COMMIT LOGS IN AN AGE OF MICROSERVICES

MARCH, 2019 / SEATTLE, WA

@GAMUSSA | #SEATTLEEVENTDRIVEN | @CONFLUENTINC

@GAMUSSA | #SEATTLEEVENTDRIVEN | @CONFLUENTINC confluent

HTTPS://GAMOV.DEV/SEATTLE-MICROSERVICES

RAFFLE, YEAH 🚀

- Follow @gamussa @confluentinc
- 📸 🖼 🏋
- Tag @gamussa
- With #seattleeventdriven

NORMAL APPLICATIONS

(I.E., MONOLITHS)

MONOLITHS ARE HARD TO THINK ABOUT

MONOLITHS ARE HARD TO CHANGE

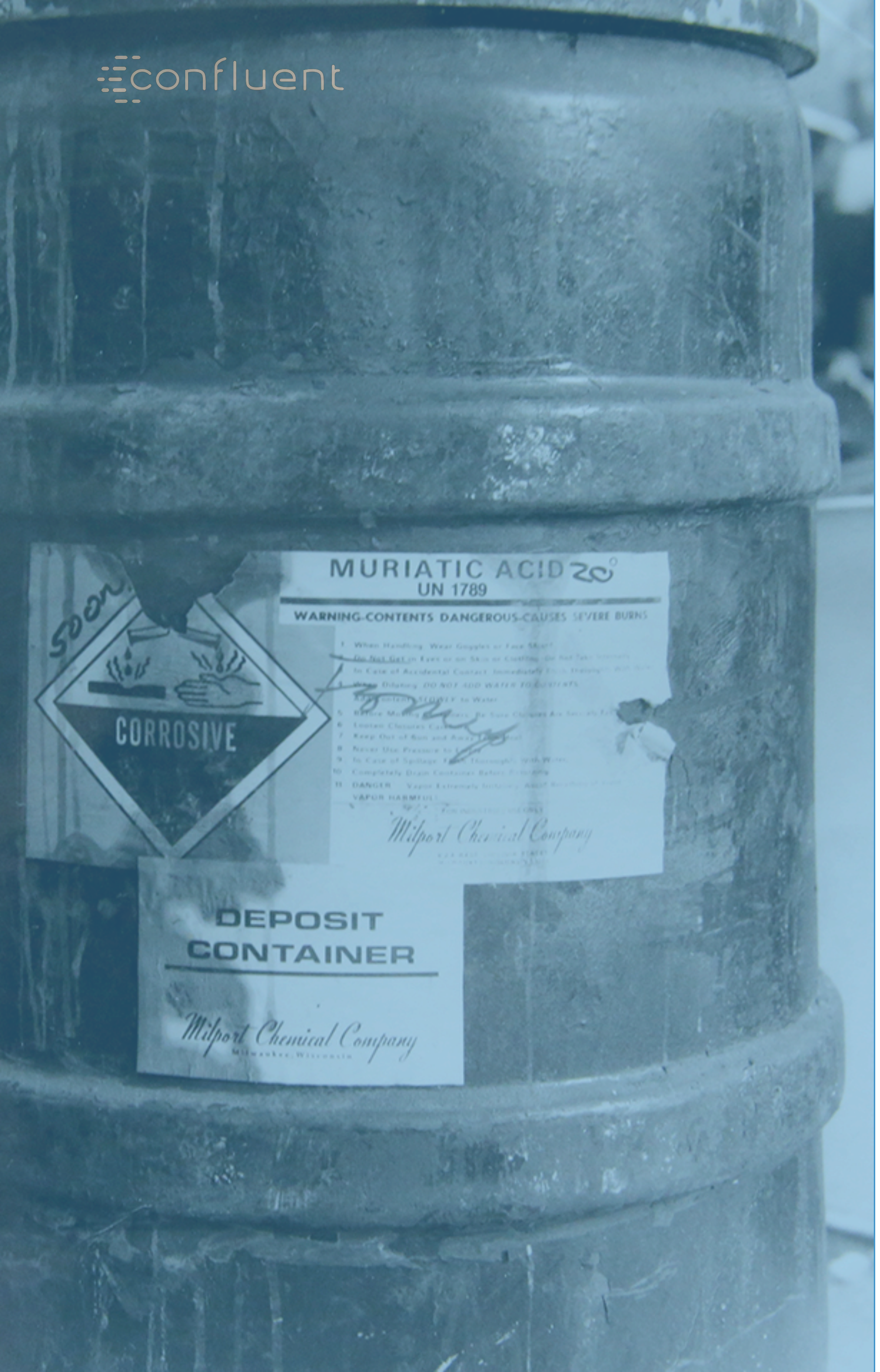
RE-INTEGRATION

THERE ARE NO GOOD WAYS TO INTEGRATE MICROSERVICES

THERE ARE NO-GOOD WAYS TO INTEGRATE MICROSERVICES

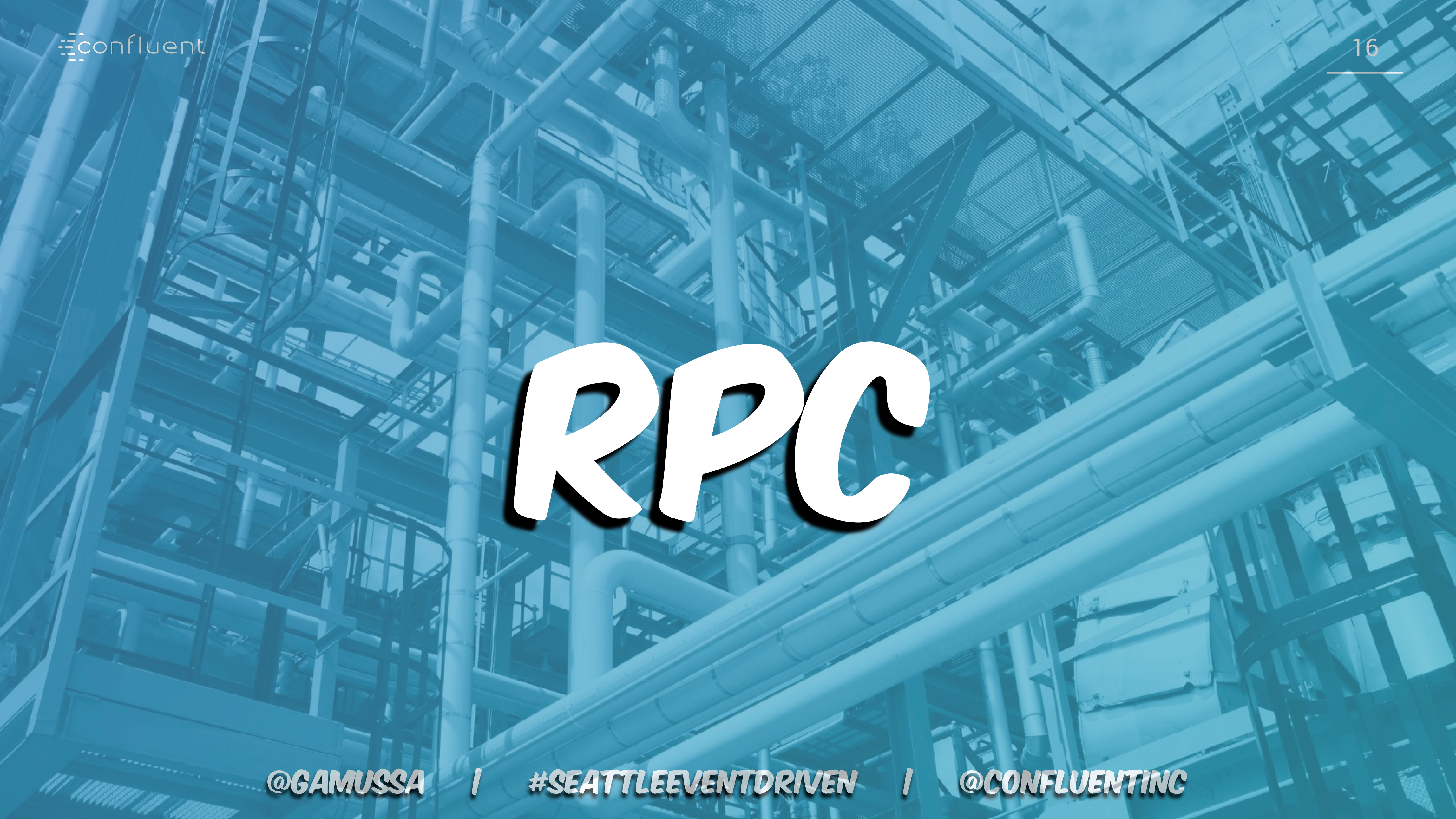
FILESYSTEM

DATABASE



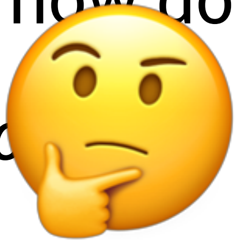
INTEGRATING MICROSERVICES THROUGH THE DATABASE

- “I have a database and I know how to use it.”
- Eventually causes services to co-mingle.
- Violation the «bounded context»
- Great to use **inside** a service boundary!
- Terrible for **sharing data** or **negotiating change**.



RPC

INTEGRATING MICROSERVICES VIA RPC

- Avoids problems of database integration
- Feels natural
- Aligns with the request/response paradigm
- Problem: cascading failures
- Question: how do you **debug** this system?
- Answer: *yo*  *a log*.

EVENTS?

WHAT'S AN EVENT?

A SHARED NARRATIVE DESCRIBING THE EVOLUTION OF THE BUSINESS OVER TIME

A COMBINATION OF:

- NOTIFICATION
- STATE TRANSFER

ALSO,

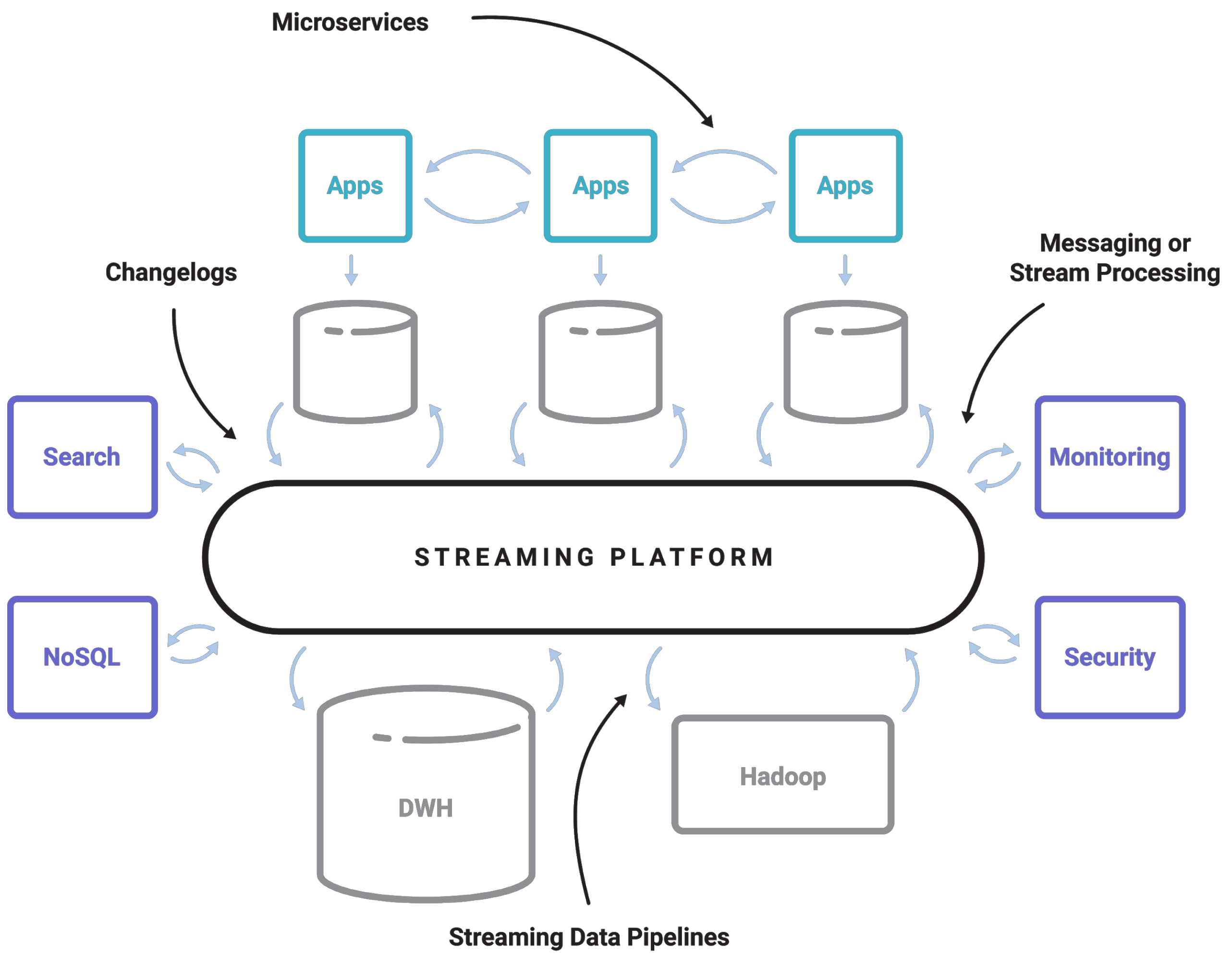
EVENTS

ARE IMMUTABLE.

KAFKA BASICS

- STORAGE
- PUB / SUB
- PROCESSING

STREAMING PLATFORM



CORE

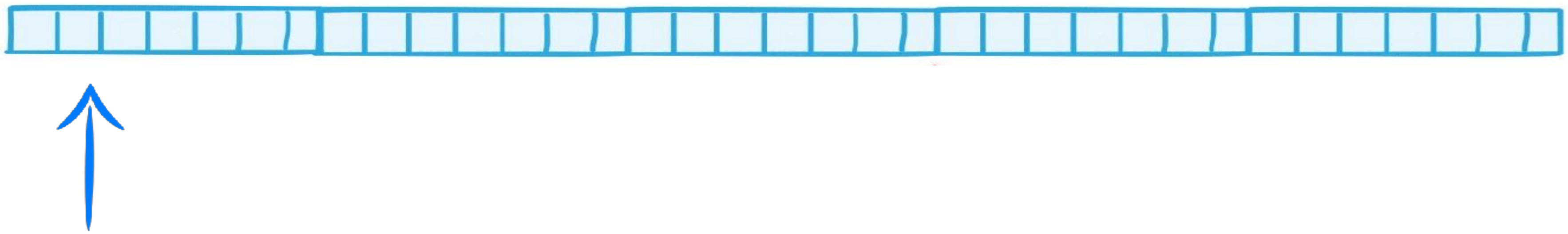
ABSTRACTION

Core abstraction

- DB - table
- Hadoop - file
- Kafka - ?

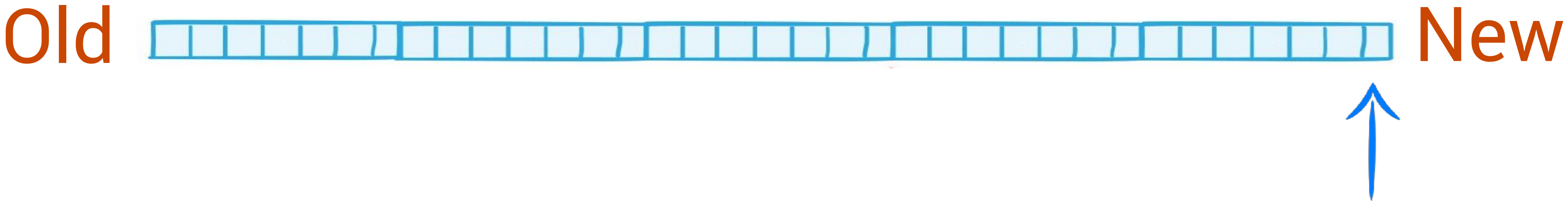
LOG

THE LOG IS A SIMPLE IDEA

Old  New

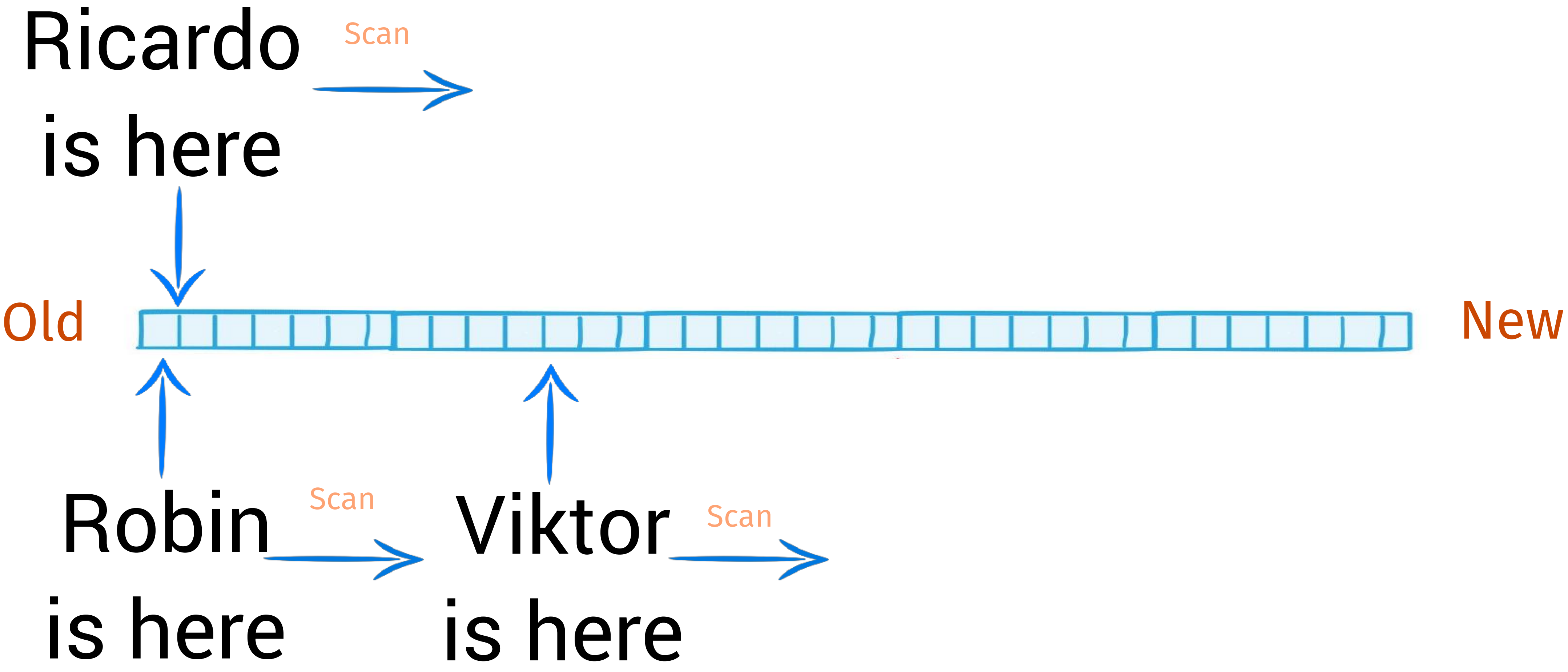
Messages are added at the end of the log

THE LOG IS A SIMPLE IDEA

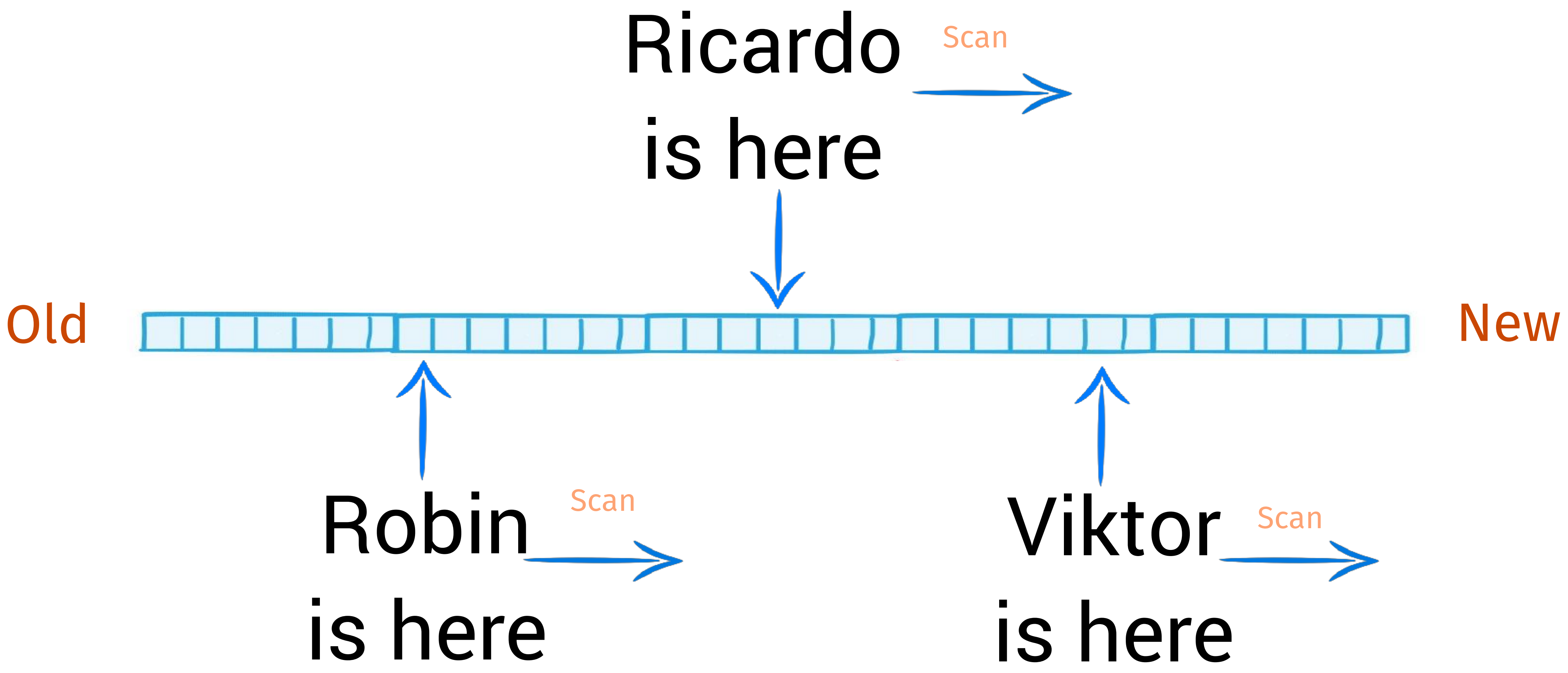


Messages are added at the end of the log

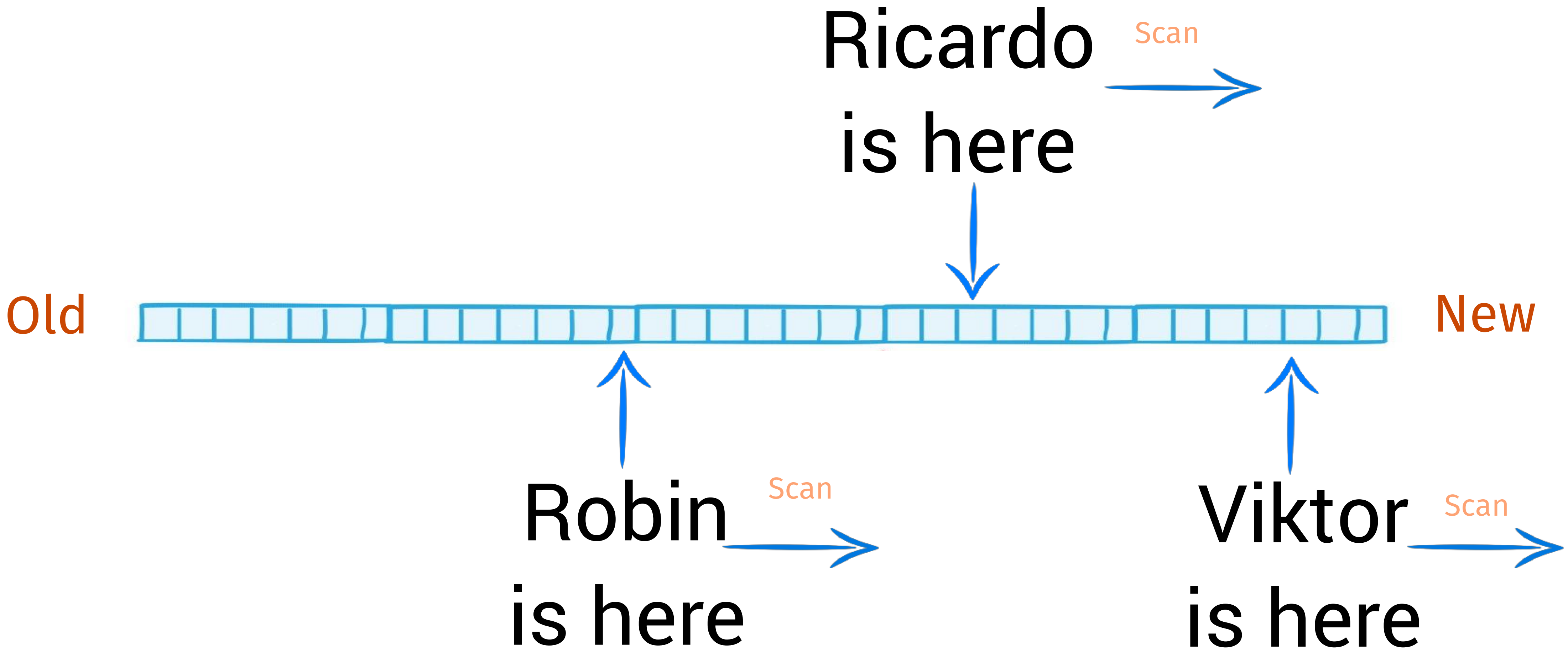
CONSUMERS HAVE A POSITION ALL OF THEIR OWN



CONSUMERS HAVE A POSITION ALL OF THEIR OWN



CONSUMERS HAVE A POSITION ALL OF THEIR OWN



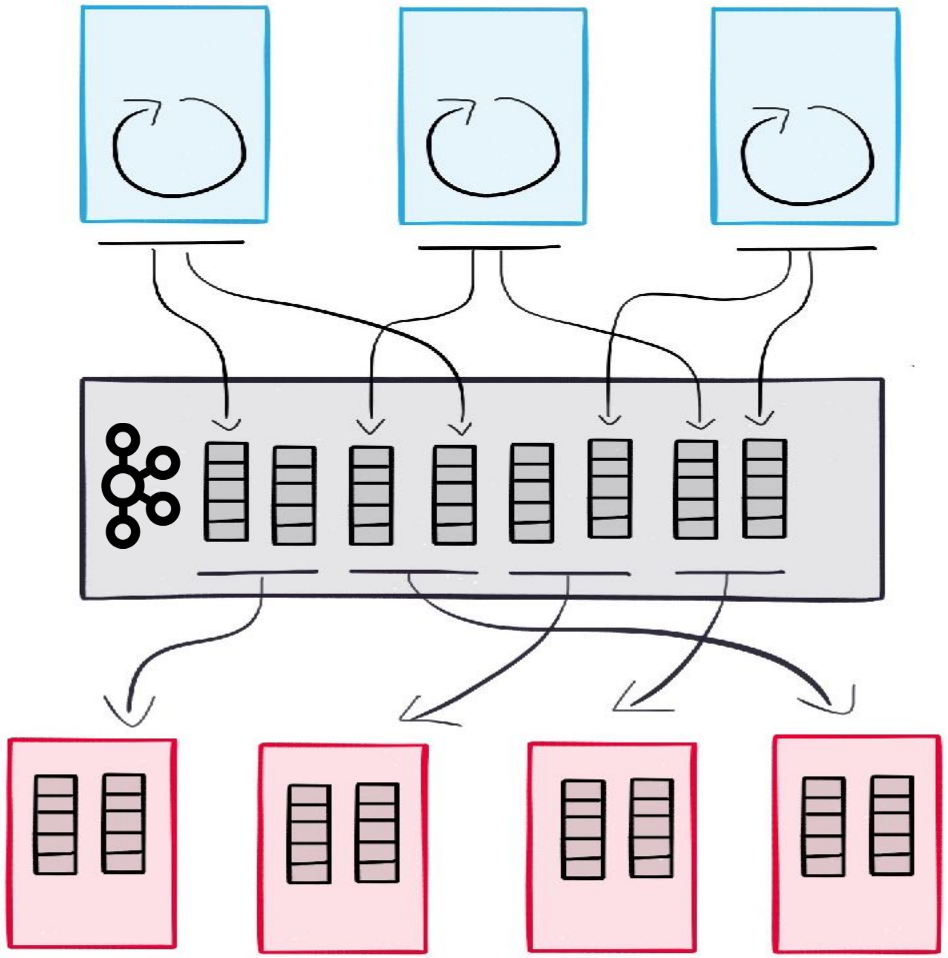
ONLY SEQUENTIAL ACCESS

Read to offset & scan

Old  New

LINEARLY SCALABLE ARCHITECTURE

Producers



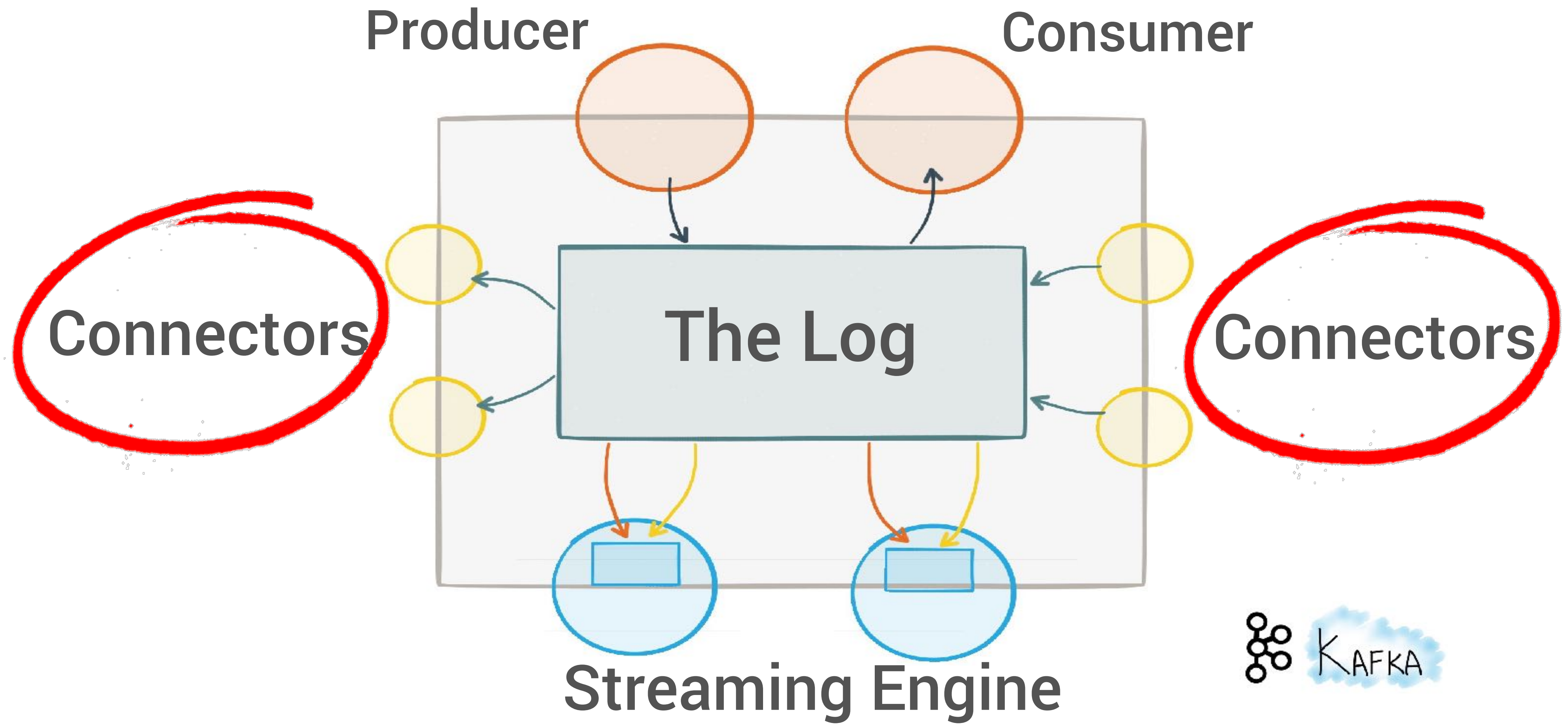
Consumers

Single topic:

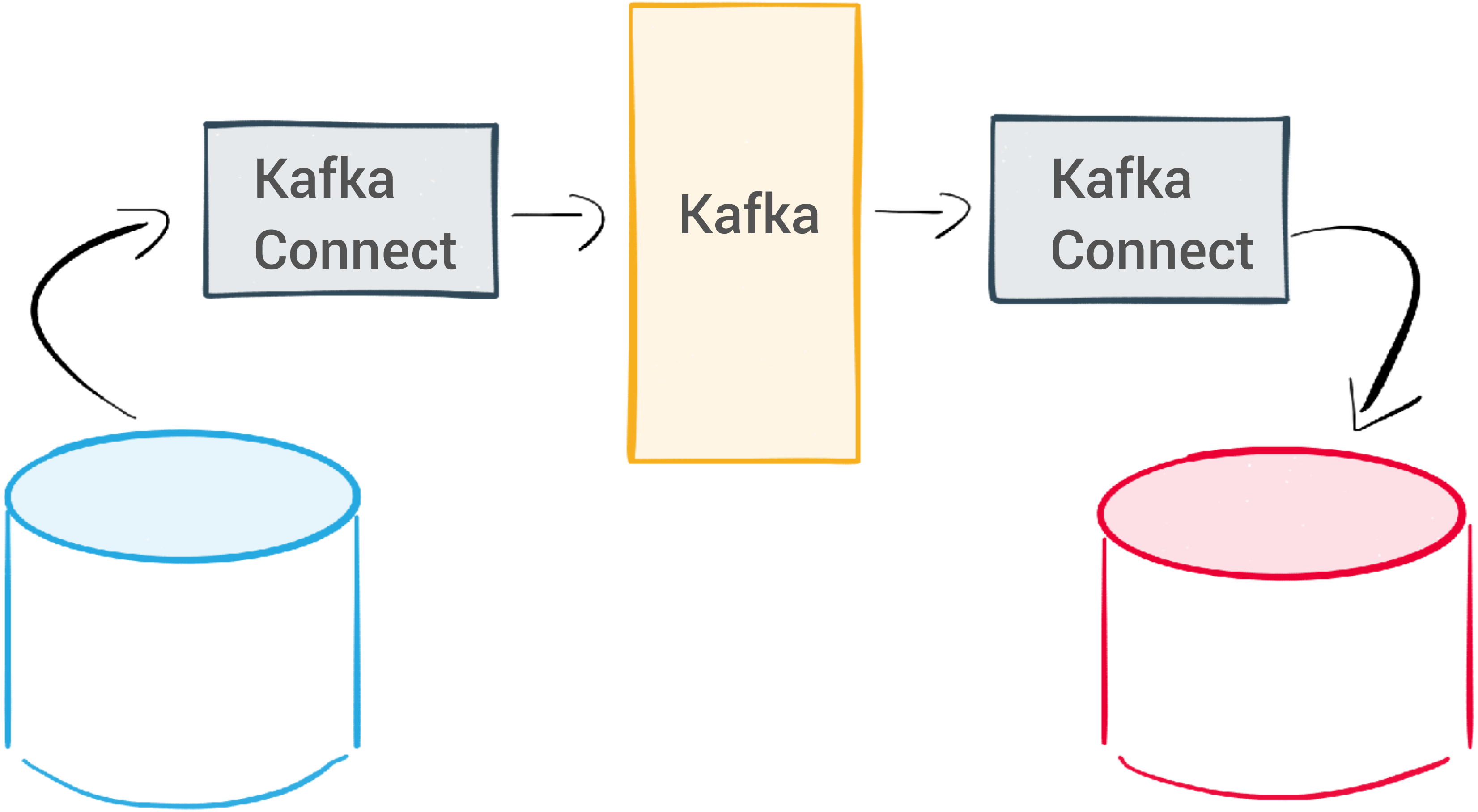
- Many producers machines
- Many consumer machines
- Many Broker machines

No Bottleneck!!

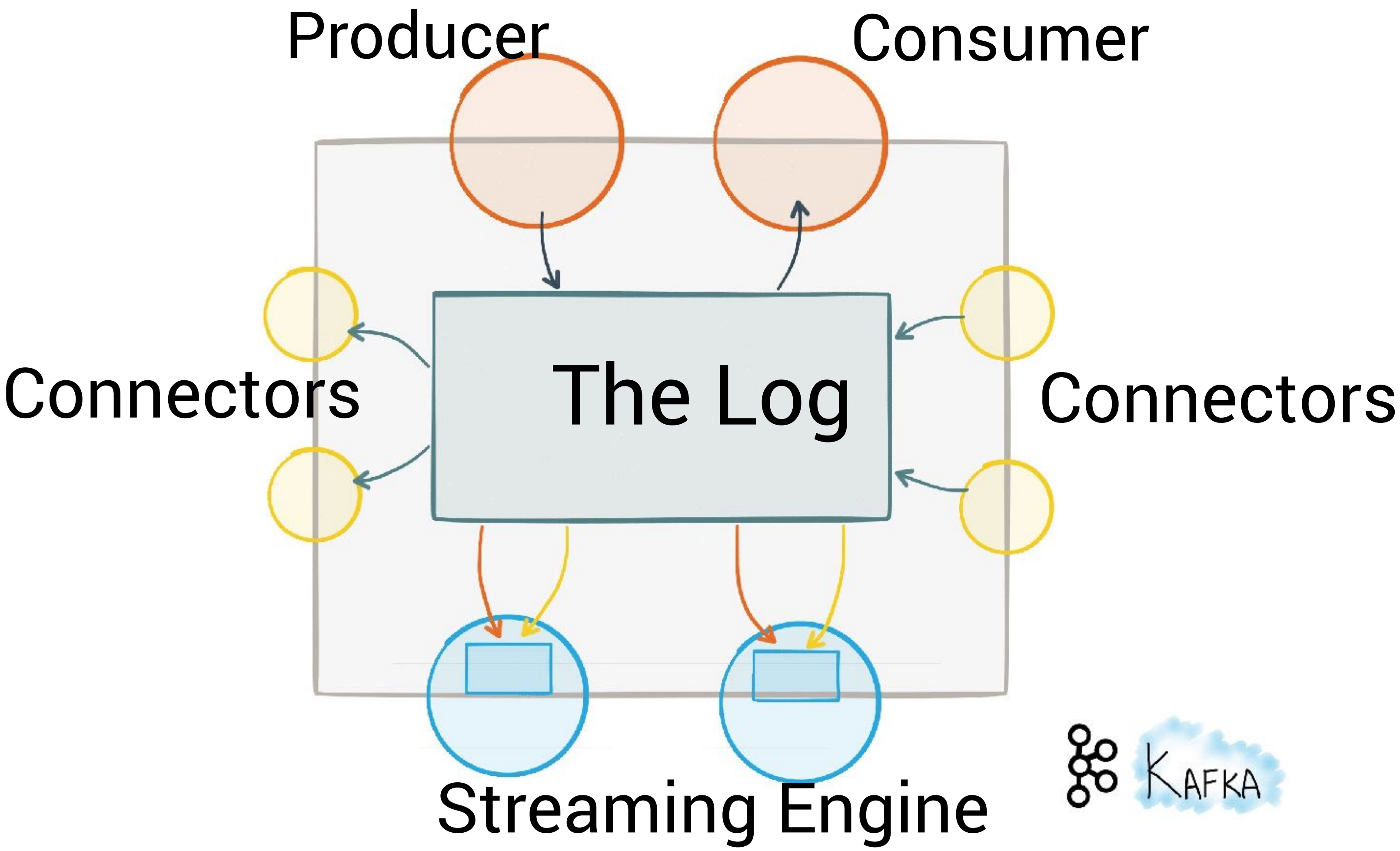
THE CONNECT API



INGEST / OUTPUT TO ANY DATA SOURCE



STREAM PROCESSING IN KAFKA



KSQL: AN ENGINE FOR CONTINUOUS COMPUTATION

```
SELECT card_number,
count(*)
FROM authorizations
WINDOW (SIZE 5 MINUTE)
GROUP BY card_number
HAVING count(*) > 3;
```

KAFKA STREAMS: A JAVA API FOR THE SAME

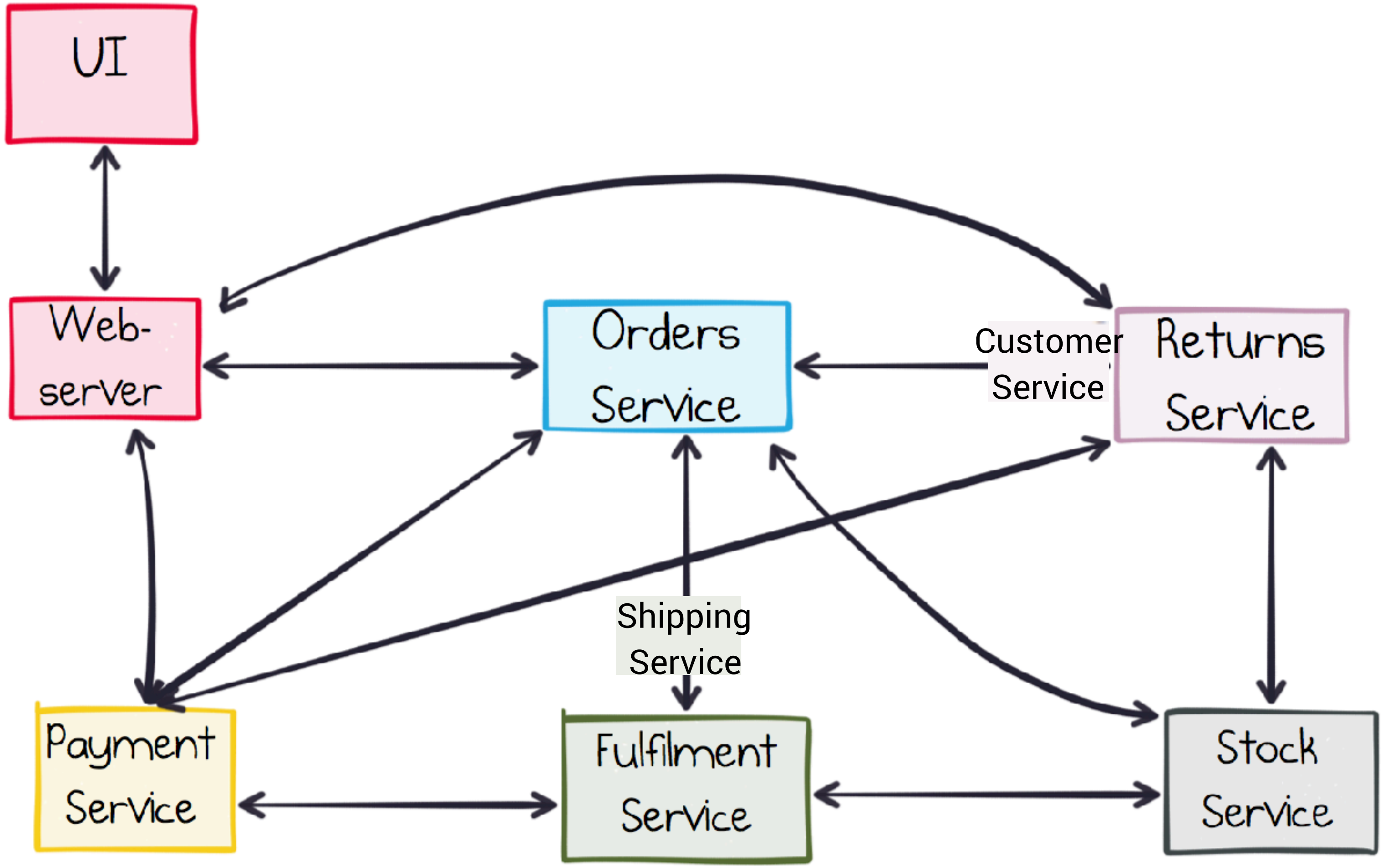
```java
StreamsBuilder builder = new StreamsBuilder();

builder
    .stream("caterpillars")
    .map(Metamorphosis::coolTransformation)
    .to("butterflies");

final Topology topology = builder.build();
new KafkaStreams(topology, config()).start();
```

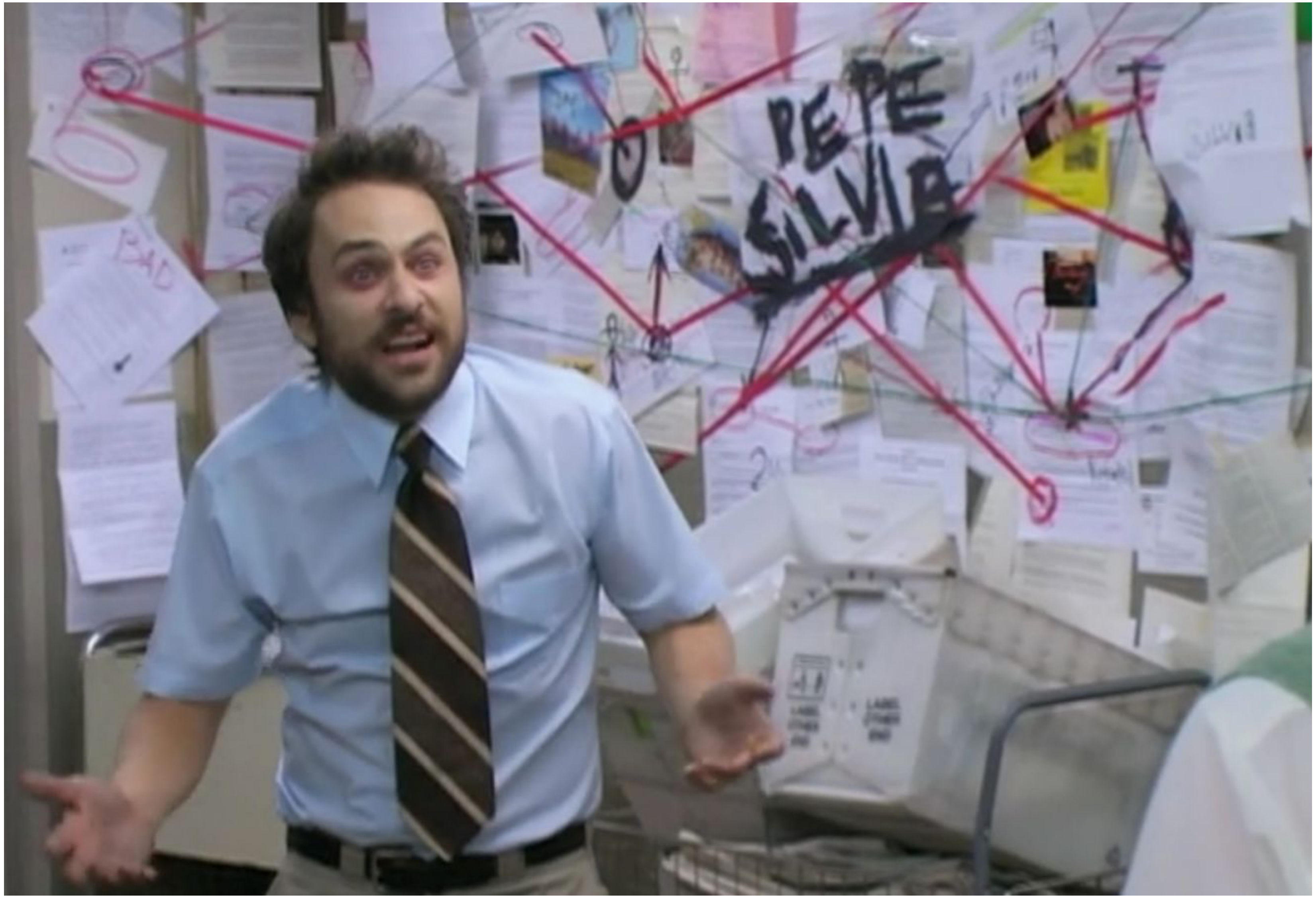
Kafka Streams: a Java API for the same

```
StreamsBuilder builder = new StreamsBuilder();

builder
      .stream("caterpillars")
      .map(Metamorphosis::coolTransformation)
      .to("butterflies");

final Topology topology = builder.build();
new KafkaStreams(topology, config()).start();
```

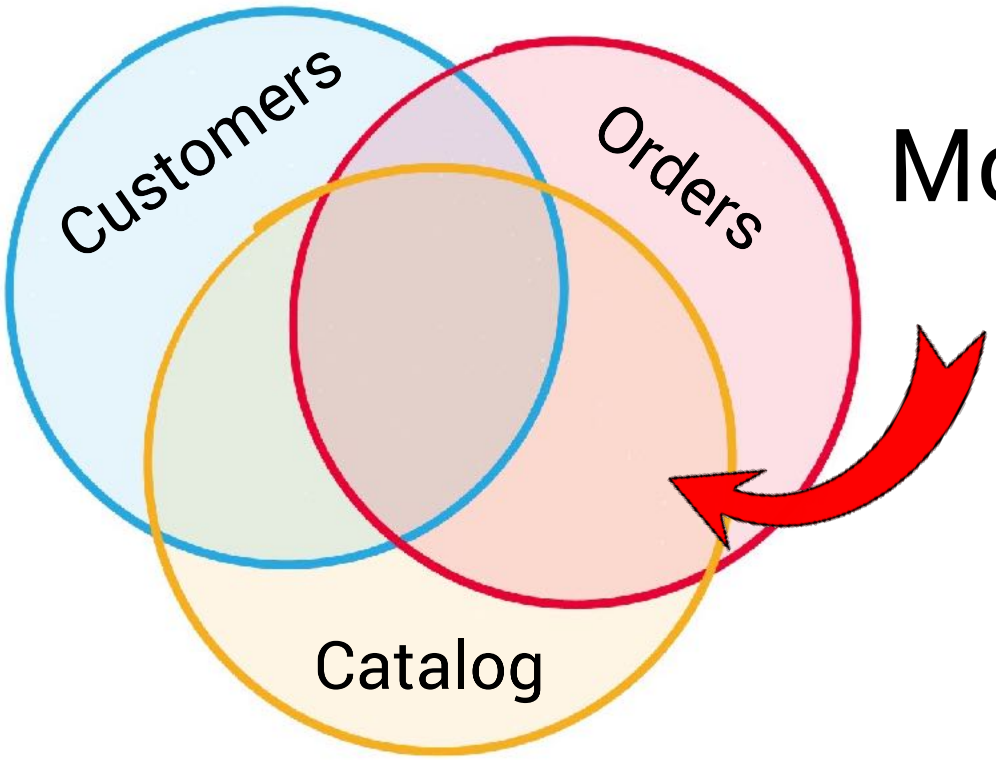
MICROSERVICES

SUPPOSE WE HAVE THESE SERVICES

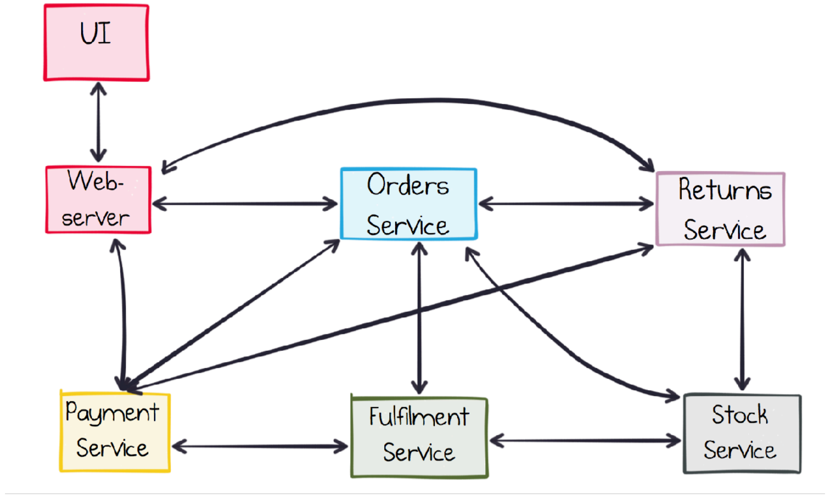



PEPE
SILVIA

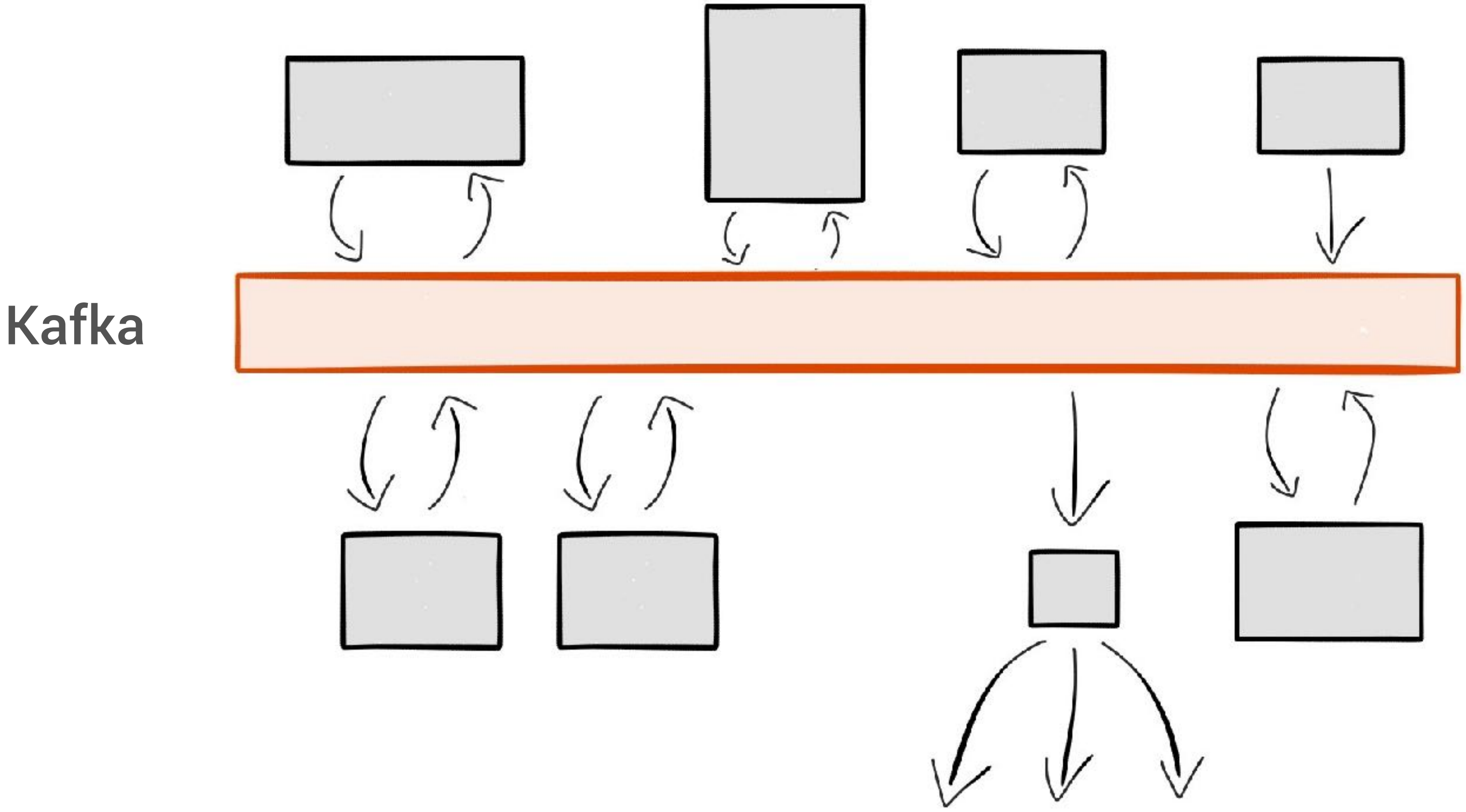
SERVICES SHARE THE SAME CORE FACTS

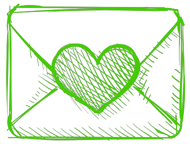


Most services live in here

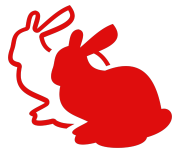


KAFKA IS A BACKBONE FOR EVENTS EXCHANGE



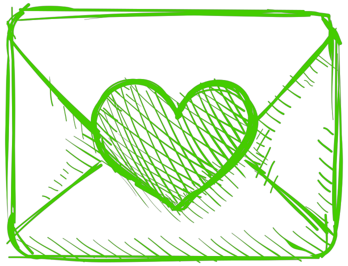


Notification

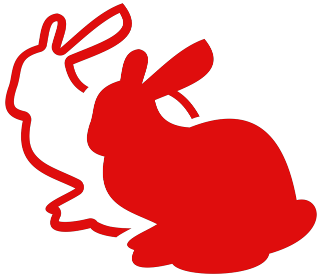


Data is replicated

TWO TYPES OF EVENTS

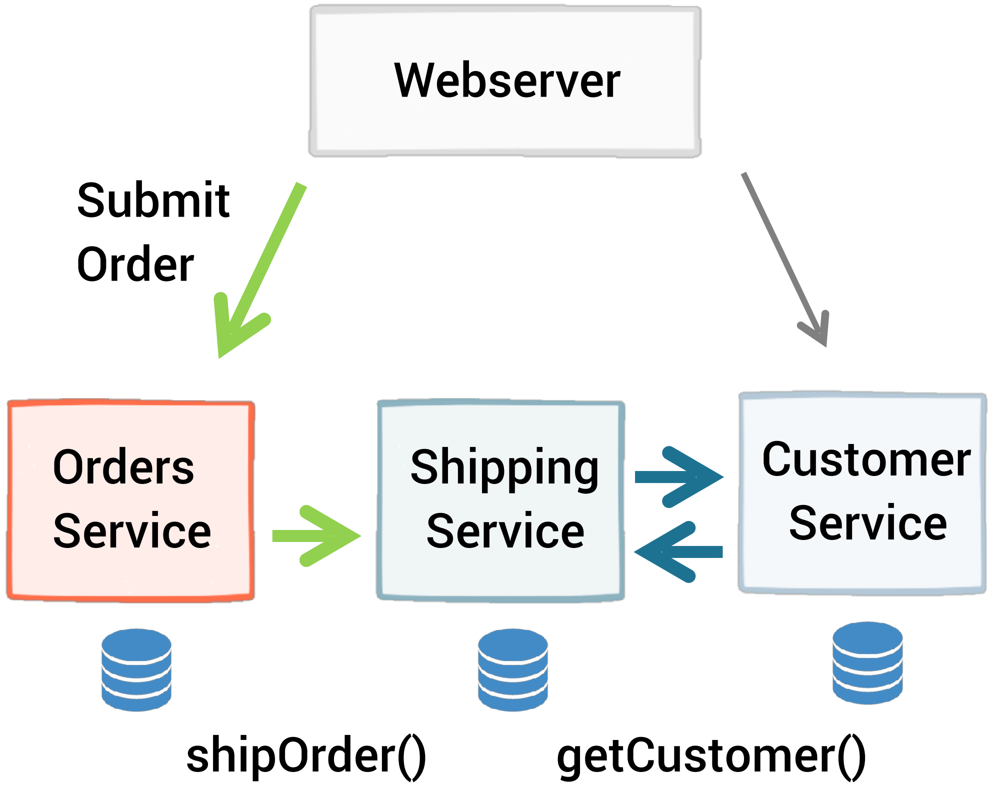


Notification



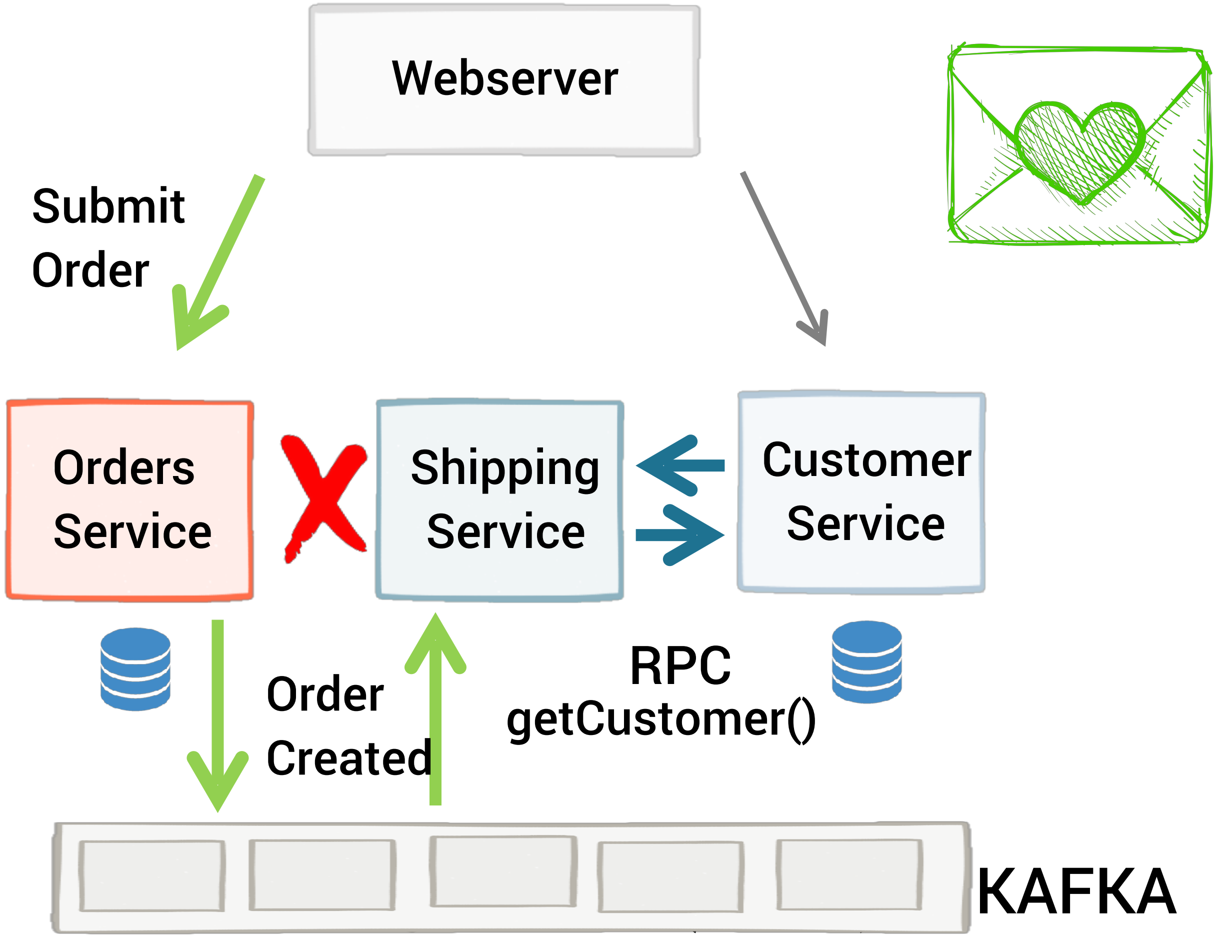
Data replication

ECOMMERCE MICROSERVICES (WITH RPC)



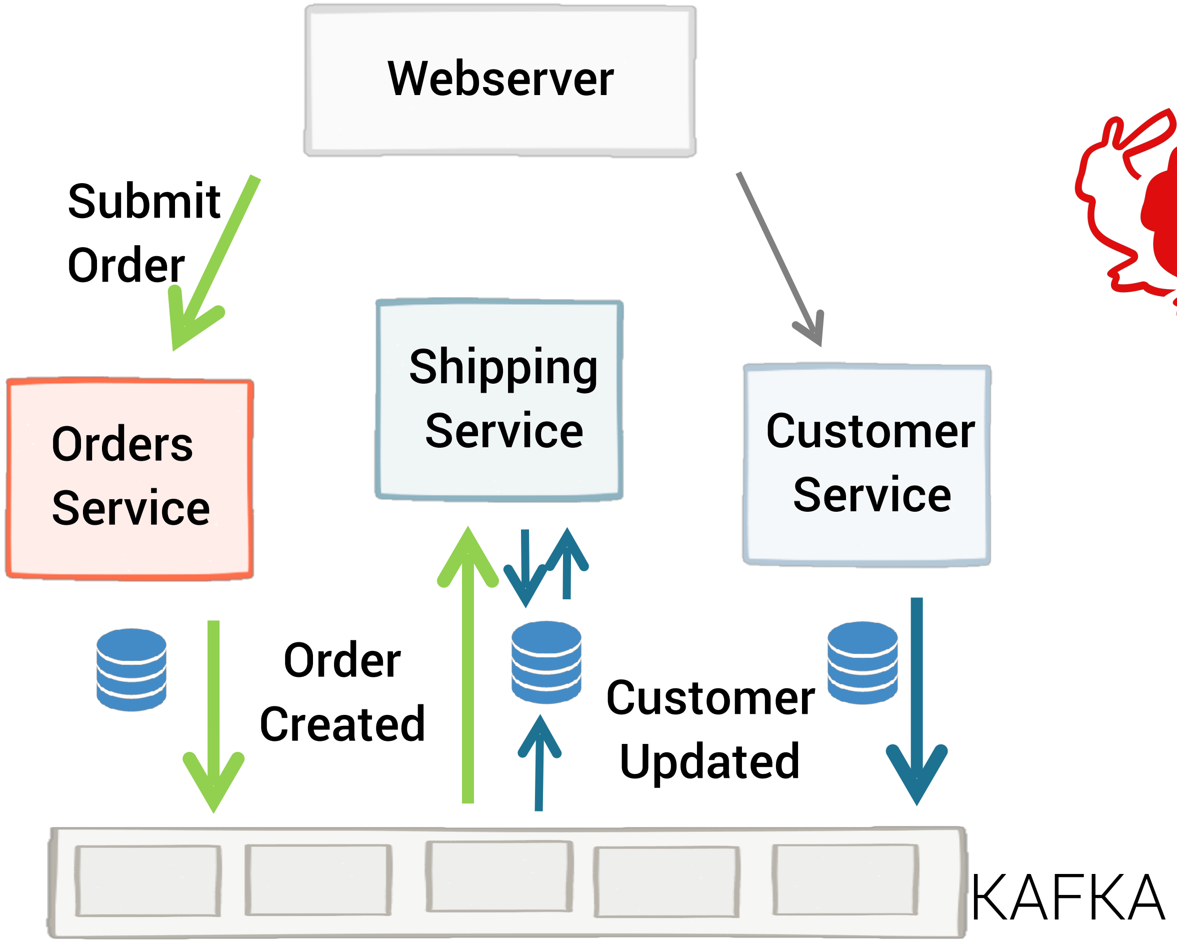
- **Orders Service** calls **Shipping Service** to tell it to ship item.
- **Shipping service** looks up address to ship to (from **Customer Service**)
- No Kafka 😢

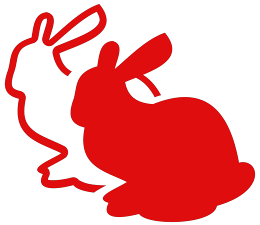
REFACTORING ORDERS AND SHIPPING



- **Orders Service** no longer knows about the **Shipping service** (or any other service).
- Events are fire and forget.

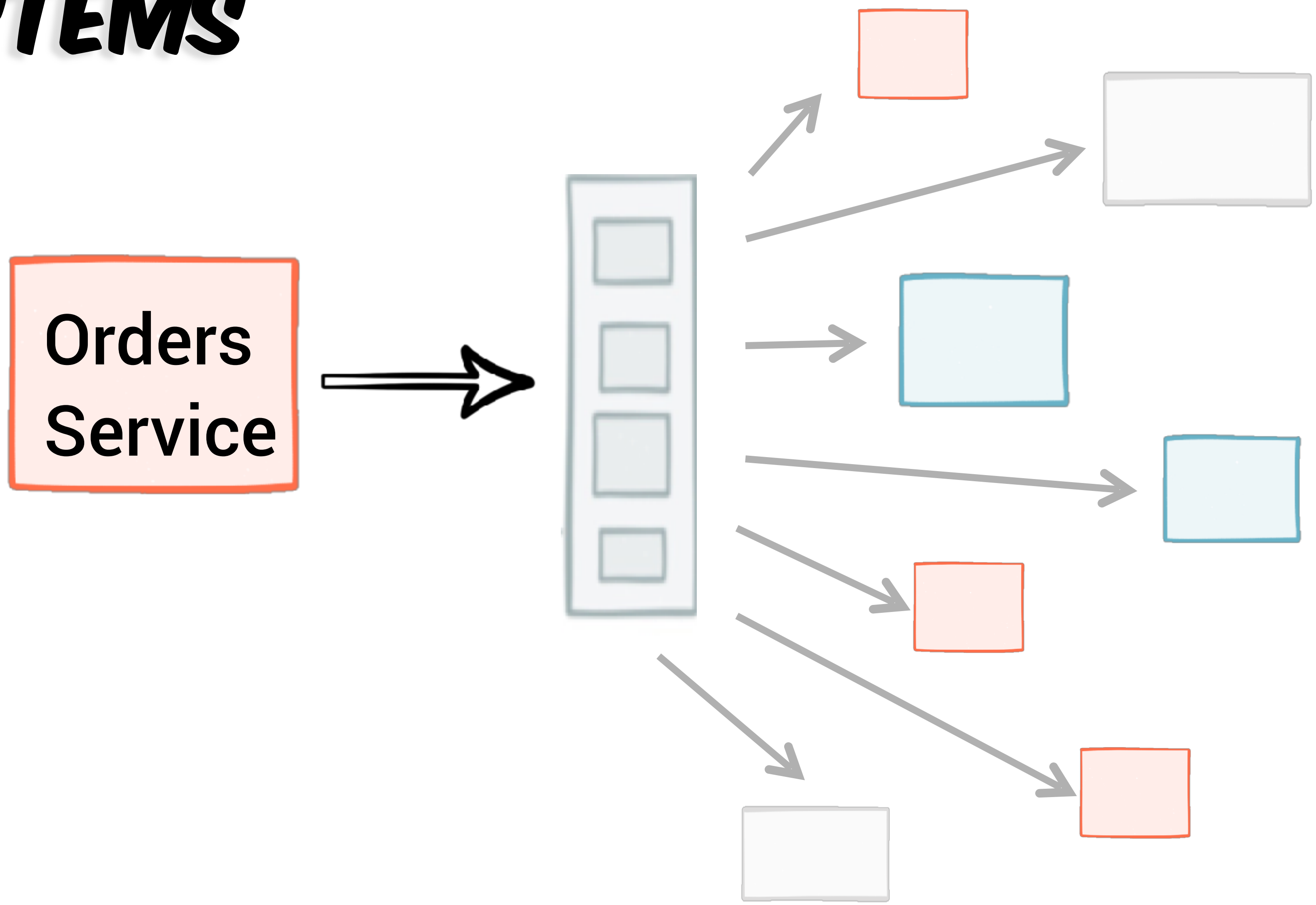
REFACTORING CUSTOMERS





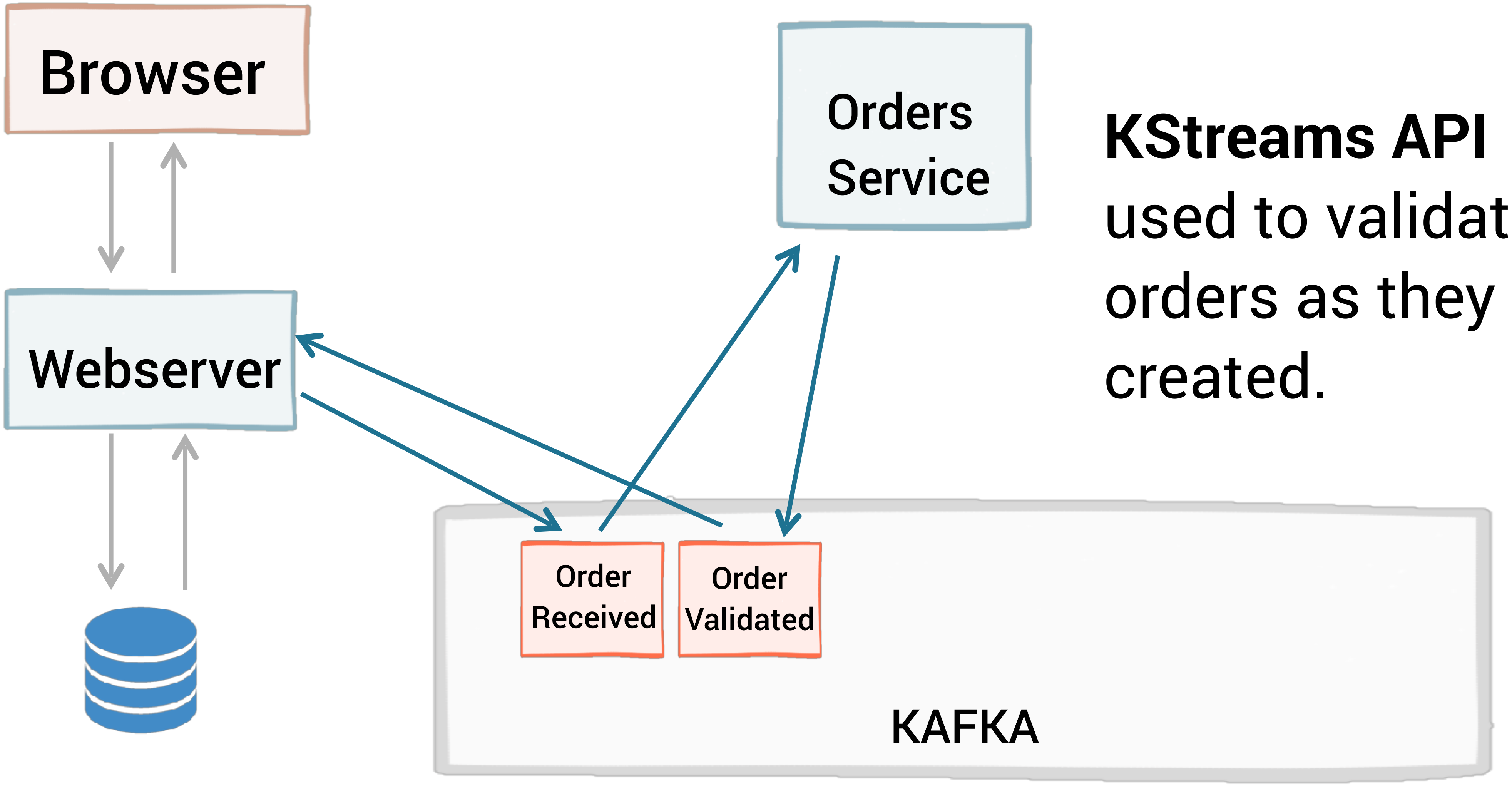
- Call to **Customer service** is gone.
- Instead data in replicated, as events, into the shipping service, where it is queried locally.

EVENTS ARE THE KEY TO SCALABLE SERVICE ECOSYSTEMS



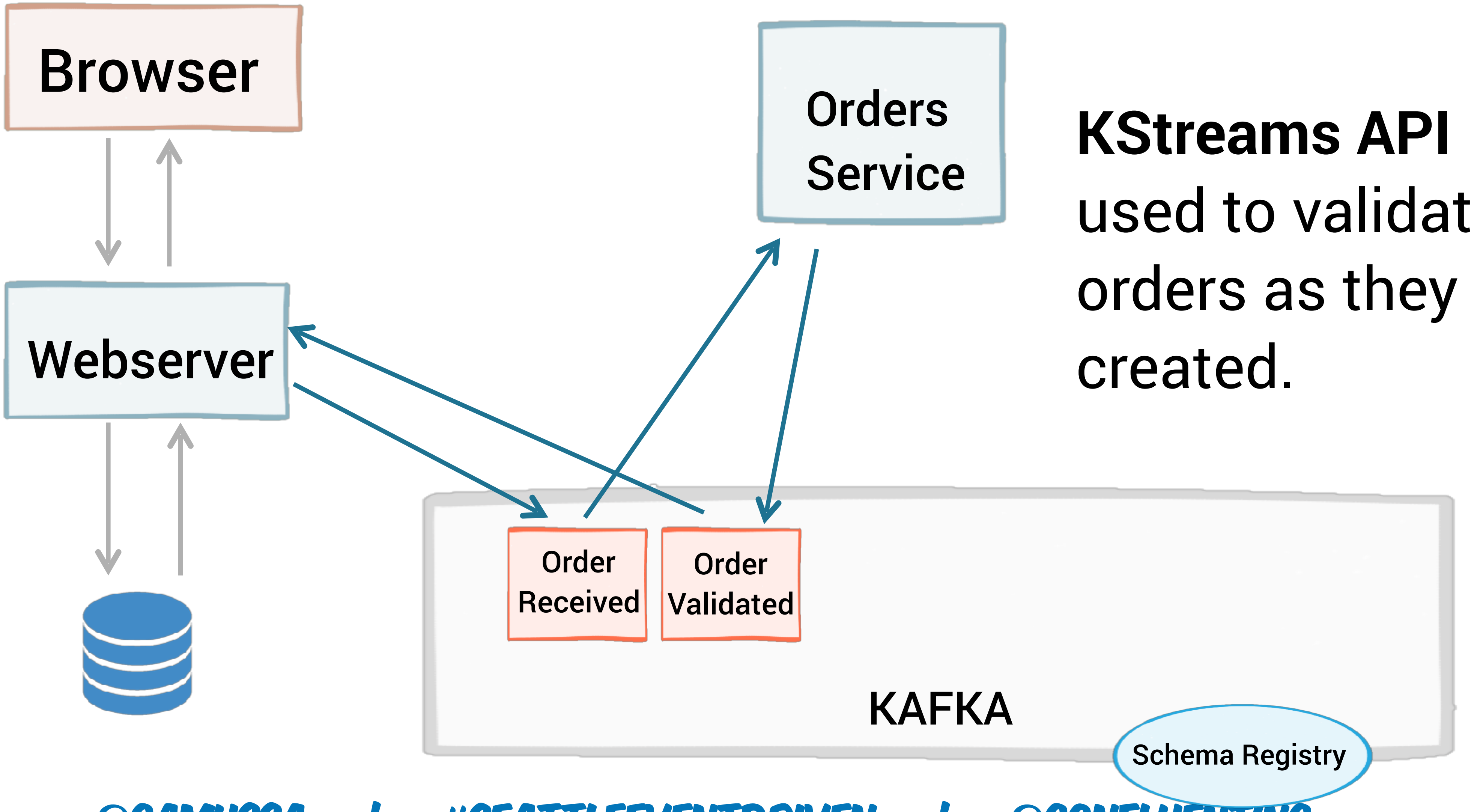
A RICHER MICROSERVICES APPLICATION

ORDERS SERVICE OPERATING ON THE EVENT STREAM



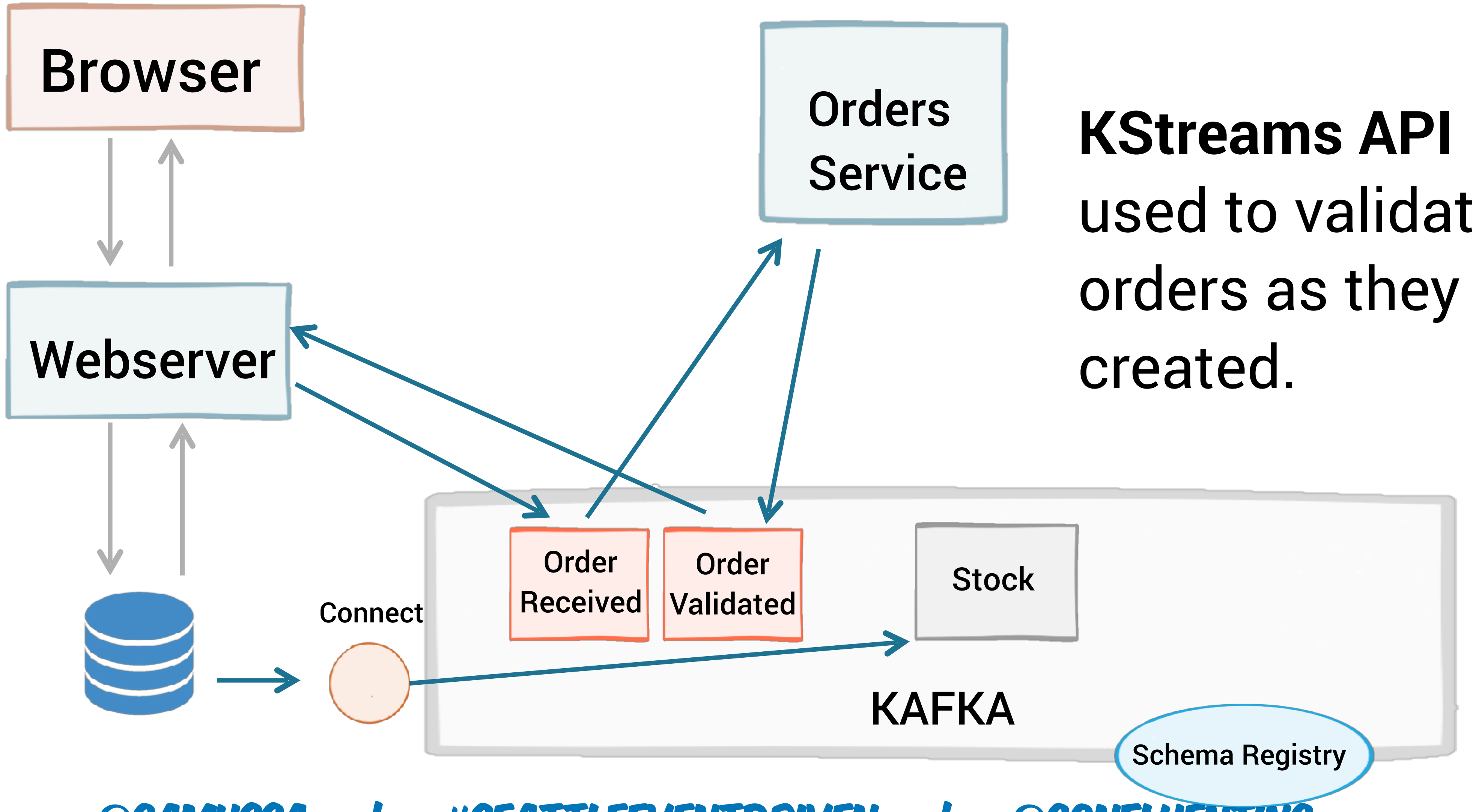
KStreams API used to validate orders as they are created.

SCHEMA MANAGEMENT



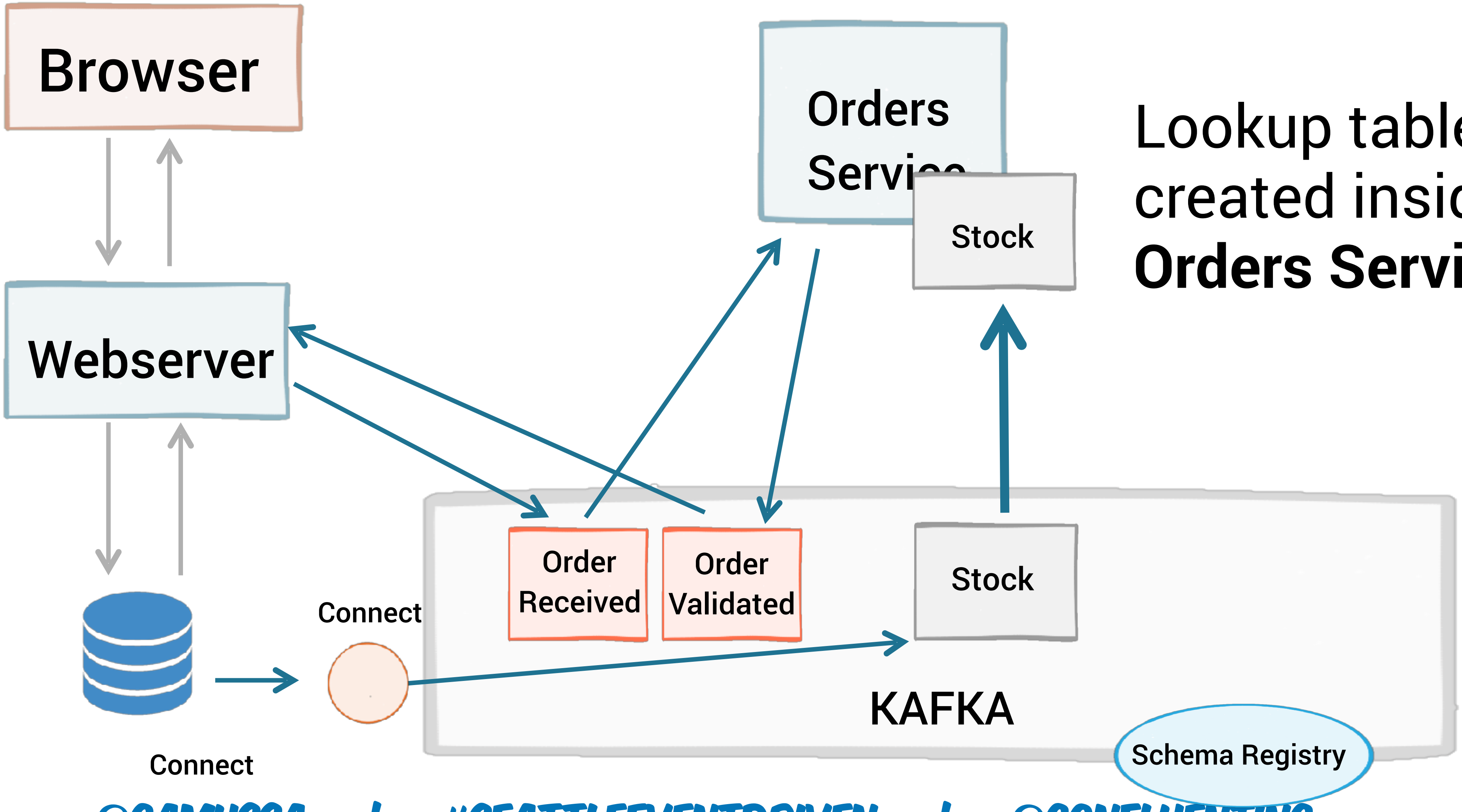
KStreams API used to validate orders as they are created.

ACCESS LEGACY RELATIONAL DATA



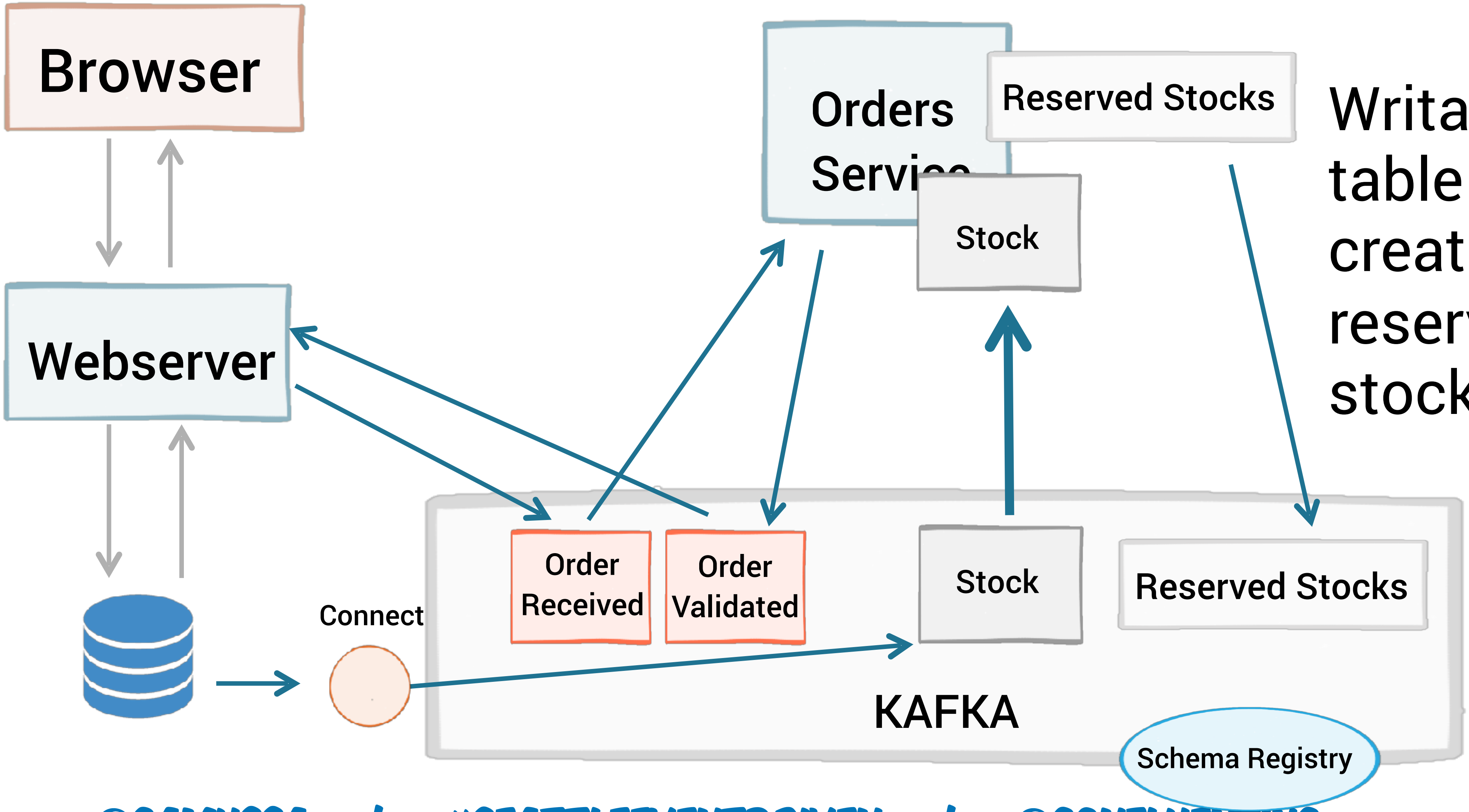
KStreams API used to validate orders as they are created.

MATERIALIZE TABLES INSIDE YOUR APP



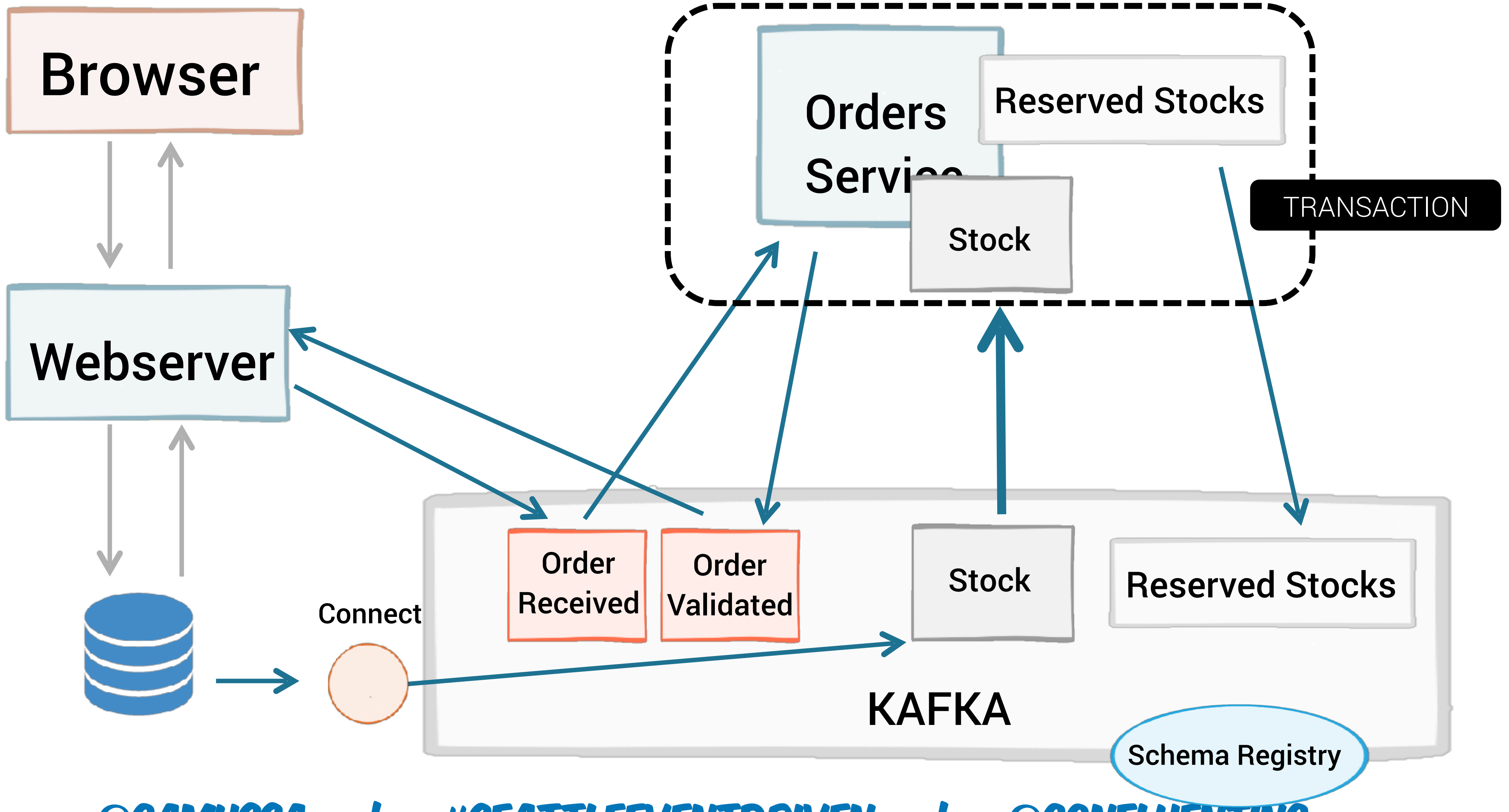
Lookup table created inside the **Orders Service**

CREATE A NEW TABLE, PERSIST IT TO THE LOG

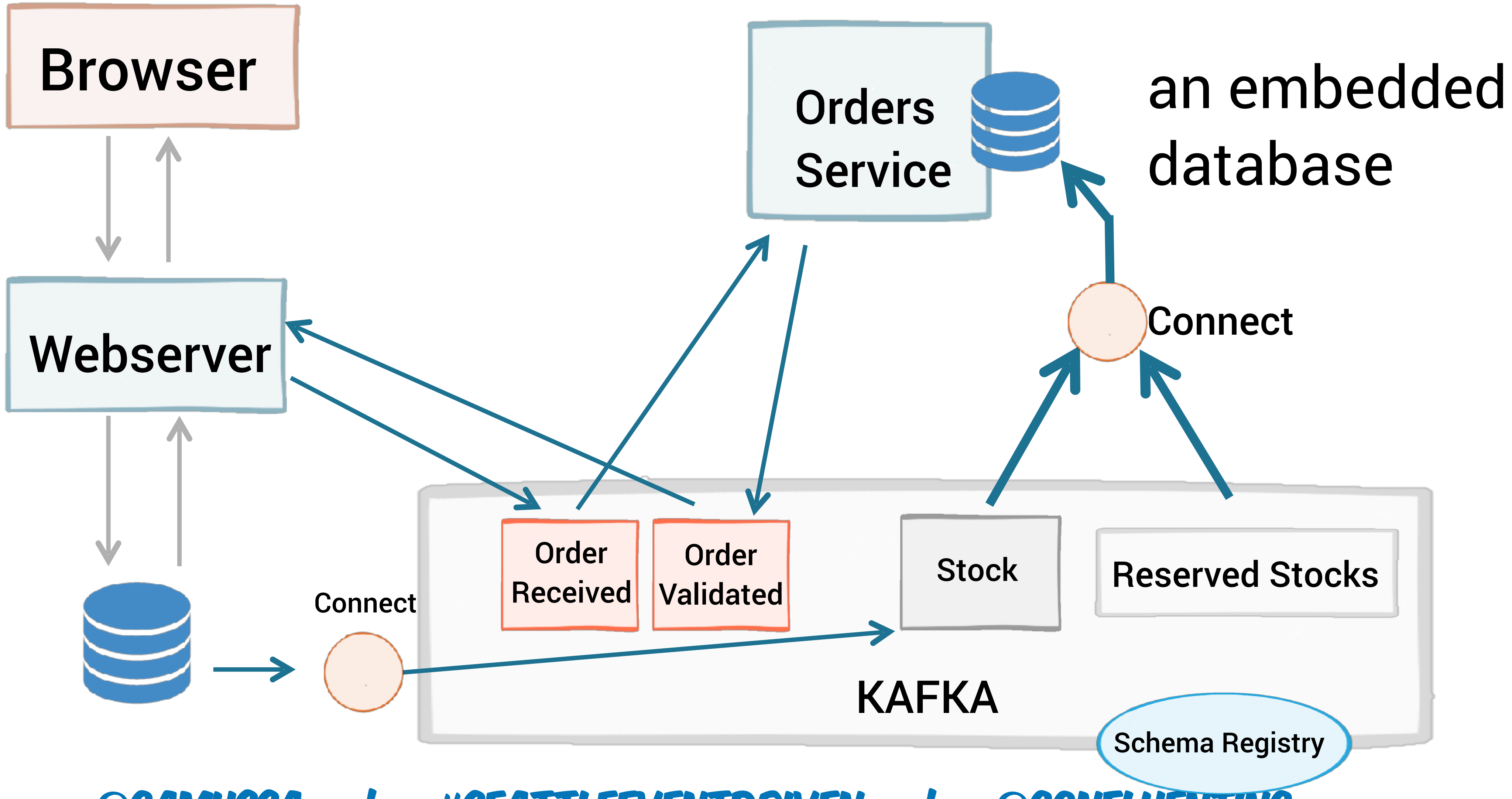


Writable table created for reserved stocks

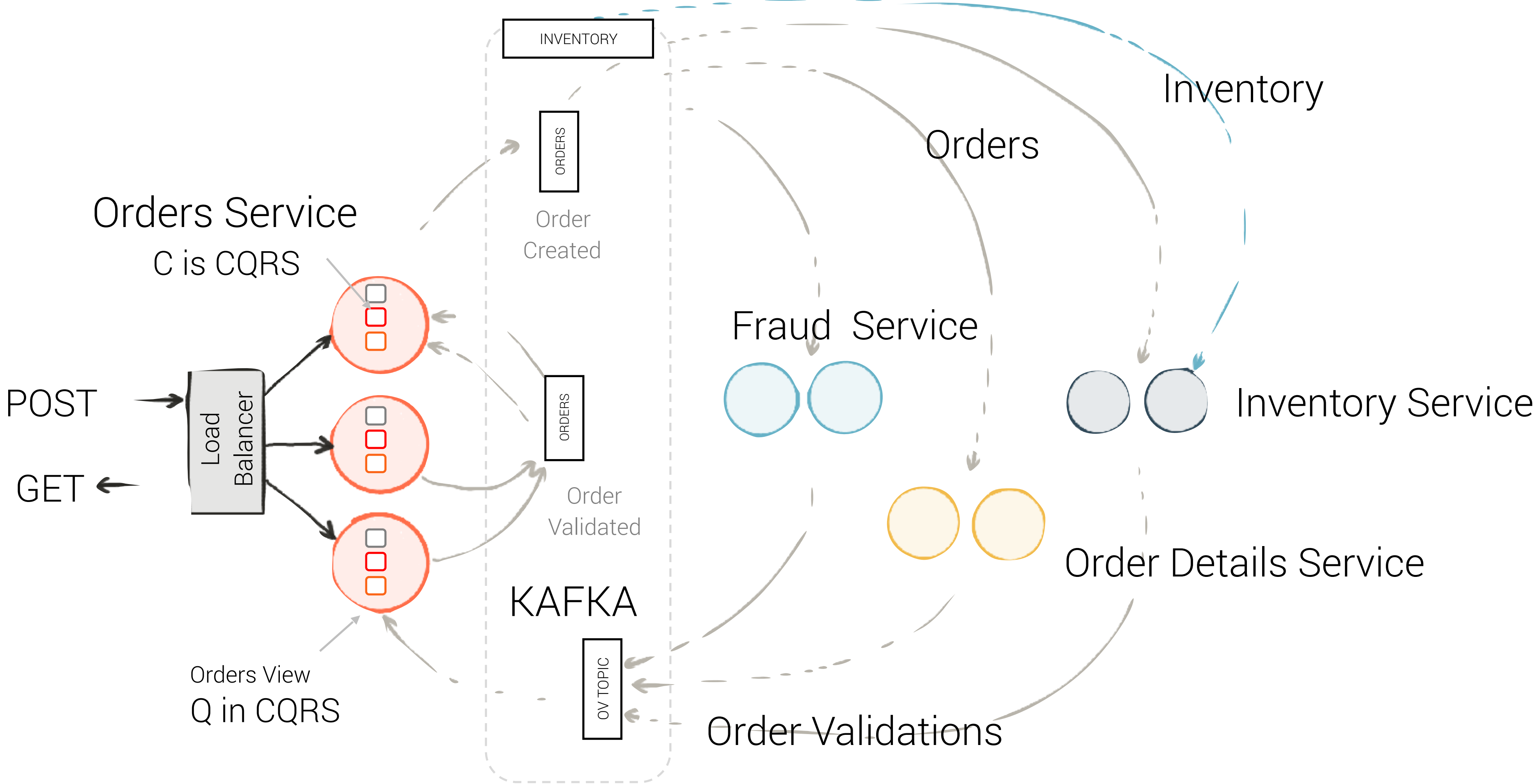
TRANSACTIONAL GUARANTEES ARE YOURS



HYDRATE A MATERIALIZED VIEW



SERVICES IN THE MICRO: ORDERS SERVICE



What's a database anyway?

confluent

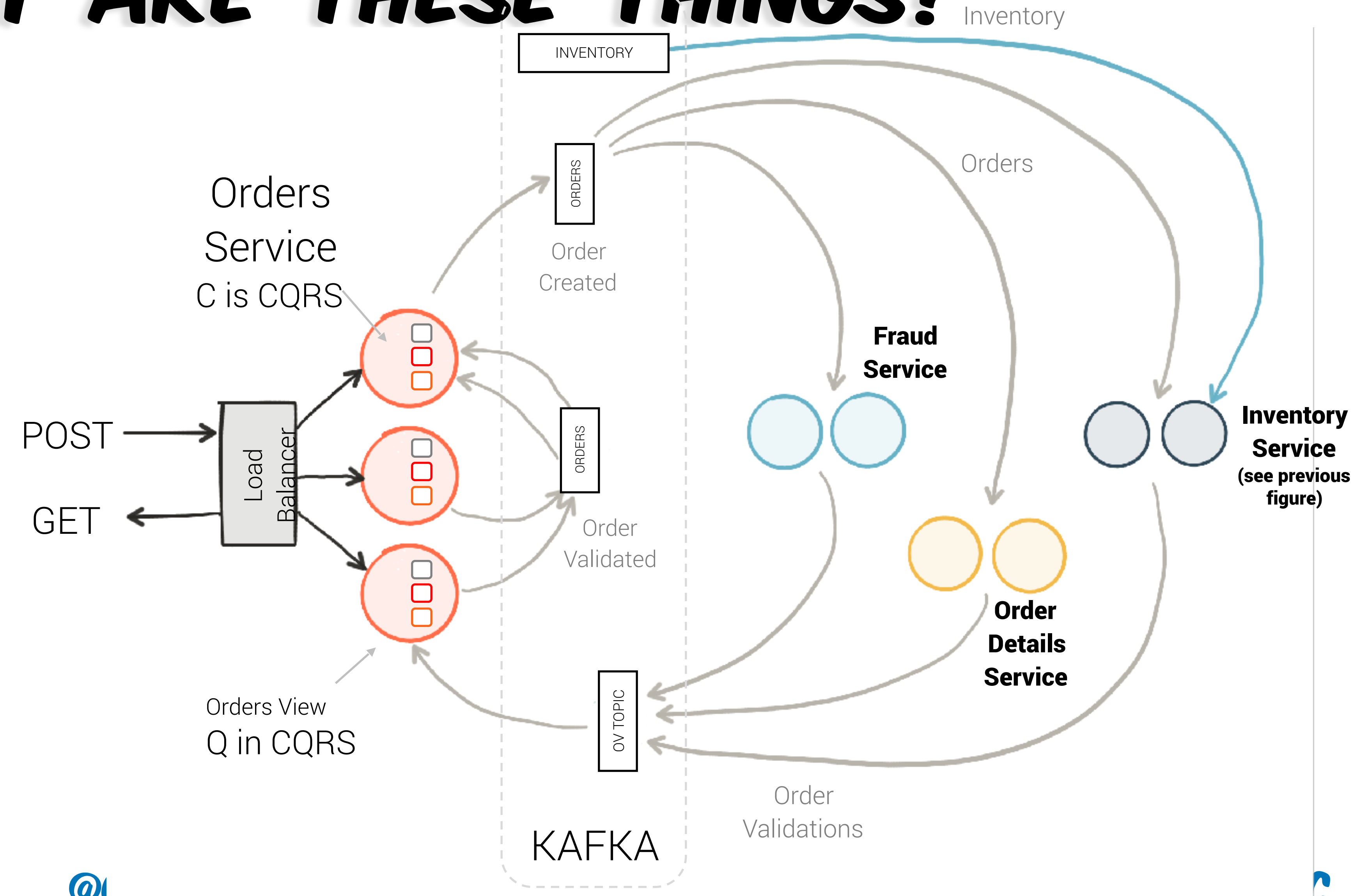
SQL

Tabular Model

Storage Engine

Commit Log

WHAT ARE THESE THINGS?



YOU ARE NOT JUST WRITING MICROSERVICES

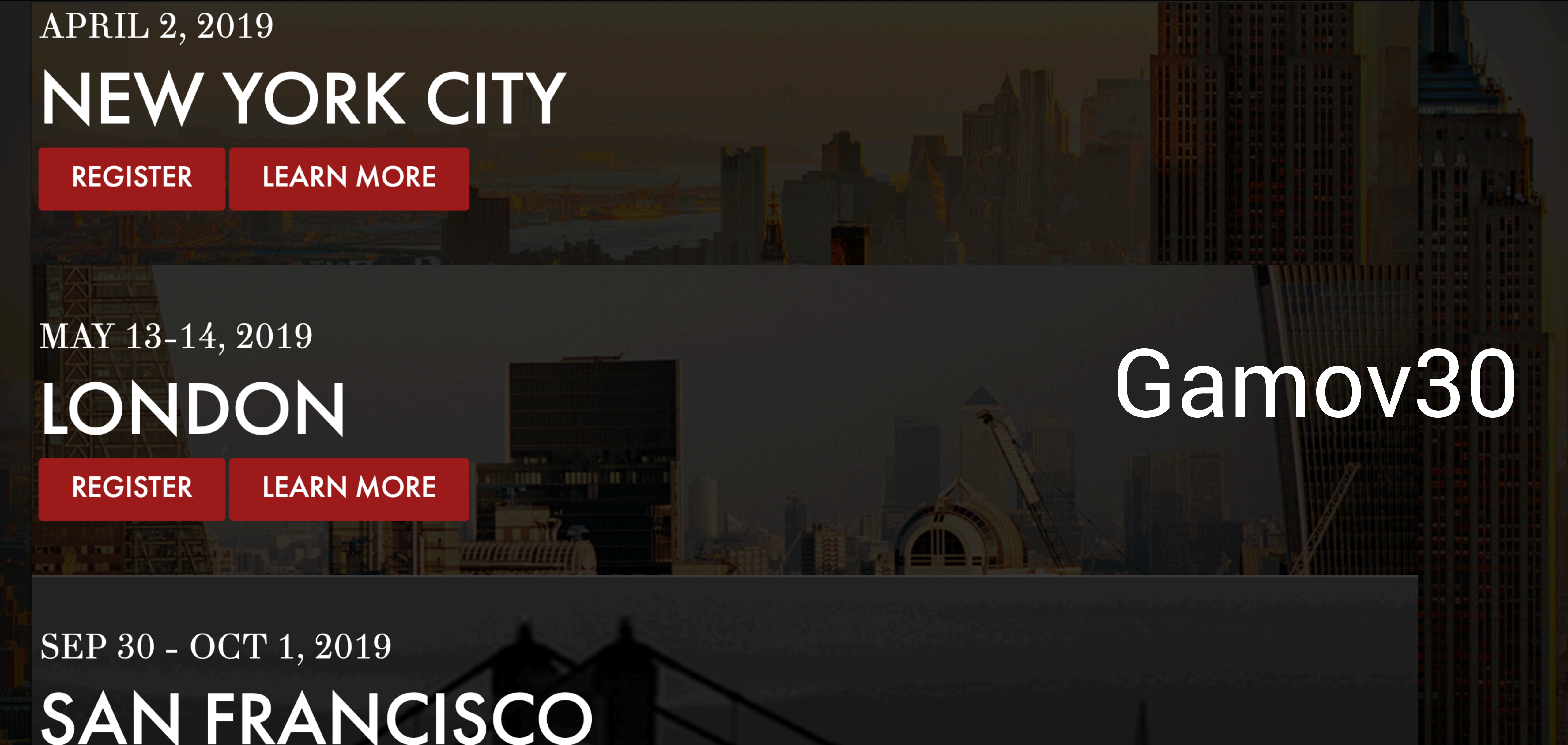
YOU ARE BUILDING AN INSIDE-OUT DATABASE.

AND THAT IS A GOOD THING…

ONE MORE THING...

kafka summit

https://kafka-summit.org



THANKS!

@gamussa
viktor@confluent.io

We are hiring!
https://www.confluent.io/careers/