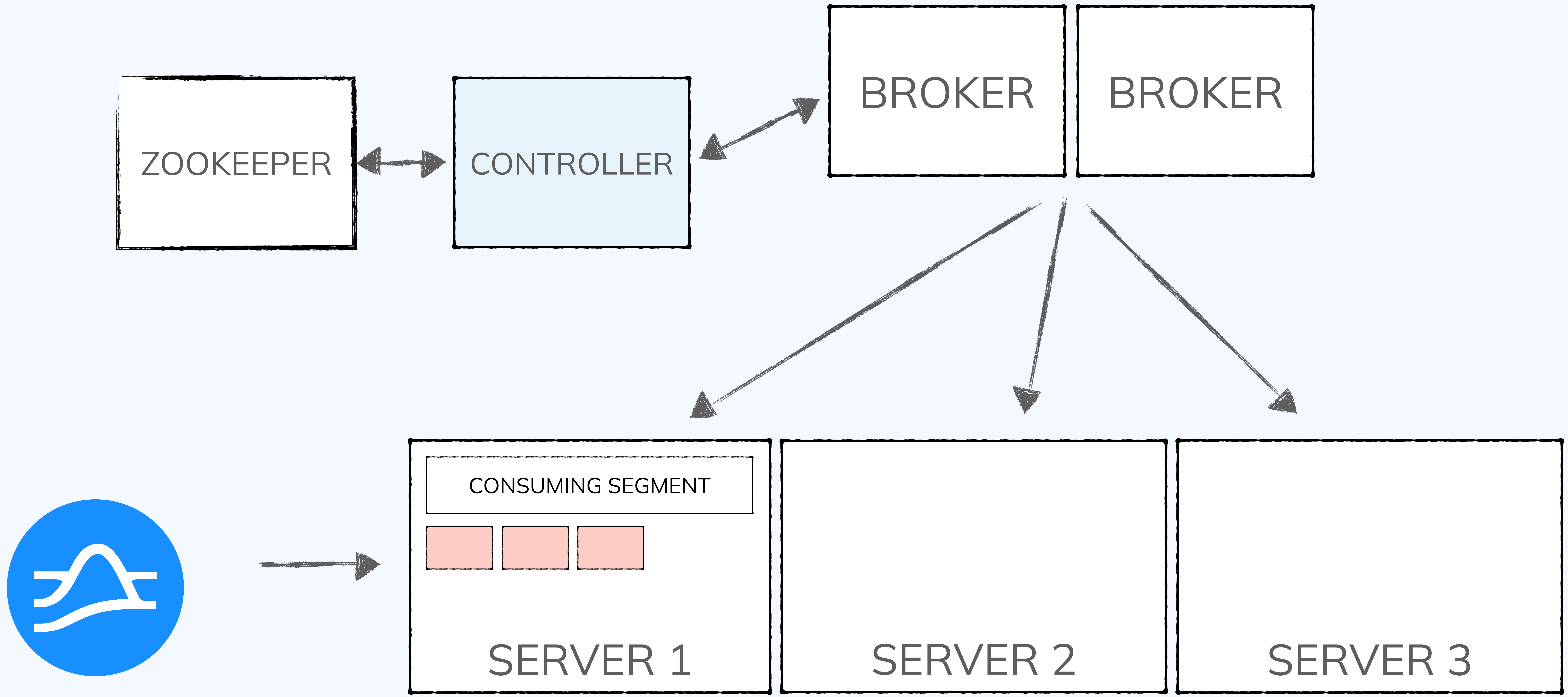

ZOOKEEPER
CONTROLLER
BROKER
BROKER
SERVER 1
SERVER 2
SERVER 3


ZOOKEEPER
CONTROLLER
BROKER
BROKER
SERVER 1
SERVER 2
SERVER 3


ZOOKEEPER
CONTROLLER
BROKER
BROKER
CONSUMING SEGMENT
SERVER 1
SERVER 2
SERVER 3


ZOOKEEPER
CONTROLLER
BROKER
BROKER
CONSUMING SEGMENT
SERVER 1
SERVER 2
SERVER 3


ZOOKEEPER
CONTROLLER
BROKER
BROKER
CONSUMING SEGMENT
SERVER 1
SERVER 2
SERVER 3


ZOOKEEPER
CONTROLLER
BROKER
BROKER
CONSUMING SEGMENT
SERVER 1
SERVER 2
SERVER 3

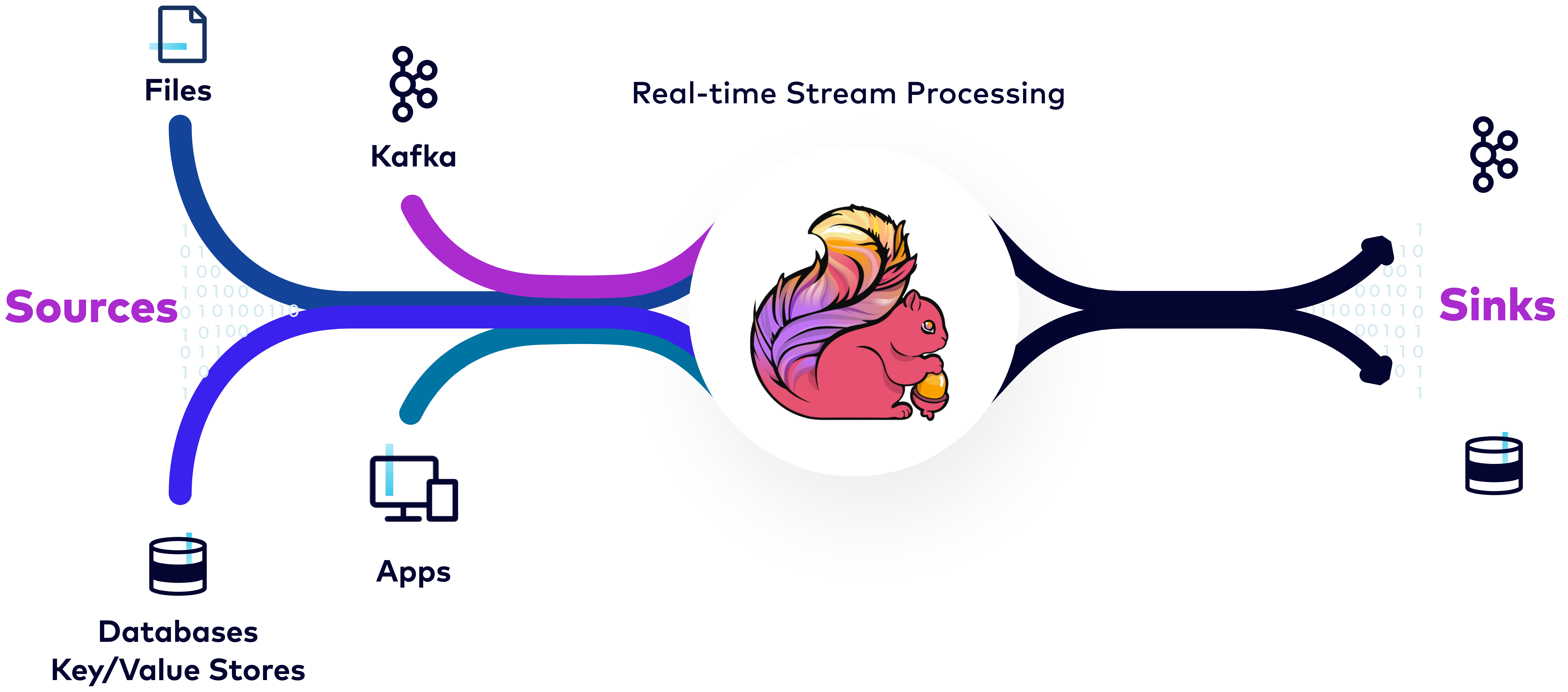
Part 3

Stream Join in Flink

Flink 101

«Apache Flink is a *framework* and *distributed processing engine* for *stateful* computations over *unbounded* and *bounded* data streams.»

Real-time services rely on stream processing



What is Flink SQL

A standards-compliant SQL engine for processing both batch and streaming data with the scalability, performance, and consistency of Apache Flink

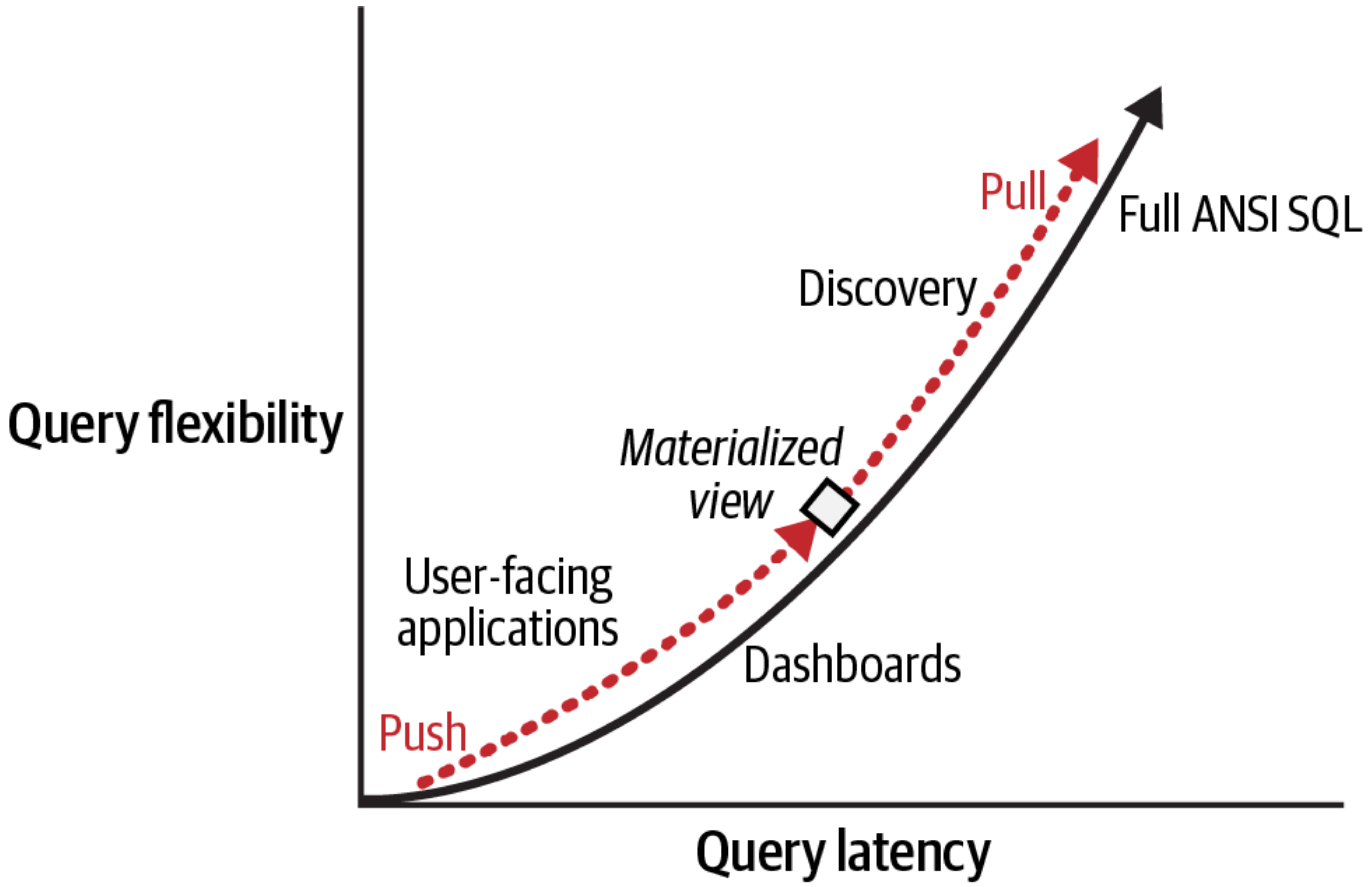
Is Flink SQL a database?

No. Bring your own data.

```
CREATE TABLE MovieRatings
(
    movieId           INT,
    rating DOUBLE,
    ratingTimeMillis BIGINT,
    ratingTime AS TO_TIMESTAMP_LTZ(ratingTimeMillis, 3)
) WITH (
      'connector' = 'pulsar',
      'topics' = 'persistent://public/default/ratings',
      'service-url' = 'pulsar://pulsar:6650',
      'value.format' = 'json',
      'source.subscription-name' = 'flink-ratings-
subscription',
      'source.subscription-type' = 'Shared'
      );
```

How does Flink work with Pulsar?

ratings
movies
rated_movies
Realtime Table Ingestion
Exhibit 3



Source: Streaming Databases, Hubert Dulay, Ralph Matthias Debusmann

Find the code of the demo 👉

https://gamov.dev/uncorking-analytics