How to Build a Scalable ETL Pipeline with Kafka

Alex

Traditional ETL

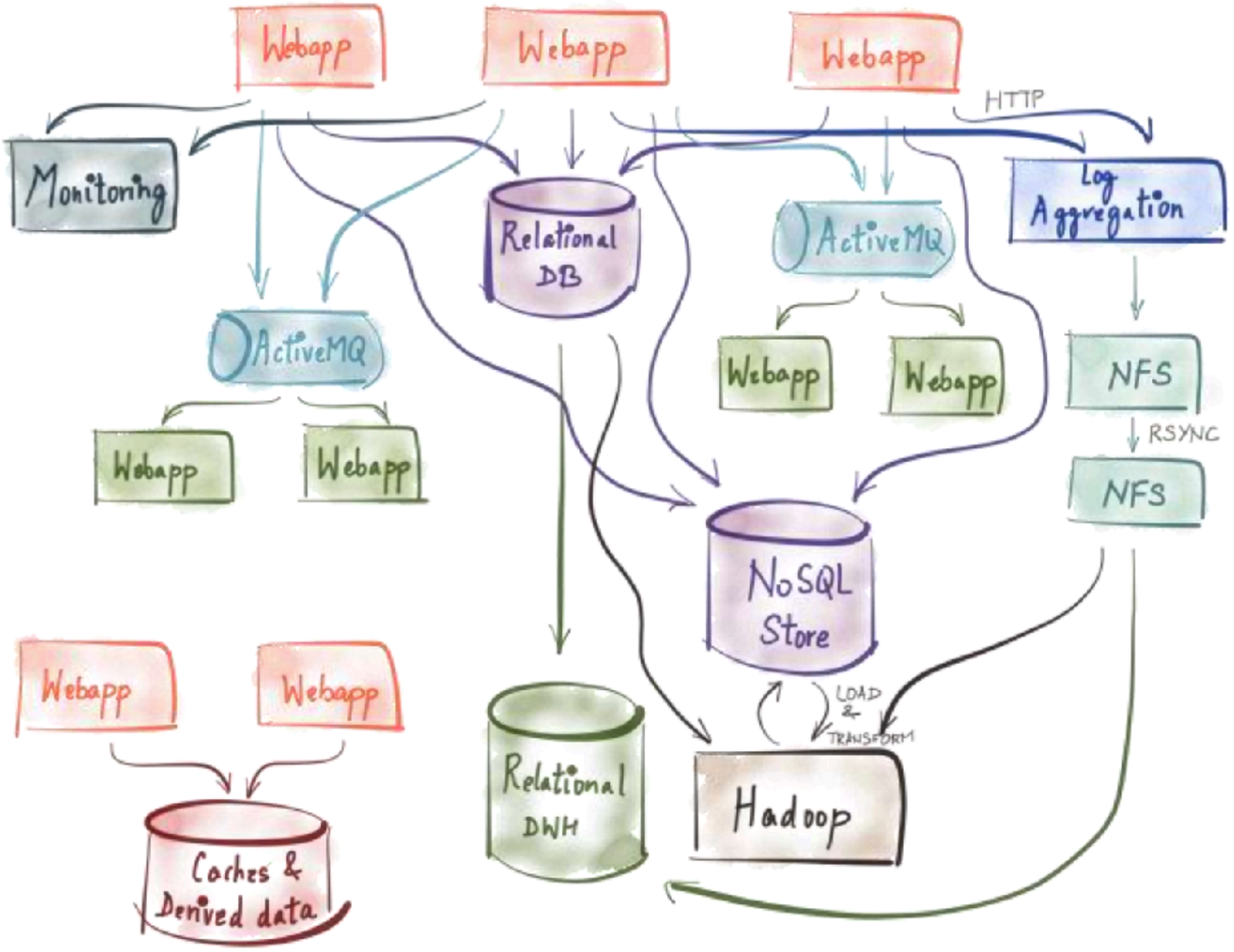
MySQL

ORACLE

RDBMS

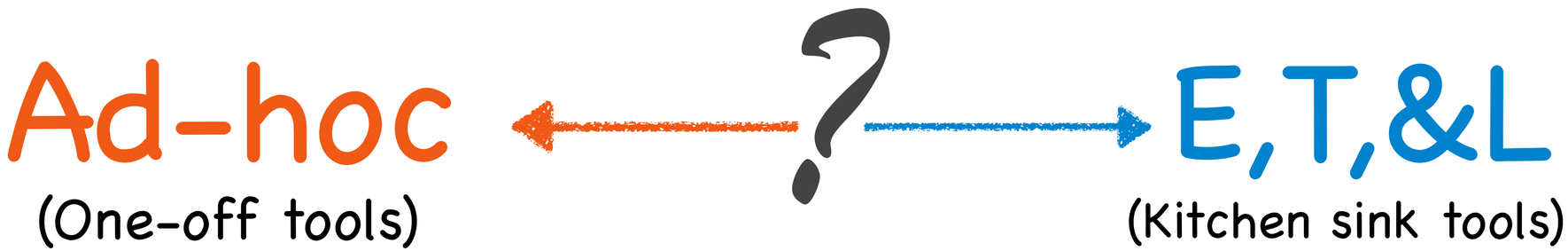
TERADATA


HIVE
cassandra
VOLTDB
APACHE HBASE
ORACLE DATABASE
KUDU
elasticsearch
Spark
EROSPIKE
OrientDB
monetdb
OMNITURE
riak
Solr
memsql
Couchbase
neo4j
RethinkDB
LOG
mongoDB
MQTT.ORG
Scalaris
APACHE GIRAPH
Flink
confluent
ArangoDB


Webapp
Webapp
Webapp
HTTP
Monitoring
Relational DB
ActiveMQ
Log Aggregation
ActiveMQ
Webapp
Webapp
Webapp
Webapp
NFS
RSYNC
NFS
NoSQL Store
Webapp
Webapp
LOAD & TRANSFORM
Relational DWH
Hadoop
Caches & Derived data

ANTI PATTERENS

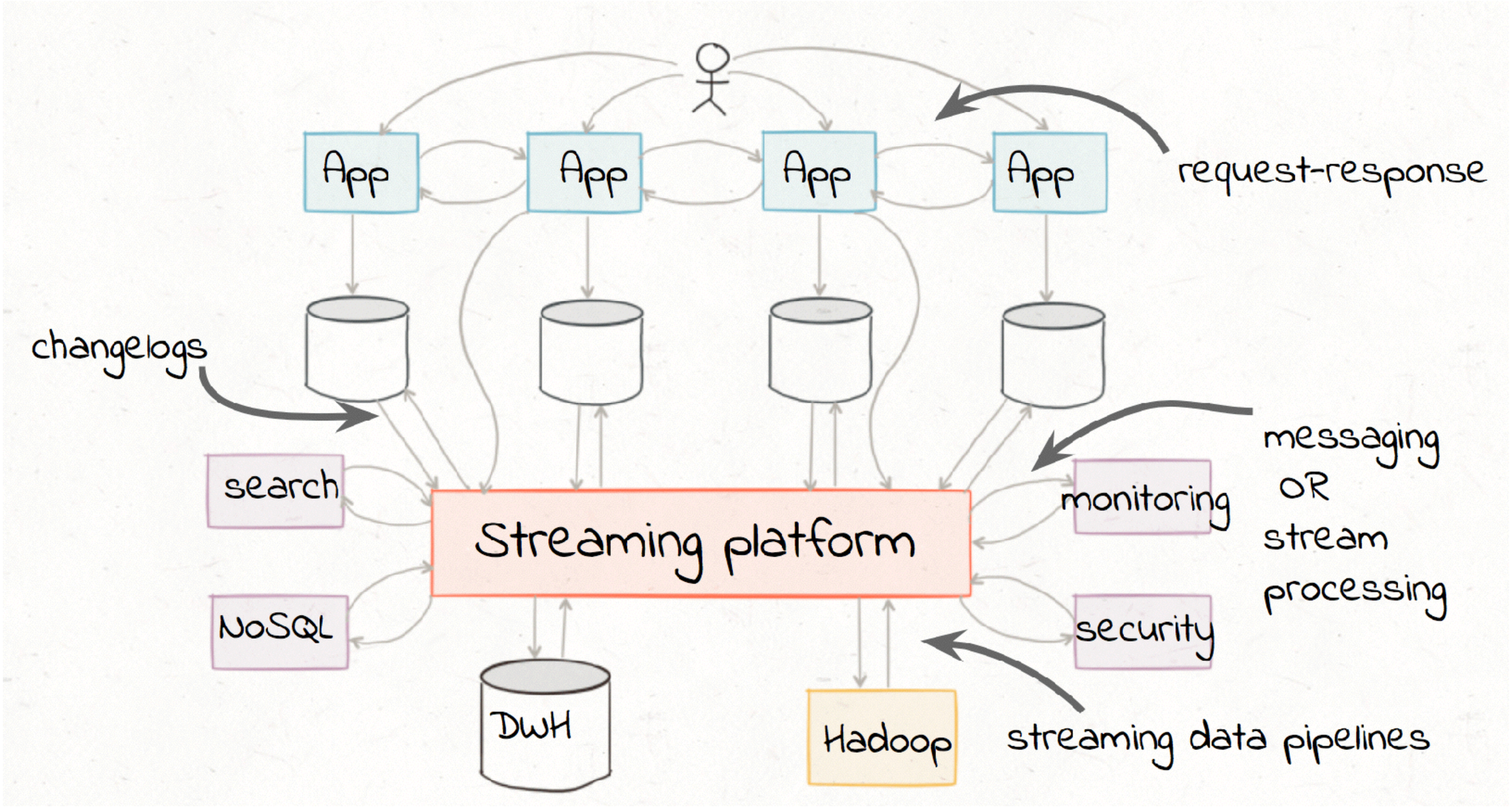
1. One-off tools
2. Kitchen sink tools
3. Streaming processing frameworks


Ad-hoc
(One-off tools)
?
E,T,&L
(Kitchen sink tools)

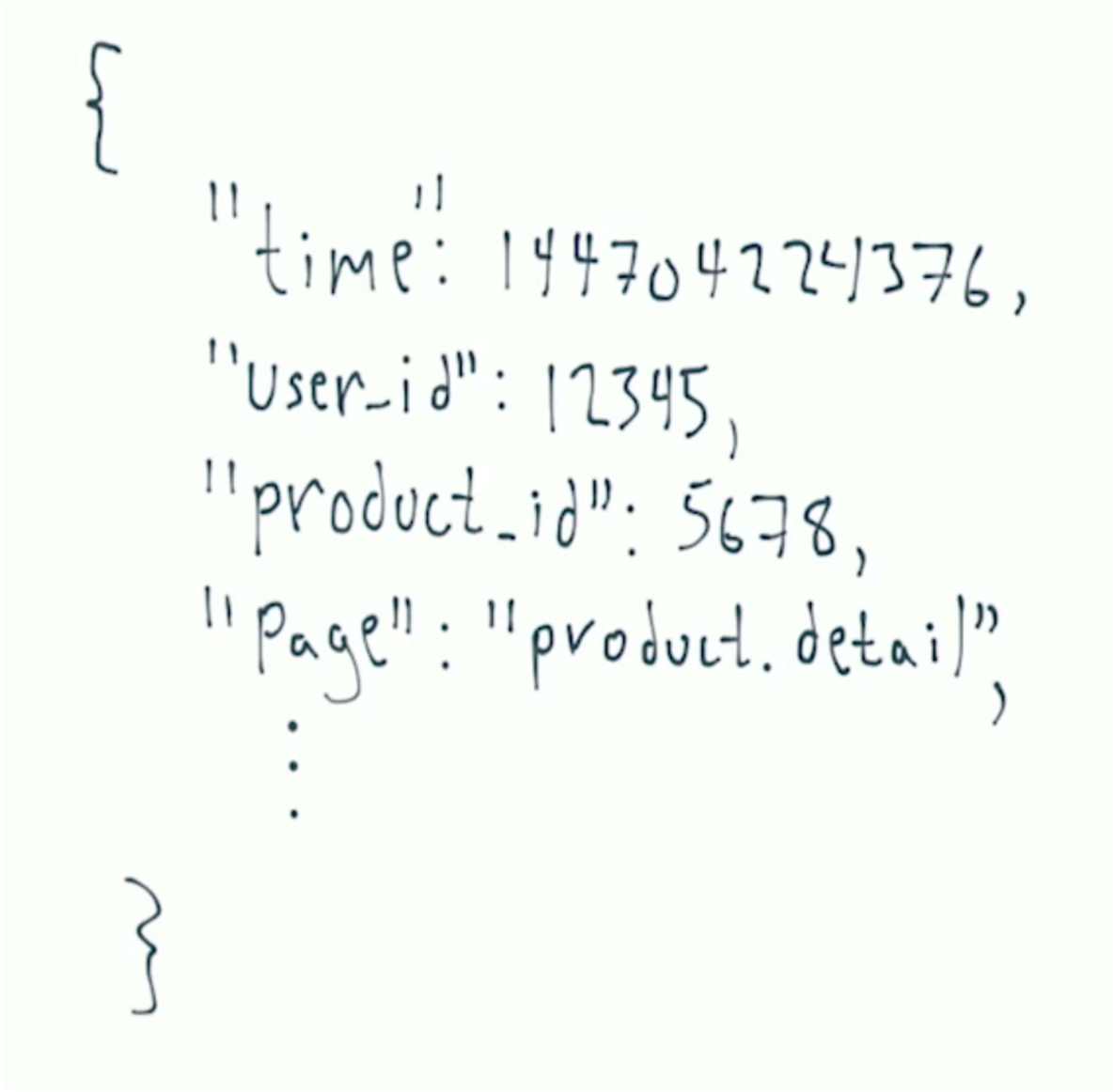
Data Integration

getting data to all the right places

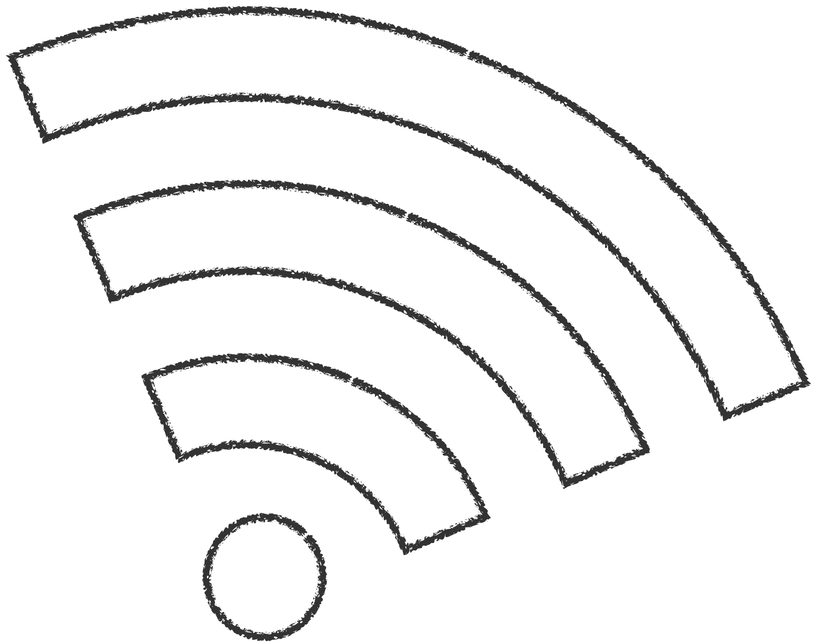
All your data is event streams


App
App
App
App
request-response
changelogs
search
Streaming platform
monitoring
messaging
OR
stream
processing
NoSQL
security
DWH
Hadoop
streaming data pipelines

EXAMPLE EVENT: PRODUCT VIEW



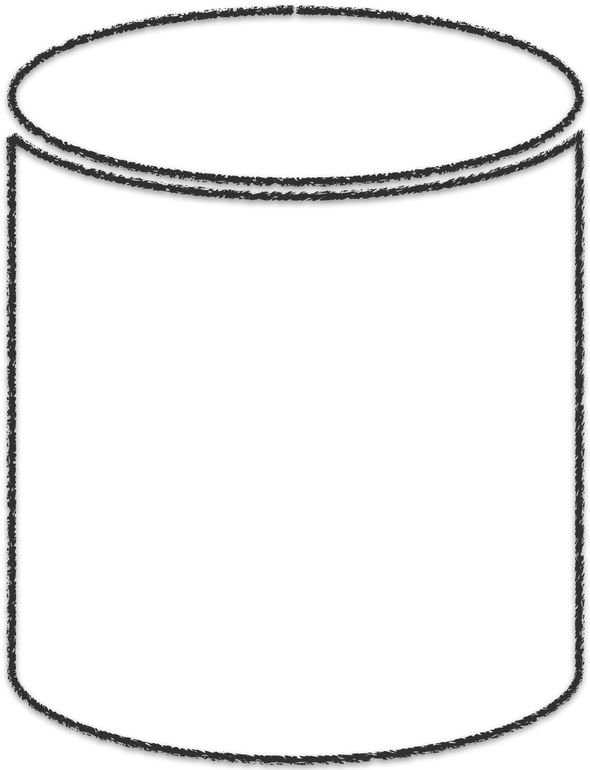
EXAMPLE EVENT: SENSORS



EXAMPLE EVENT: LOG FILES

```
jkreps-mn:~ jkreps$ tail -f -n 20 /var/log/apache2/access_log
::1 - - [23/Mar/2014:15:07:00 -0700] "GET /images/apache_feather.gif HTTP/1.1" 200 4128
::1 - - [23/Mar/2014:15:07:04 -0700] "GET /images/producer_consumer.png HTTP/1.1" 200 8(
::1 - - [23/Mar/2014:15:07:04 -0700] "GET /images/log_anatomy.png HTTP/1.1" 200 19579
::1 - - [23/Mar/2014:15:07:04 -0700] "GET /images/consumer-groups.png HTTP/1.1" 200 268
::1 - - [23/Mar/2014:15:07:04 -0700] "GET /images/log_compaction.png HTTP/1.1" 200 4141
::1 - - [23/Mar/2014:15:07:04 -0700] "GET /documentation.html HTTP/1.1" 200 189893
::1 - - [23/Mar/2014:15:07:04 -0700] "GET /images/log_cleaner_anatomy.png HTTP/1.1" 200
::1 - - [23/Mar/2014:15:07:04 -0700] "GET /images/kafka_log.png HTTP/1.1" 200 134321
::1 - - [23/Mar/2014:15:07:04 -0700] "GET /images/mirror-maker.png HTTP/1.1" 200 17054
::1 - - [23/Mar/2014:15:08:07 -0700] "GET /documentation.html HTTP/1.1" 200 189937
::1 - - [23/Mar/2014:15:08:07 -0700] "GET /styles.css HTTP/1.1" 304 -
::1 - - [23/Mar/2014:15:08:07 -0700] "GET /images/kafka_logo.png HTTP/1.1" 304 -
::1 - - [23/Mar/2014:15:08:07 -0700] "GET /images/producer_consumer.png HTTP/1.1" 304 -
::1 - - [23/Mar/2014:15:08:07 -0700] "GET /images/log_anatomy.png HTTP/1.1" 304 -
::1 - - [23/Mar/2014:15:08:07 -0700] "GET /images/consumer-groups.png HTTP/1.1" 304 -
::1 - - [23/Mar/2014:15:08:07 -0700] "GET /images/log_cleaner_anatomy.png HTTP/1.1" 304
::1 - - [23/Mar/2014:15:08:07 -0700] "GET /images/log_compaction.png HTTP/1.1" 304 -
::1 - - [23/Mar/2014:15:08:07 -0700] "GET /images/kafka_log.png HTTP/1.1" 304 -
::1 - - [23/Mar/2014:15:09:55 -0700] "GET /images/mirror-maker.png HTTP/1.1" 304 -
                                     "GET /documentation.html HTTP/1.1" 200 195264
```

EXAMPLE EVENT: DATABASES



What is a table?

key	value
key1	value2
key2	value3

EXAMPLE EVENT: DATABASES

key	value
key1	value1

Time = 0

put(key1, value1)

key	value
key1	value2
key2	value2

Time = 1

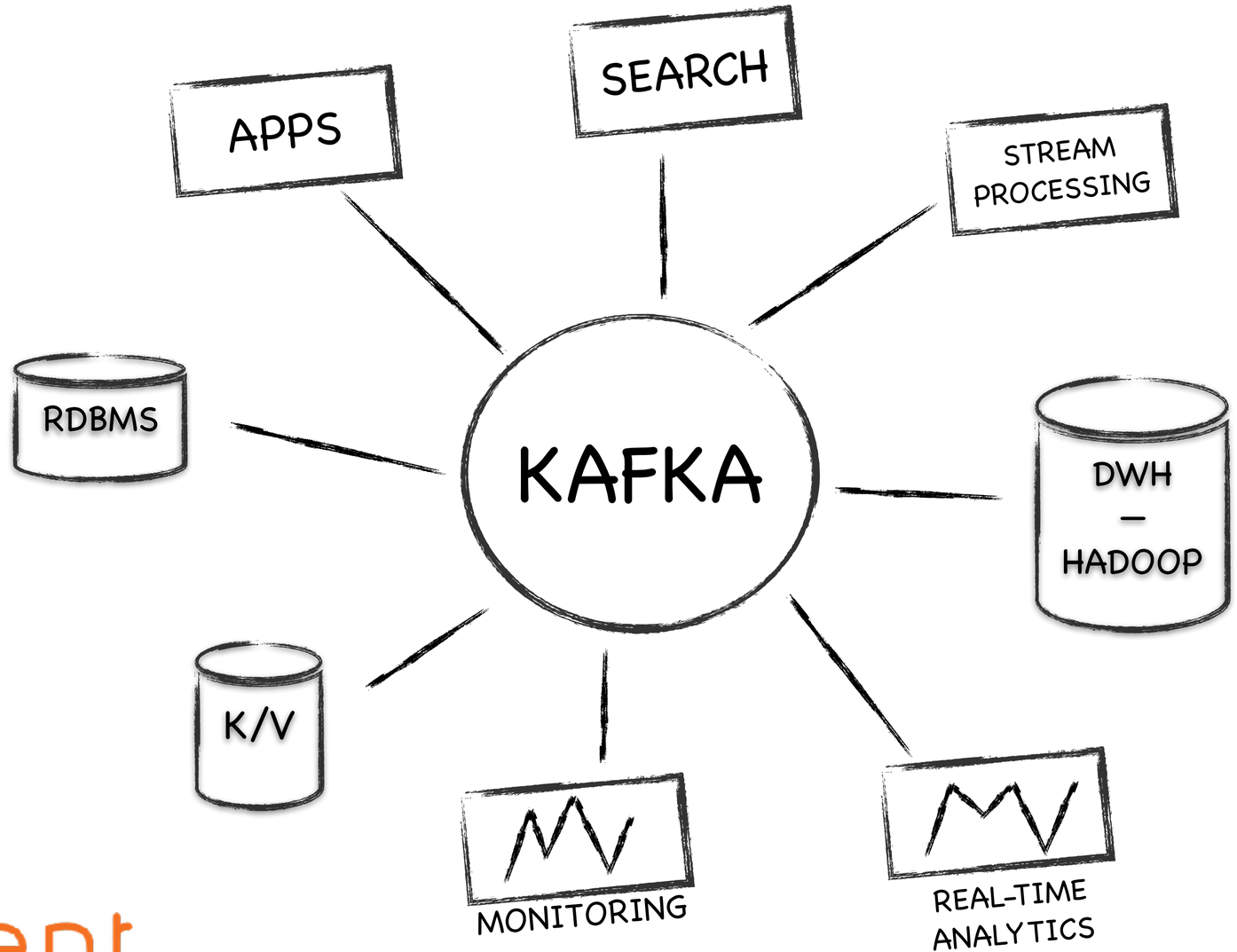
put(key2, value2)

key	value
key1	value3
key2	value2

Time = 2

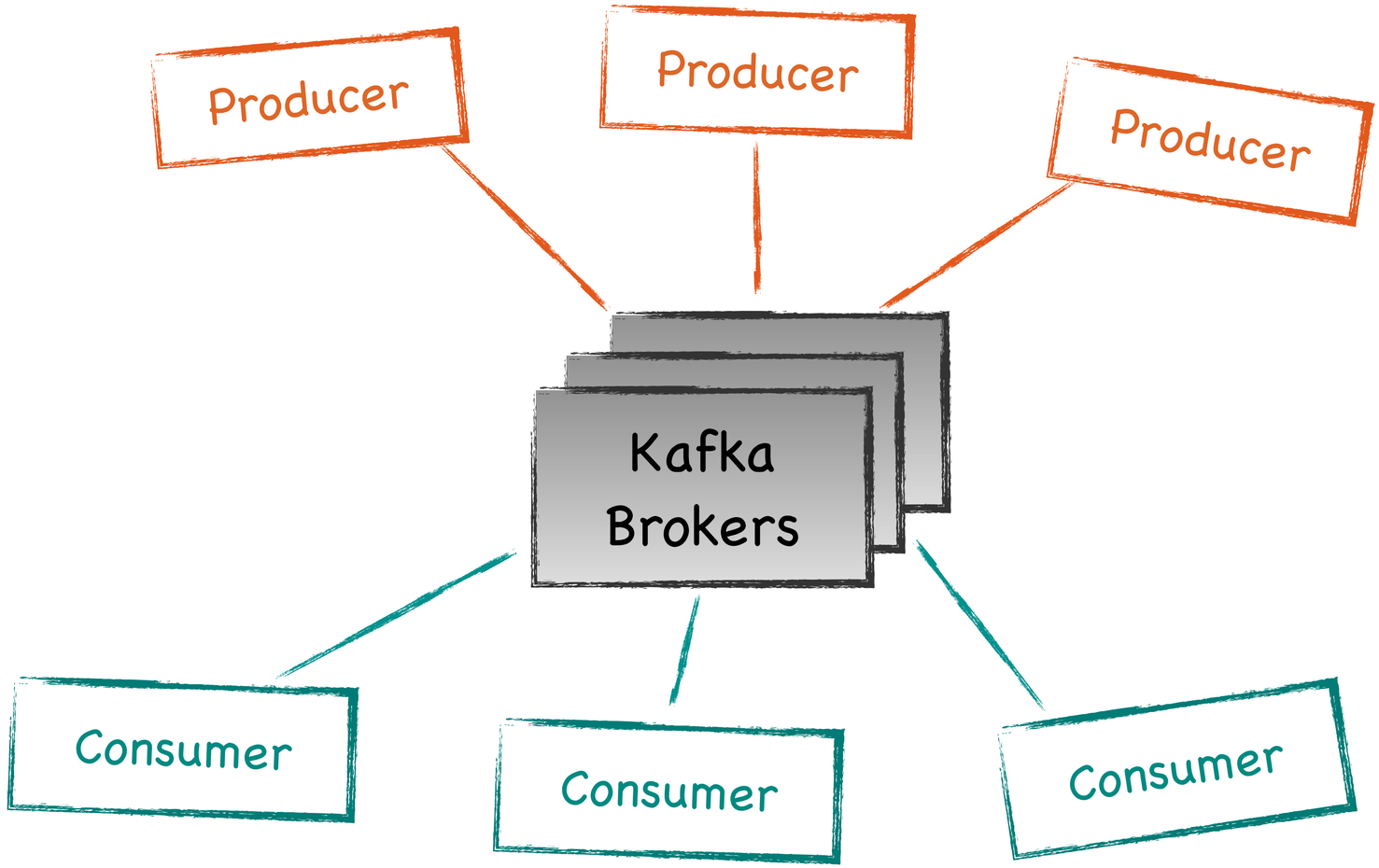
put(key1, value3)

STREAMING PLATFORM

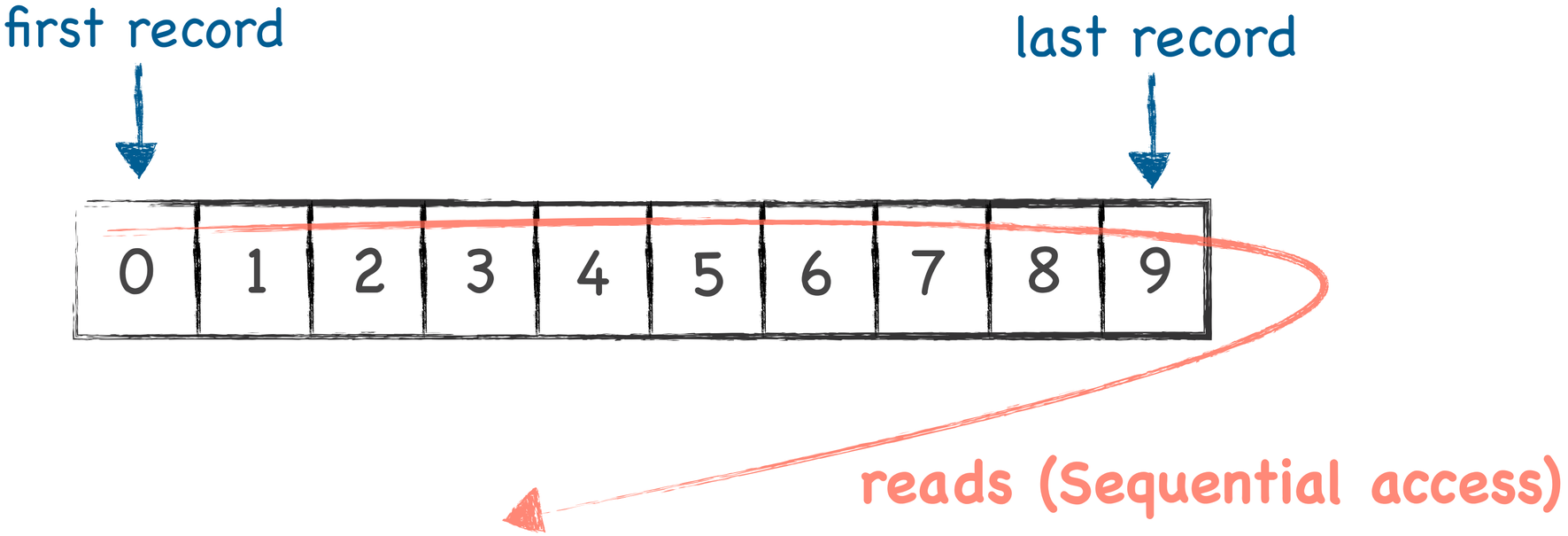


confluent

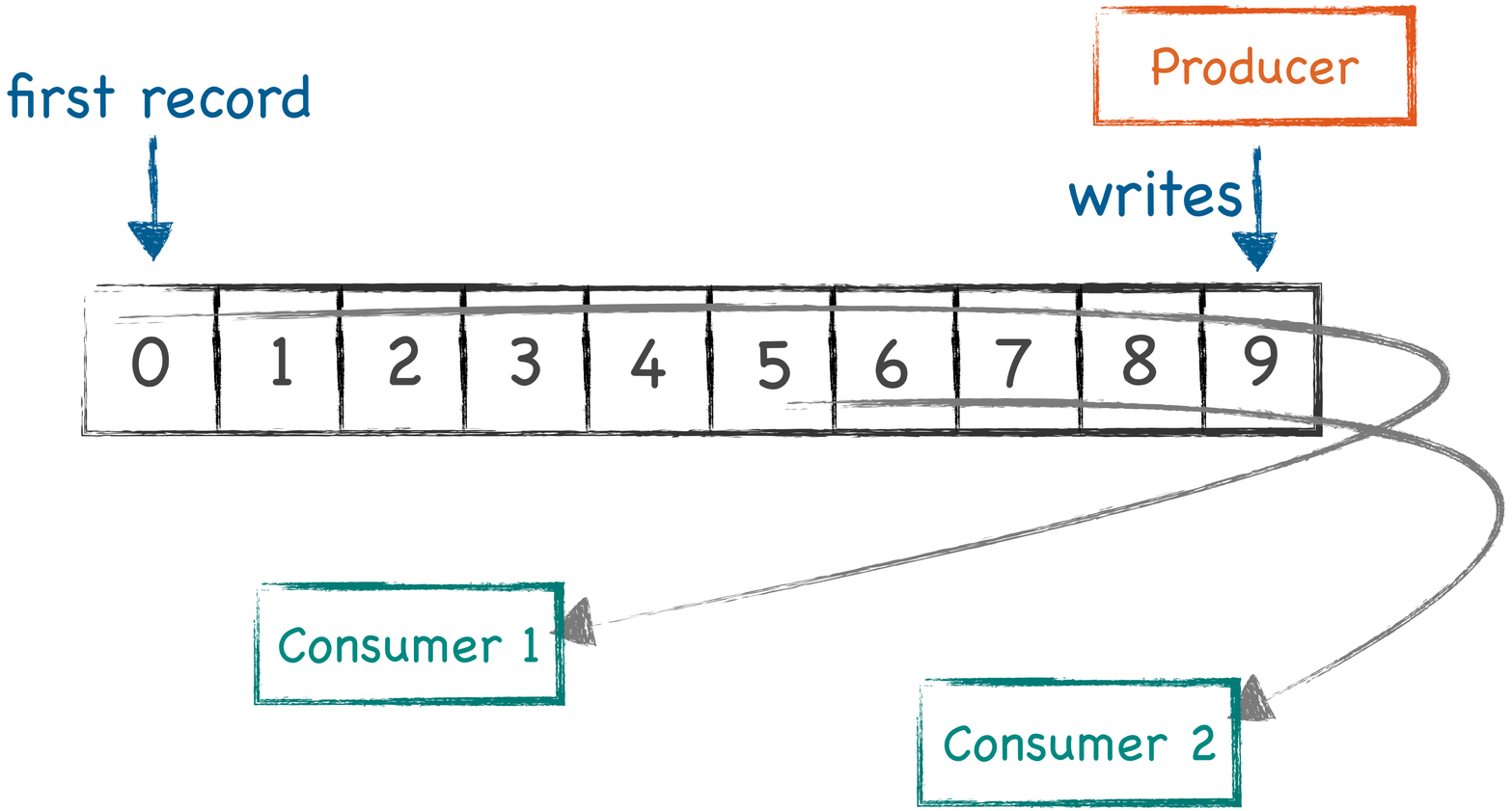
APACHE KAFKA:



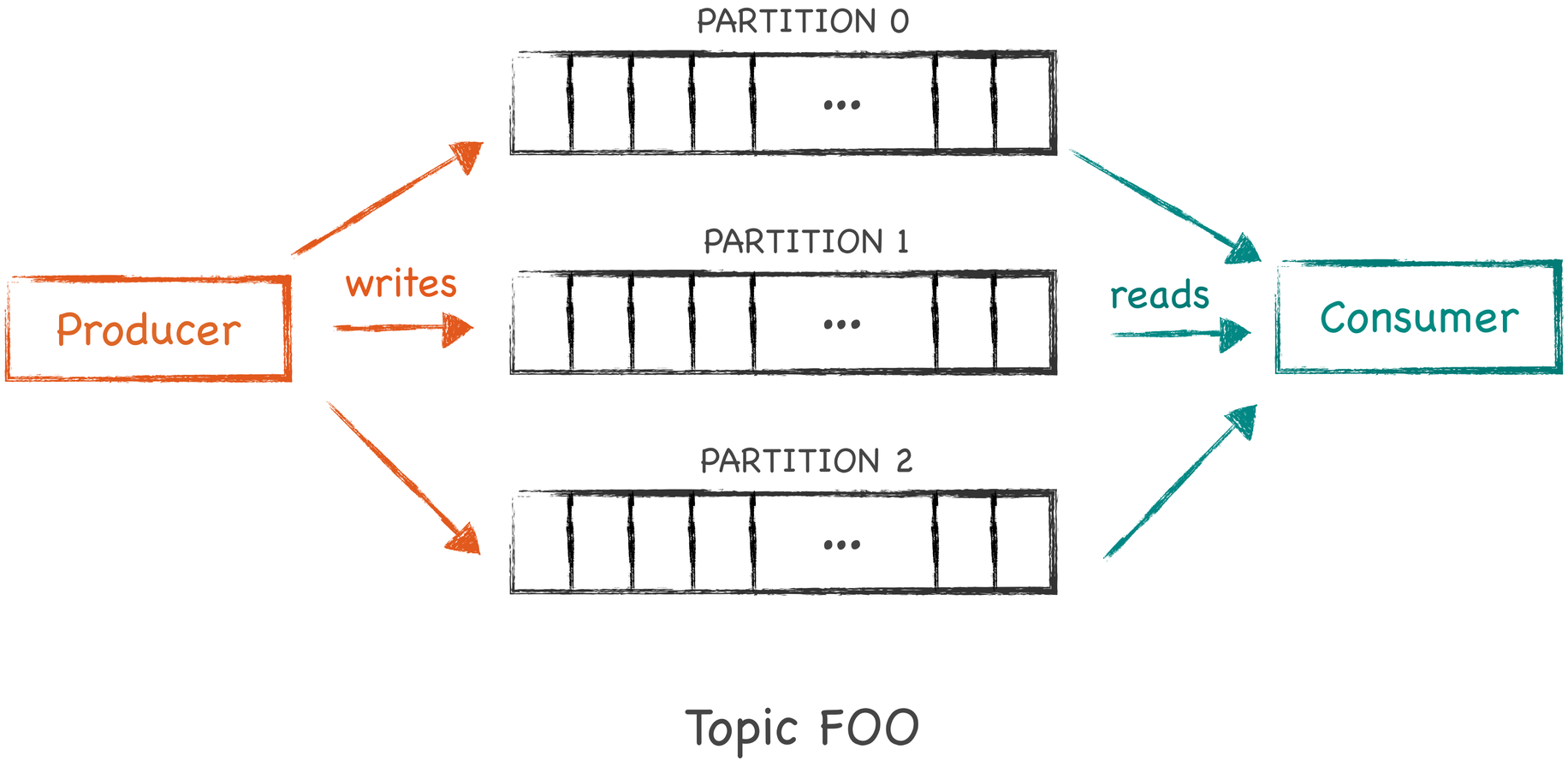
APACHE KAFKA: Key idea(Log)



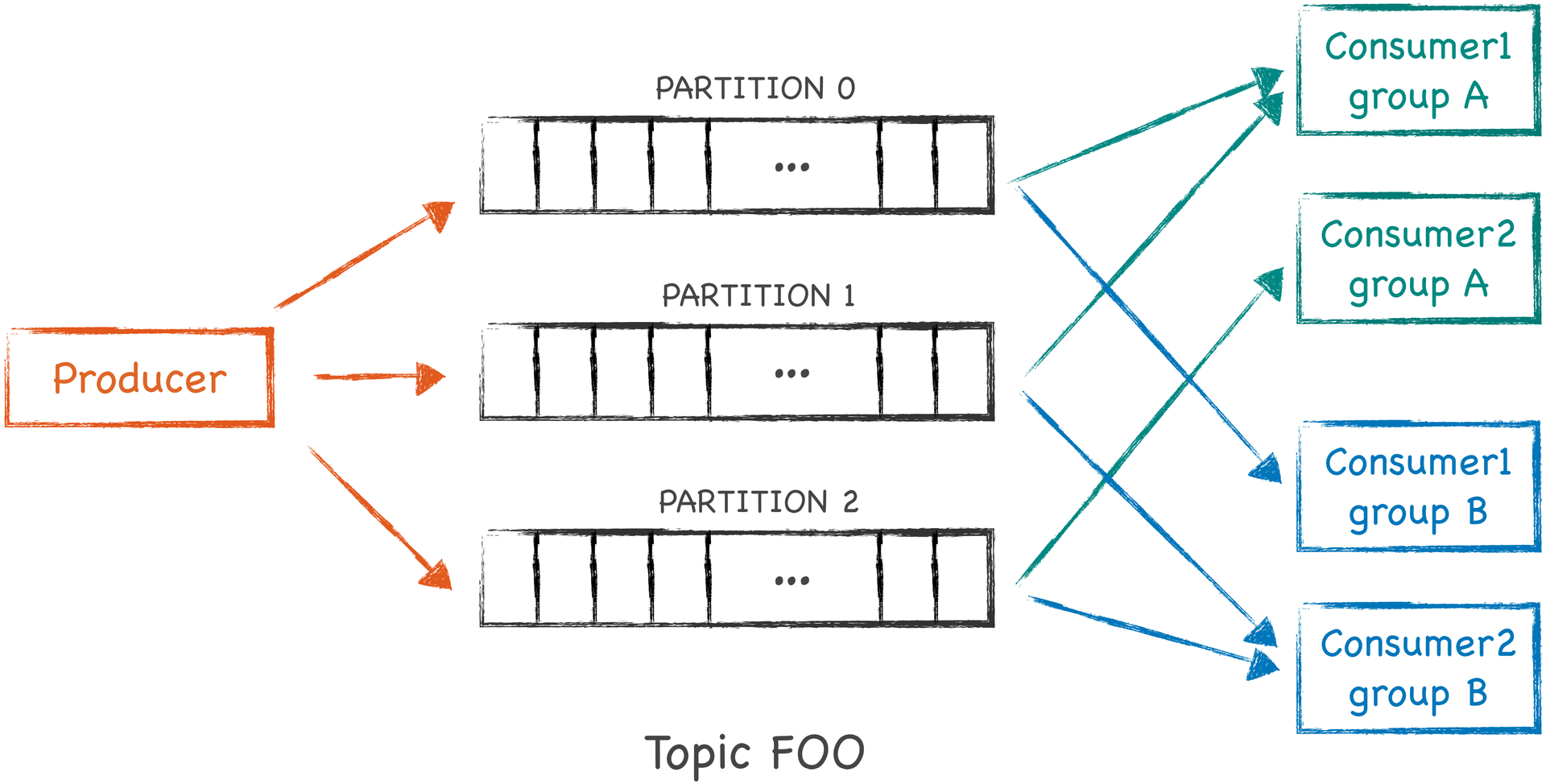
APACHE KAFKA: Logs & Pub-Sub



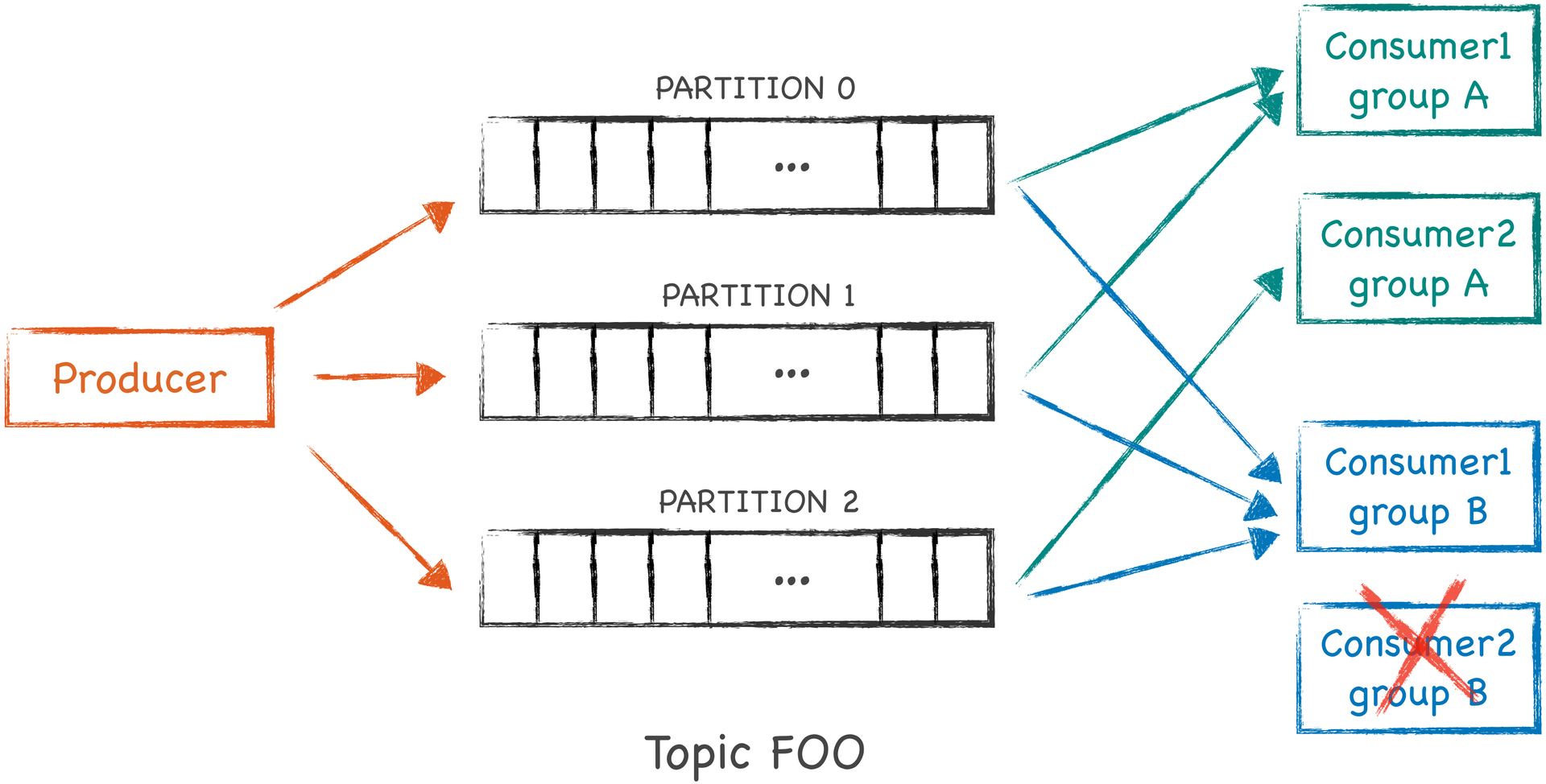
APACHE KAFKA: Topic = partitioned log



APACHE KAFKA: Scalable Consumption

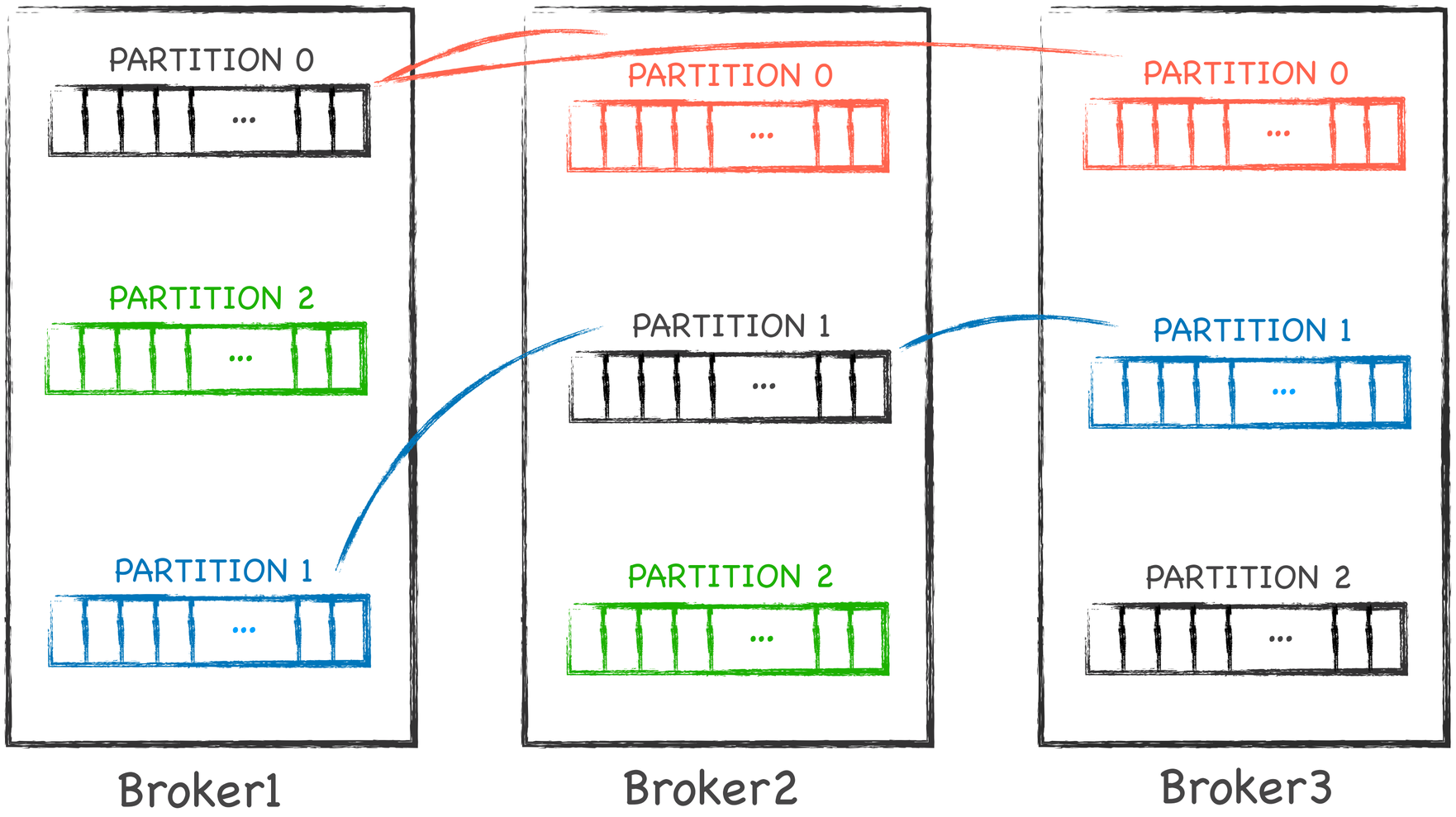


APACHE KAFKA: Fault Tolerant Consumption



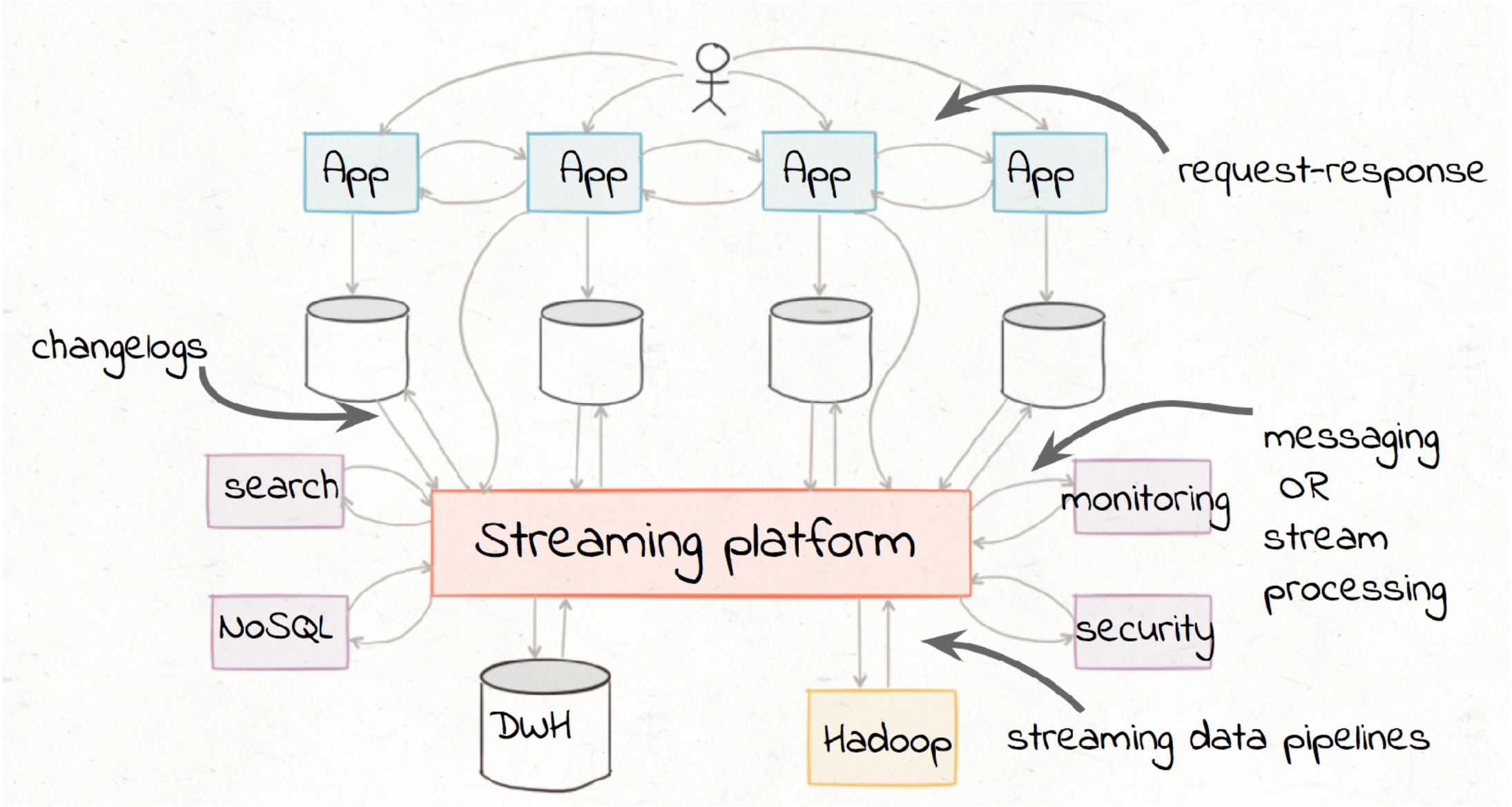
APACHE KAFKA: Partition & Replication

Topic FOO



KAFKA CONNECT

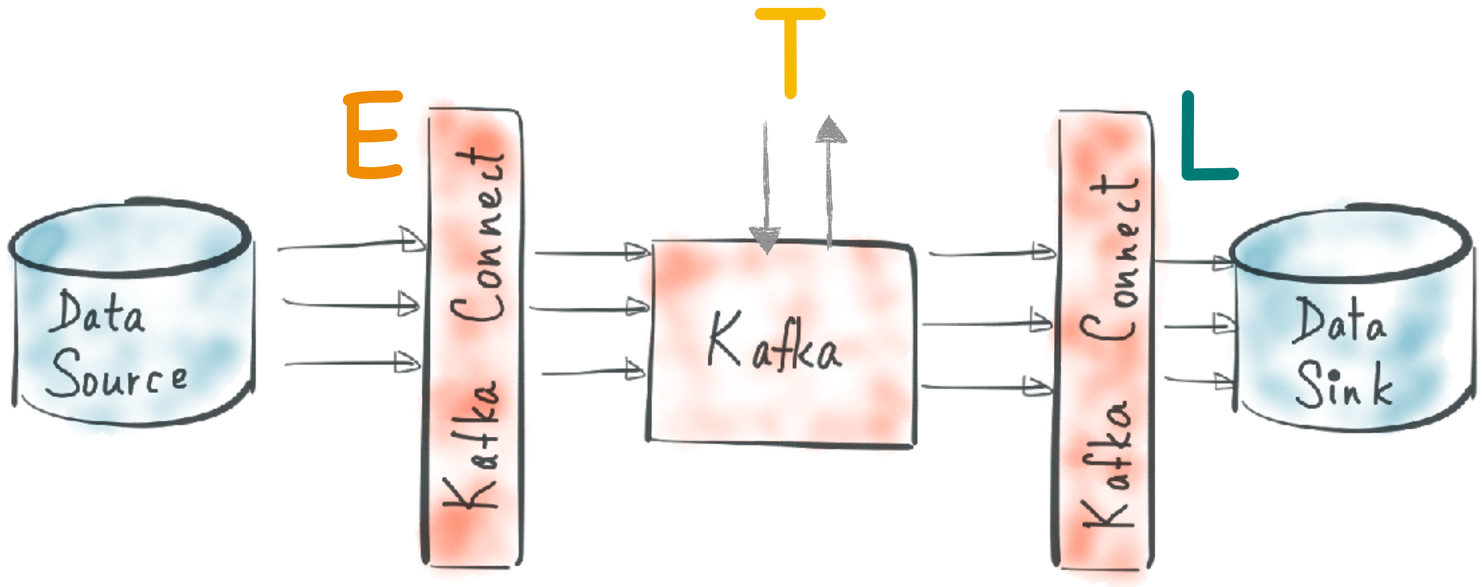
Large-scale streaming data import/export for Kafka


App
App
App
App
request-response
changelogs
search
Streaming platform
monitoring
messaging
OR
stream
processing
NoSQL
security
DWH
Hadoop
streaming data pipelines

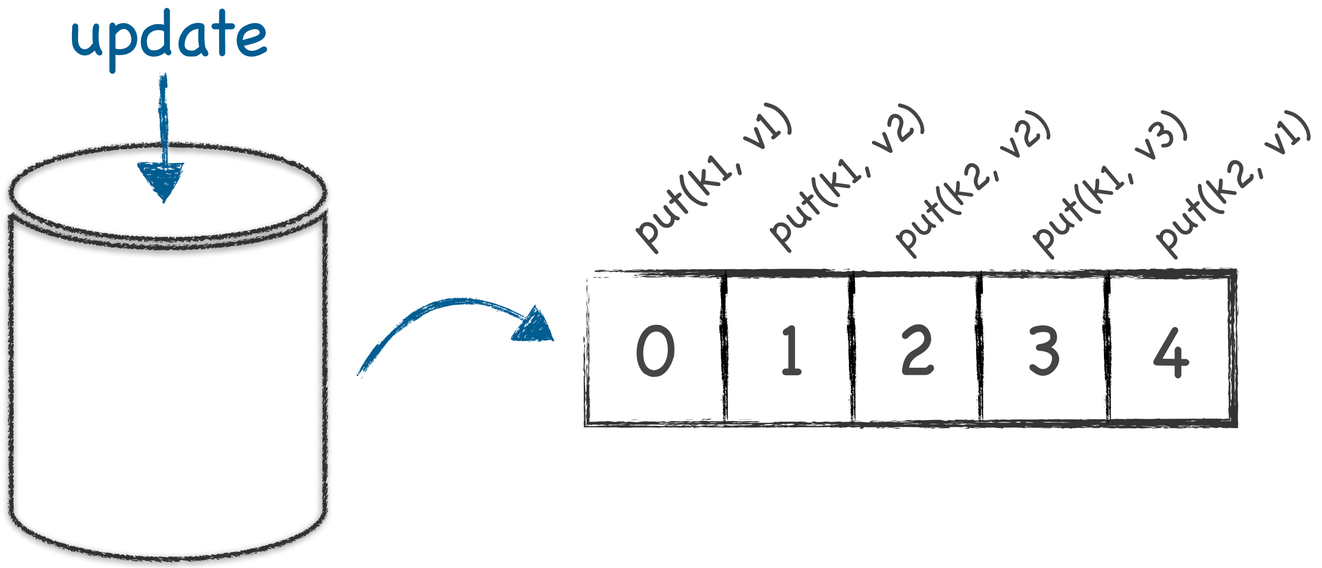
KAFKA CONNECT

GOAL

1. Focus on copying
2. Batteries included
3. Parallelism
4. Scale



Database change events/logs

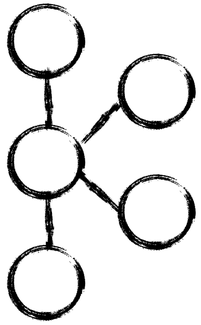


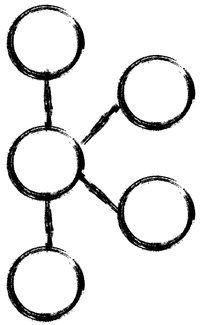
Table

TS	DATA
12:00	A
12:20	B
12:30	C
13:00	D
13:05	E

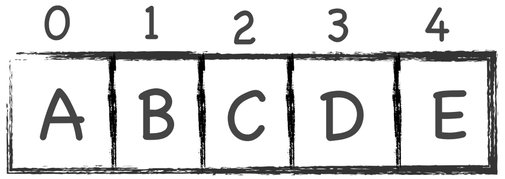
Database Source (connector)

offset = timestamp



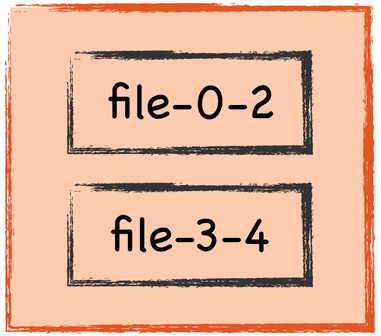


HDFS Sink (connector)

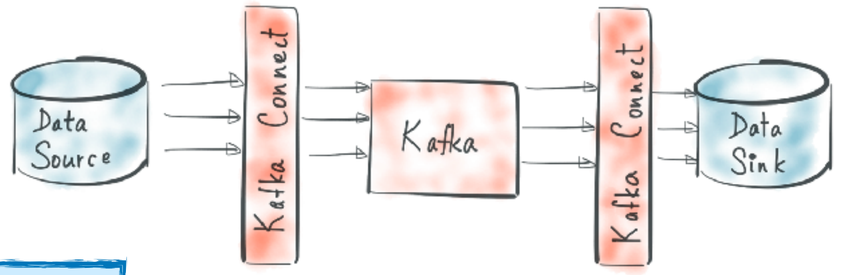


offset = kafka offset

HDFS Directory



Data Model



Partitioned Stream - DataBase

Partition 1 (Table 1)

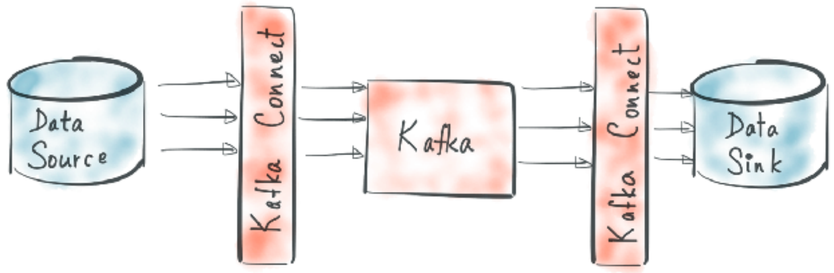
Partition 2 (Table 2)

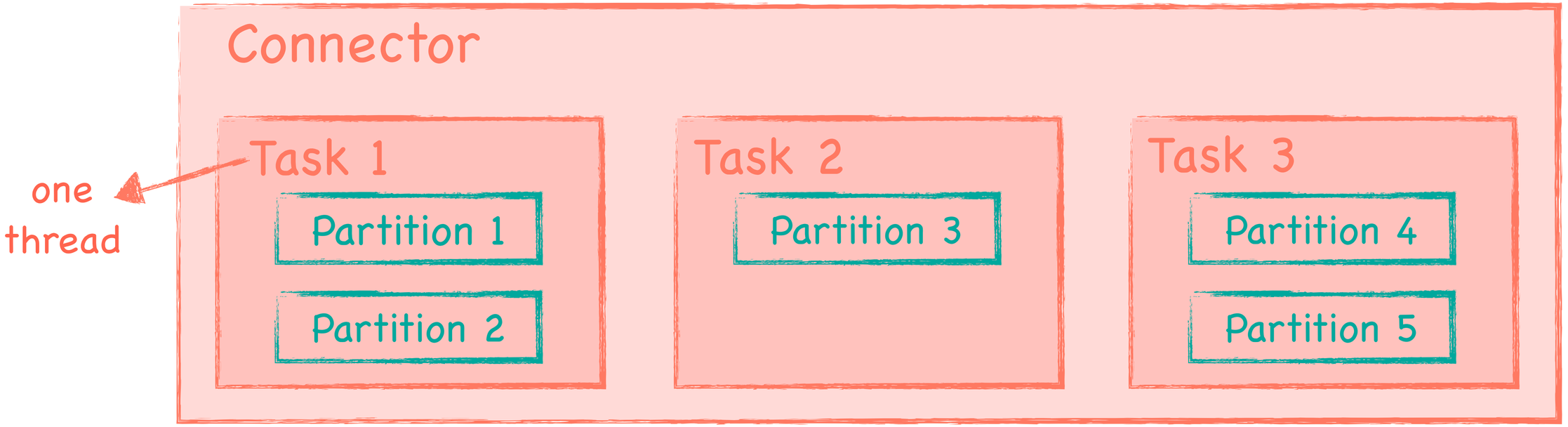
Partition 3 (Table 3)

Partition 4 (Table 4)

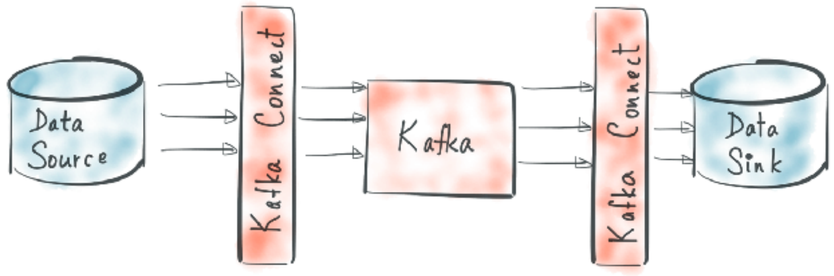
id=1 id=2 id=3 id=4 id=5 id=6 id=7 id=8

Parallelism Model





standalone & Distributed



Kafka connect (worker1)

Conn-1
Conn-2
Conn-1, task1
Conn-2, task1
Conn-1, task2
Conn-2, task2
Conn-1, task3
Conn-2, task3

worker1

Conn-1
Conn-1, task1

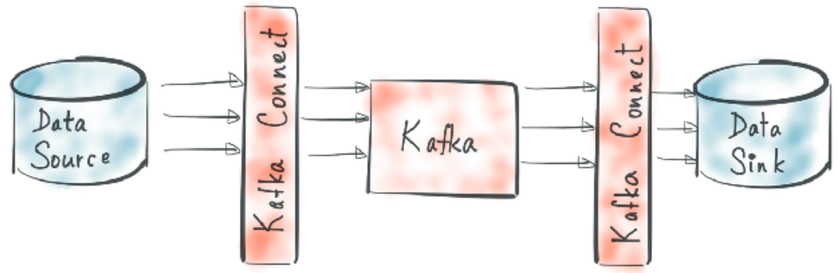
worker2

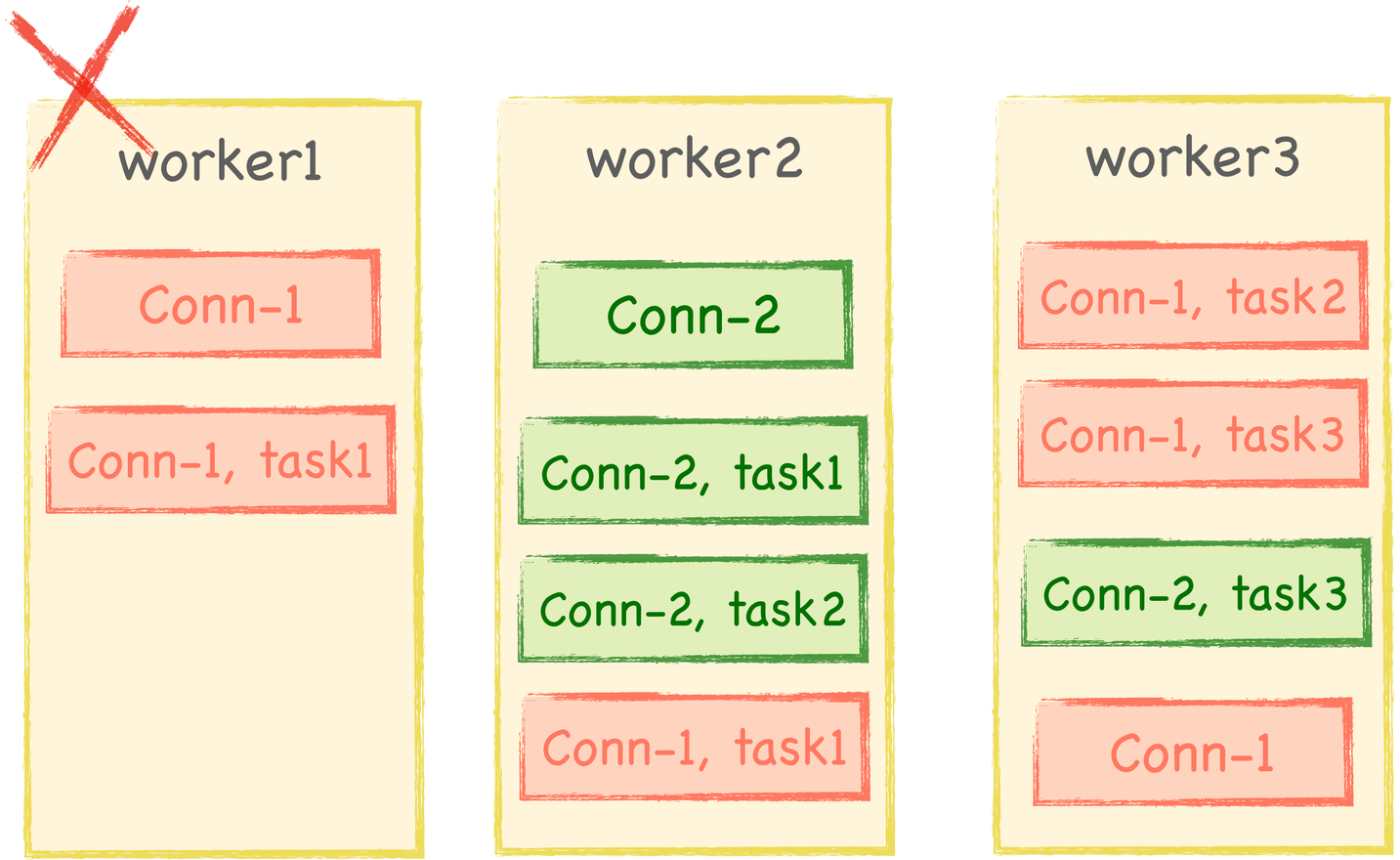
Conn-2
Conn-2, task1
Conn-2, task2

worker3

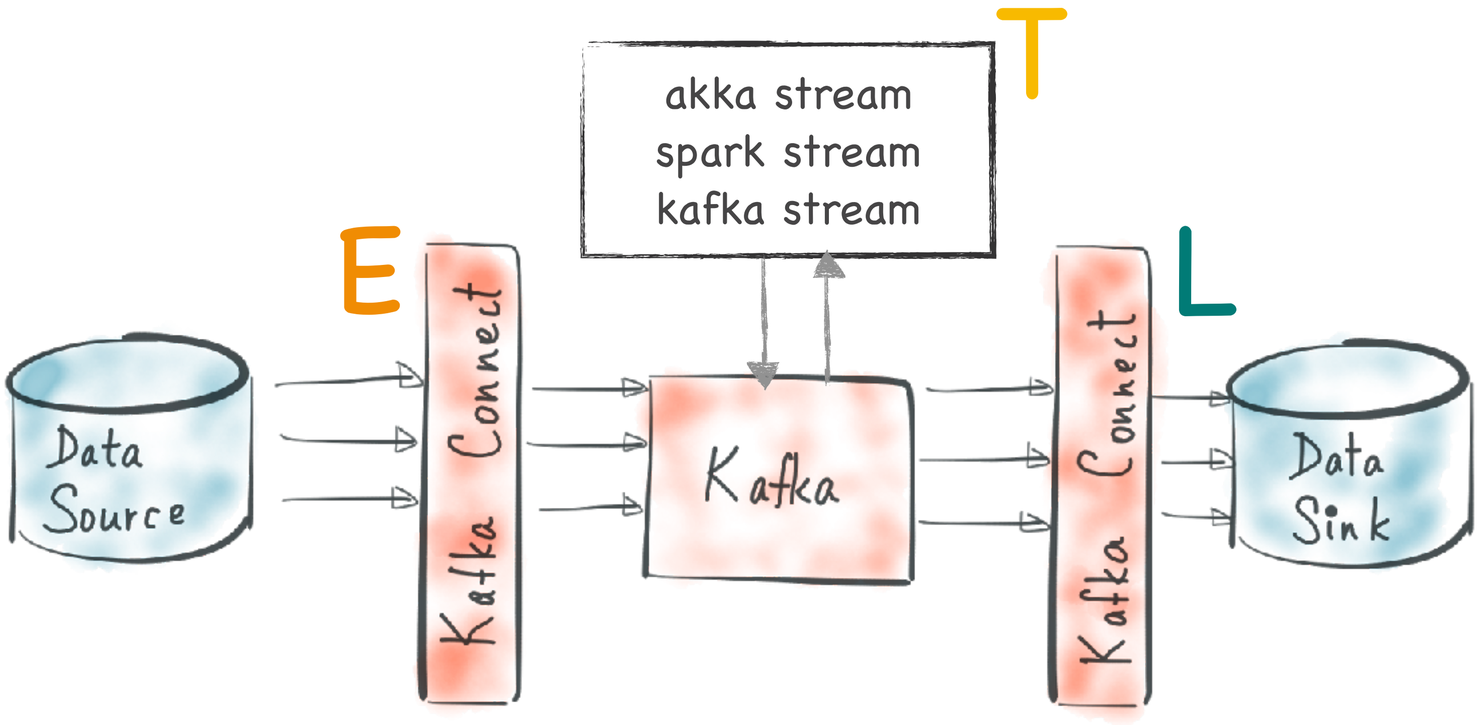
Conn-1, task2
Conn-1, task3
Conn-2, task3

standalone & Distributed





KAFKA CONNECT



Kafka Connect Quick Start - Confluent Platform

Confluent Platform Install

- Start all available services:

```
confluent start
```

Your output should resemble:

```
Starting zookeeper
zookeeper is [UP]
Starting kafka
kafka is [UP]
Starting schema-registry
schema-registry is [UP]
Starting kafka-rest
kafka-rest is [UP]
Starting connect
connect is [UP]
```

Confluent Connectors

confluent

Product Solutions Developers Blog Docs Download

Additional Connectors Available

Other notable Connectors that have been developed utilizing the Kafka Connect framework.

CONNECTOR	TAGS	DEVELOPER/SUPPORT	DOWNLOAD
Amazon Kinesis (Sink)	messaging	Community	Community
Apache Ignite (Source)	File System	Community	Community
Apache Ignite (Sink)	File System	Community	Community
ArangoDB (Sink)	NoSQL	Community	Community
AWS Lambda (Sink)	Cloud, AWS, Lambda	Community	Community
Azure DocumentDB (Sink)	DocumentDB, Azure, NoSQL	Community	Community
Blockchain (Source)	Bitcoin, Blockchain	Community	Community

Want to build a connector?

Interested Open Source Developers and Vendors can get started with the following resources:

> Kafka Connect Overview
> Kafka Connect Developers Guide
> Partner Development Guide for Kafka Connect

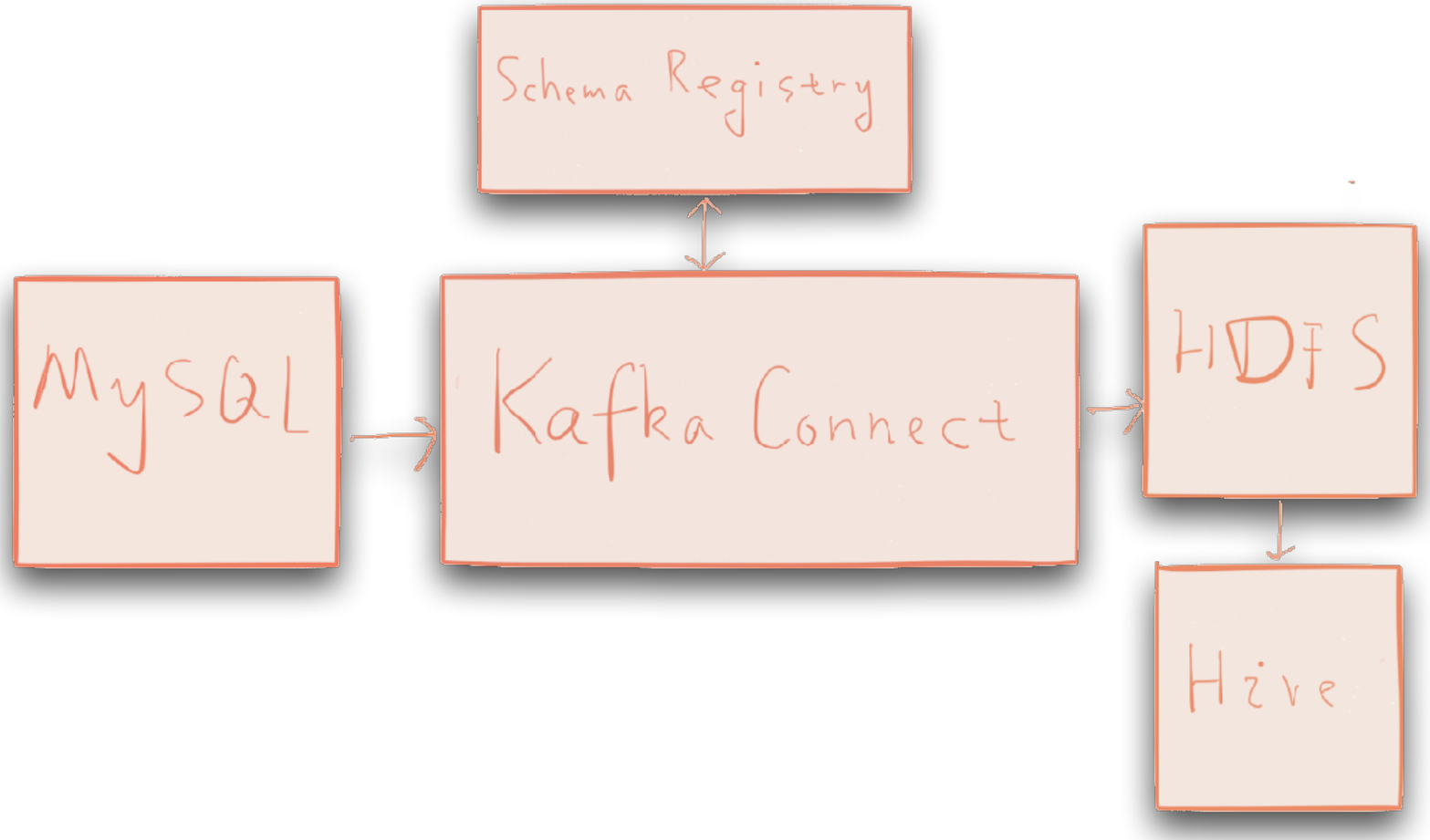
Already built a connector?

If you have a connector that you'd like to see added to our list, let us know at: confluent-platform@googlegroups.com

If your connector meets our criteria, we'll

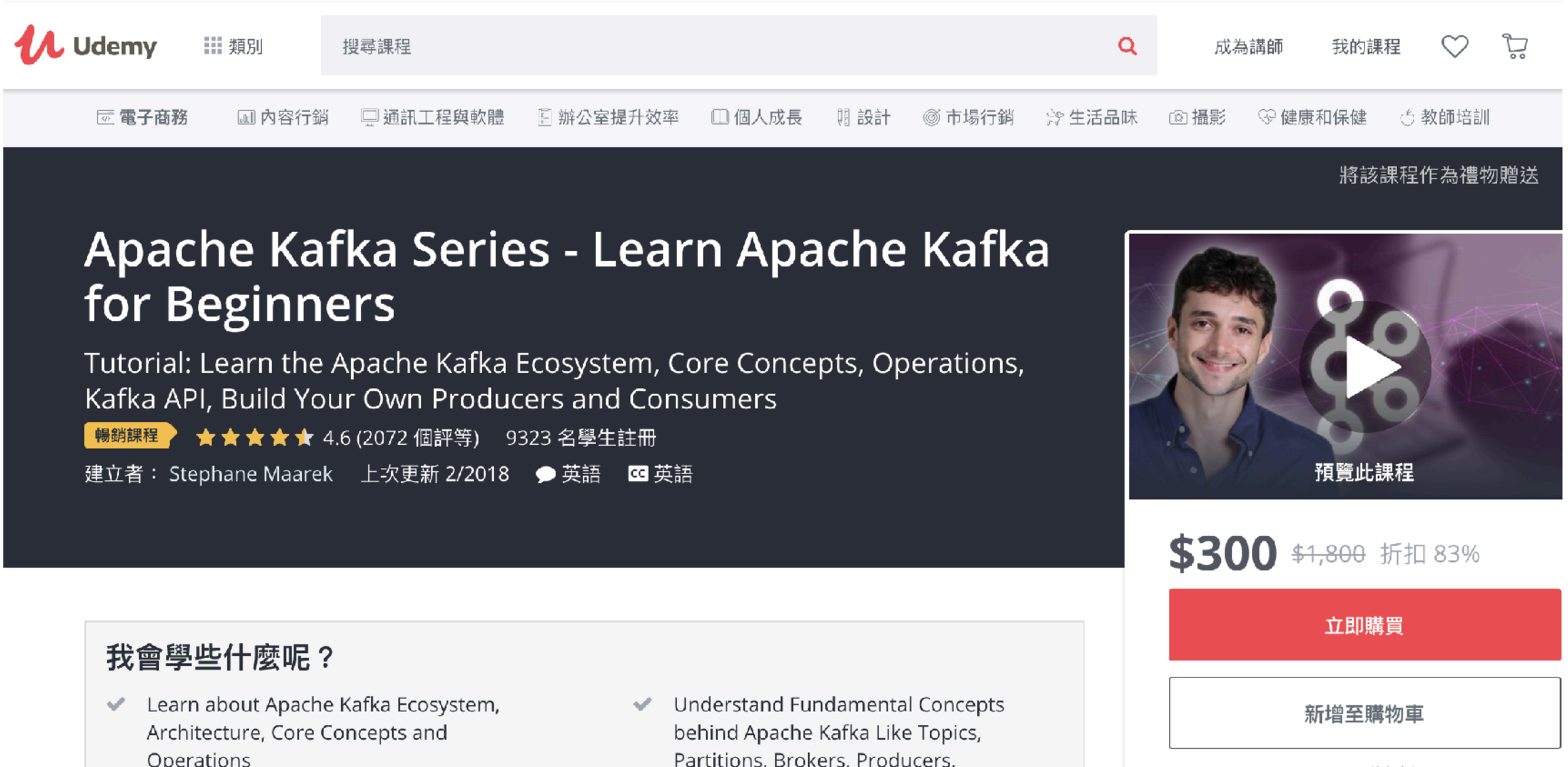
To demonstrate Kafka Connect,

A simple data pipeline: MySQL → Kafka → HDFS → Hive

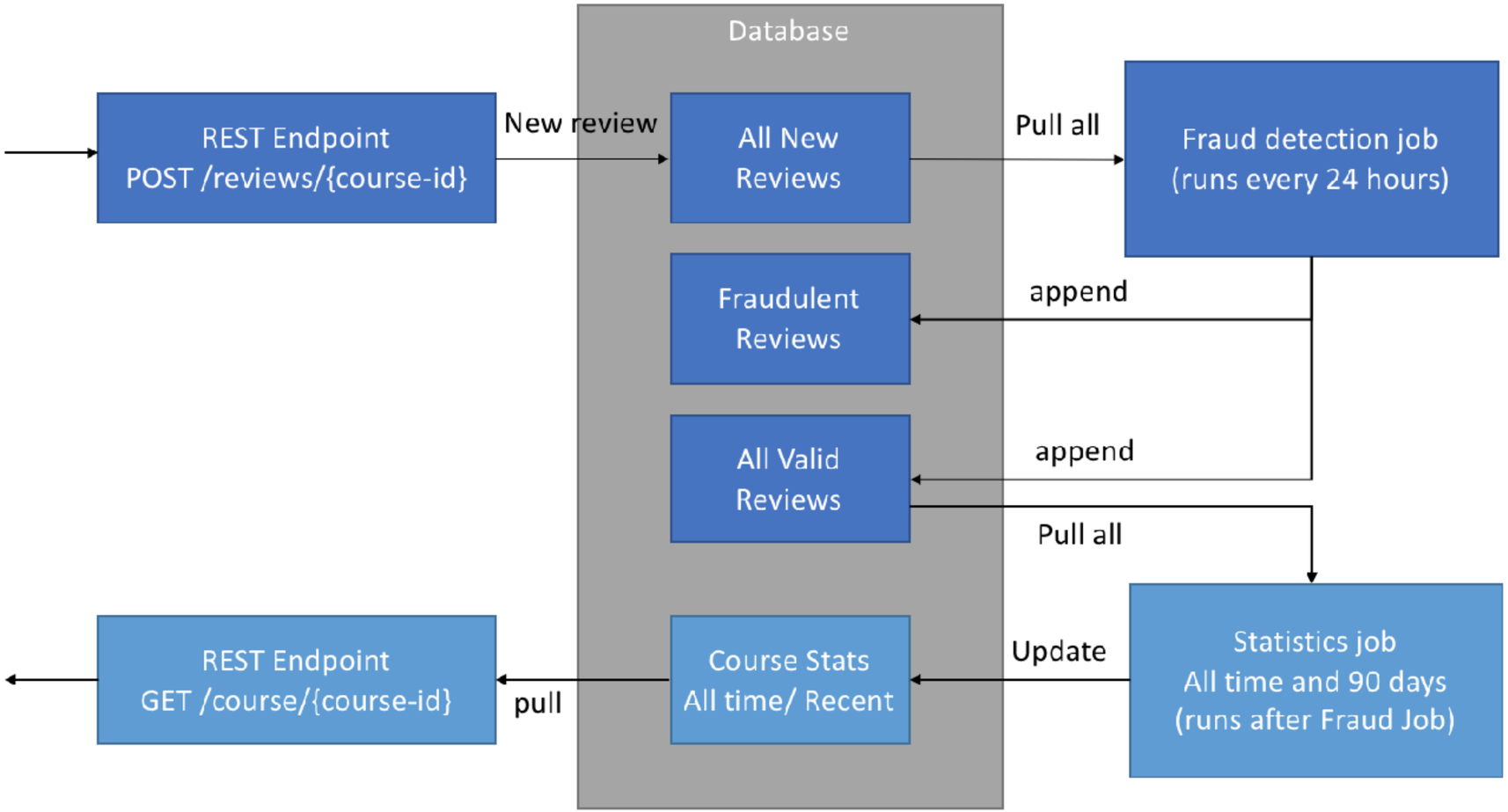


kafka connect tutorial

Simple Application:
transform a batch pipeline into a real-time one



Simple Application:
transform a batch pipeline into a real-time one



Simple Application:
transform a batch pipeline into a real-time one