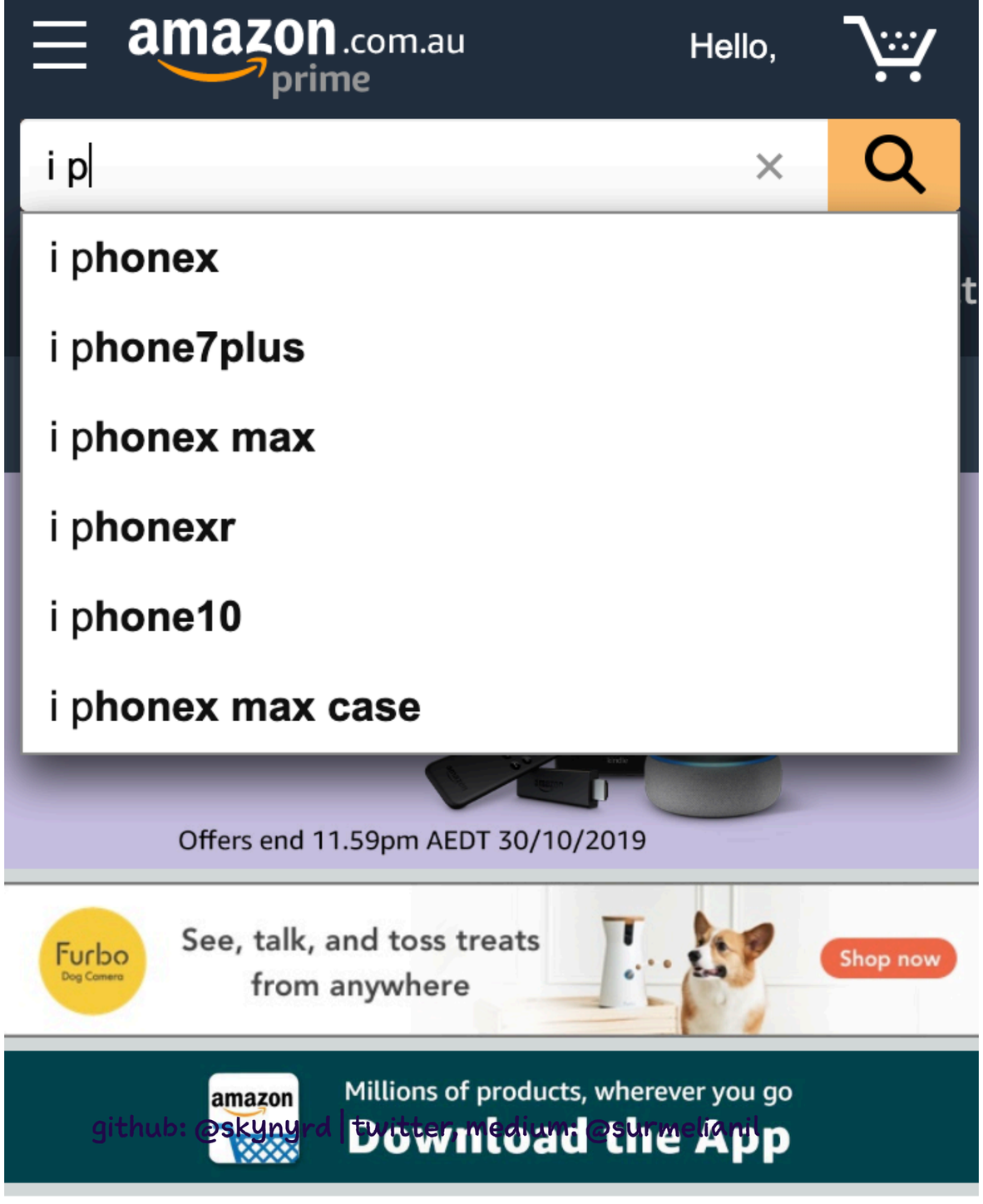
Real-Time Streaming ETL with Kafka Stack

github: @skynyrd | twitter, medium: @surmelianil

The Problem

How fast can we see the latest updates of a product in an e-commerce application?

github: @skynyrd | twitter, medium: @surmelianil



- It must perform well
- You can't cache user input
- It must be durable on high load

Shamo's Case for iPhone 11 Clear Shock Absorption with TPU Bumpers Anti-Scratch...

★★★★☆ 3

$6.99

✓prime Get it by **Thursday, October 24**

Best Seller

Tech Armor Ballistic Glass Screen Protector for Apple iPhone 11 / iPhone Xr - Case-Friendly...

★★★★★ 25

$10.95

✓prime Get it by **Thursday, October 24**

TOZO for iPhone Xs Max Screen Protector 6.5 Inch (2018) [3-Pack] Premium Tempered...

★★★★½ 40

$12.99

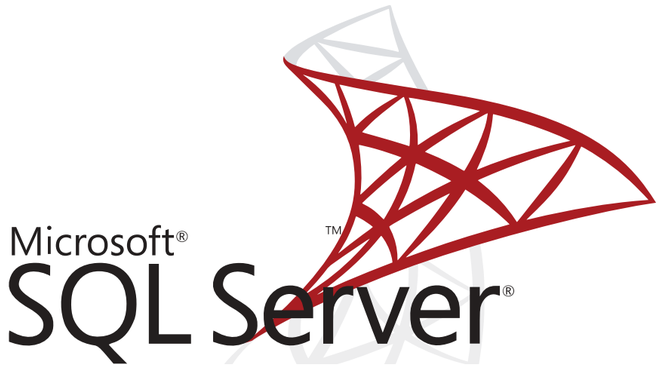
✓prime Get it by **Thursday, October 24**

- Dynamic filters are created with respect to the result set
  - e.g. RAM, Screen Size, Camera etc.
- Also should perform well in milliseconds

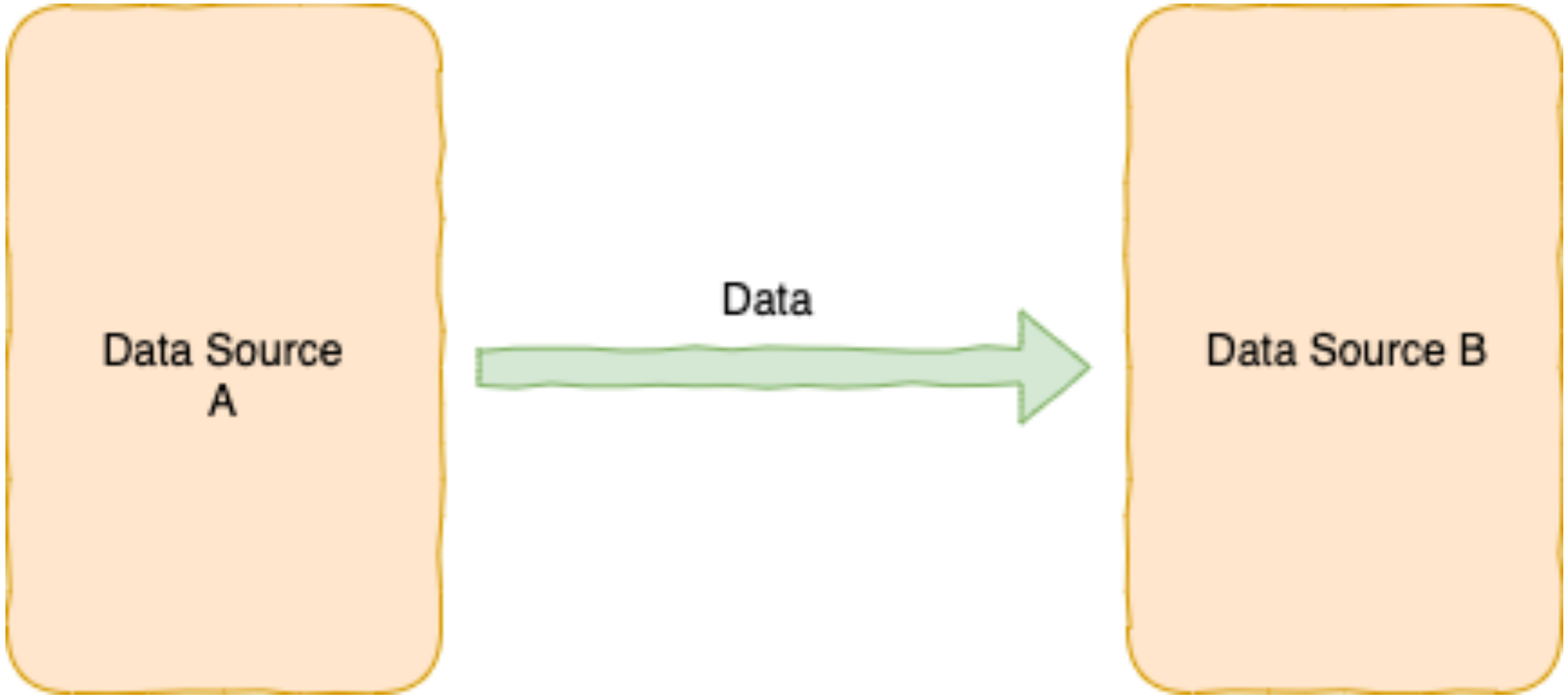
Product information should be retrieved from an additional data store

Additional Data Sources - Designed for search purposes

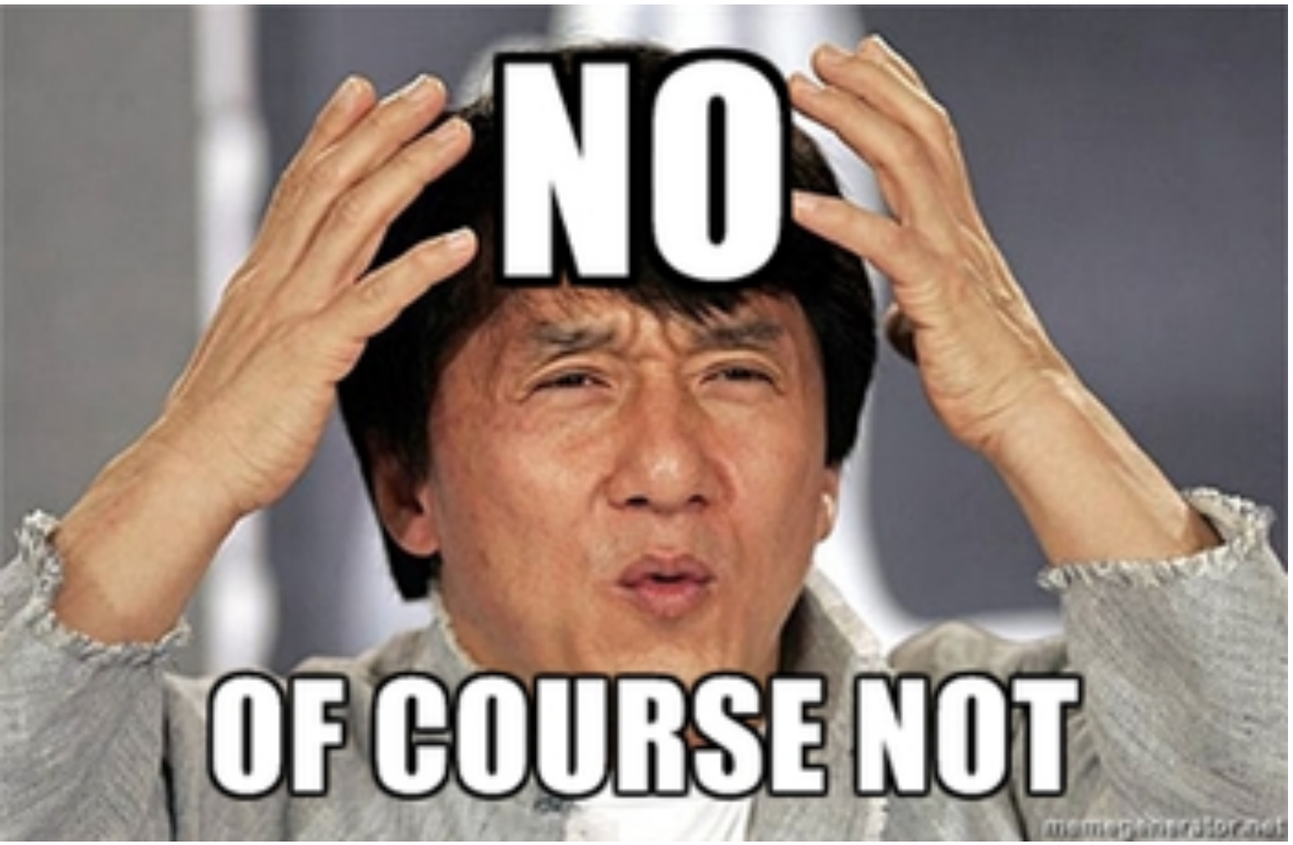
Primary Data Sources



The Common Problem


Data Source
A
Data
Data Source B

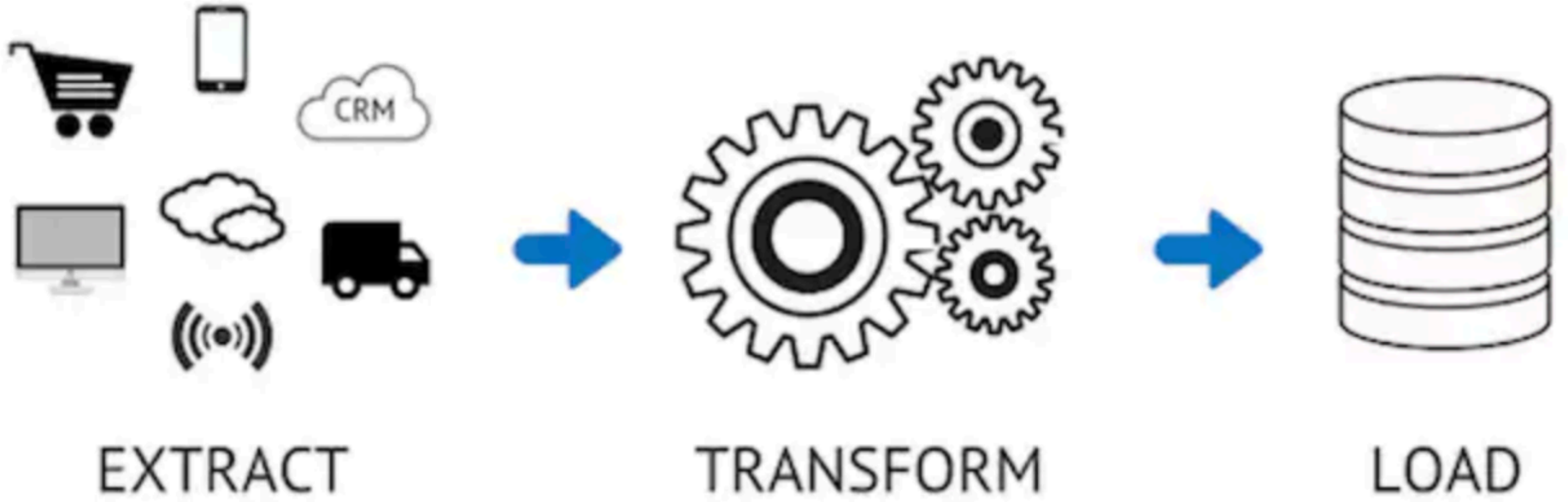
Can we use Data Source A directly ?



Probably;

- SourceA is not designed for your purpose
- SourceA is not horizontally scalable.
- SourceA is in another domain.
- You need to modify/enhance the data in SourceA

ETL



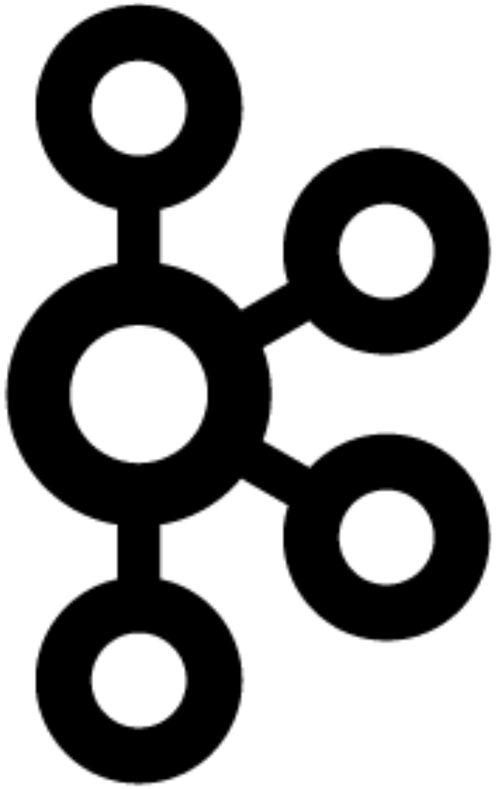
Traditional ETL

* Pull based systems
(e.g. cron jobs, schedulers, pull services)
* Not scalable
* Not real-time

Cool ETL

* Push based
* Real-time streaming
(Less than 20ms)
* Horizontally Scalable

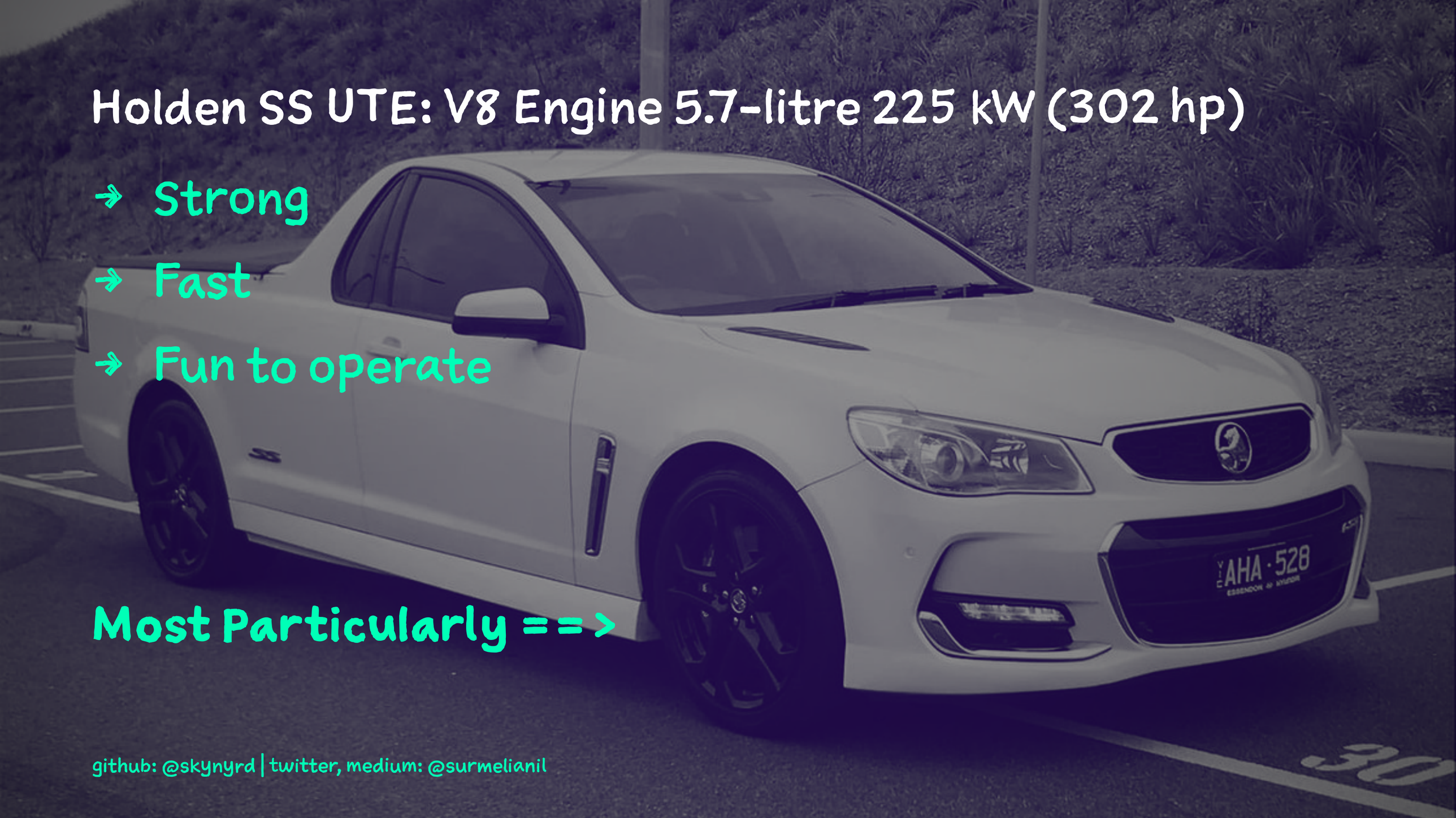
github: @skynyrd | twitter, medium: @surmelianil



kafka

Community distributed streaming platform capable of handling trillions of events a day.

Confluent Docs


Holden SS UTE: V8 Engine 5.7-litre 225 kW (302 hp)
➔ Strong
➔ Fast
➔ Fun to operate
Most Particularly = = >
AHA·528
github: @skynyrd | twitter, medium: @surmelianil

You can tune it!

Horizontuning

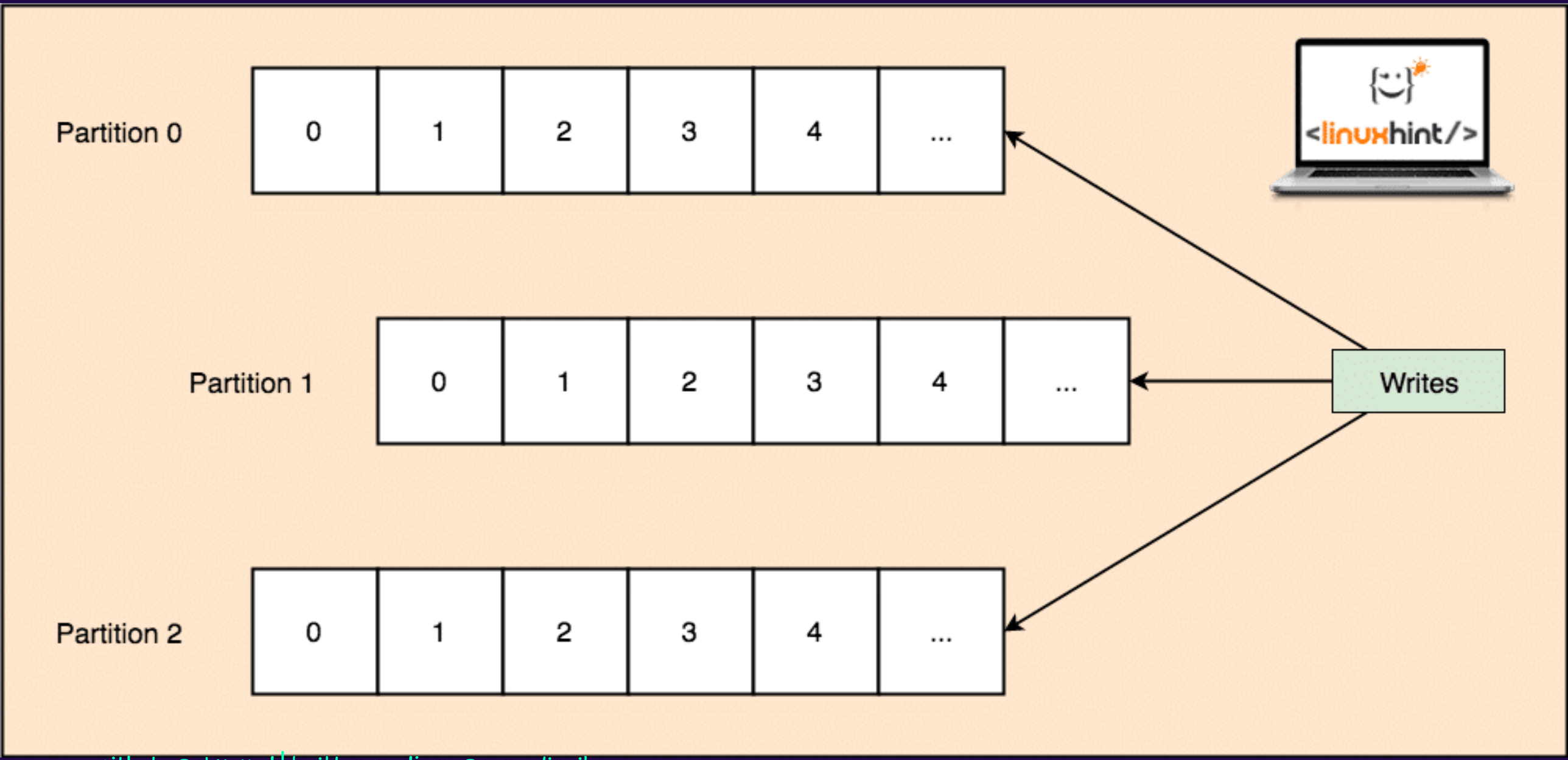
github: @skynyrd | twitter, medium: @surmelianil

Kafka World

Topics, Partitions, Brokers, Zookeeper, Producers, Consumers, Kafka Stream and Kafka Connect

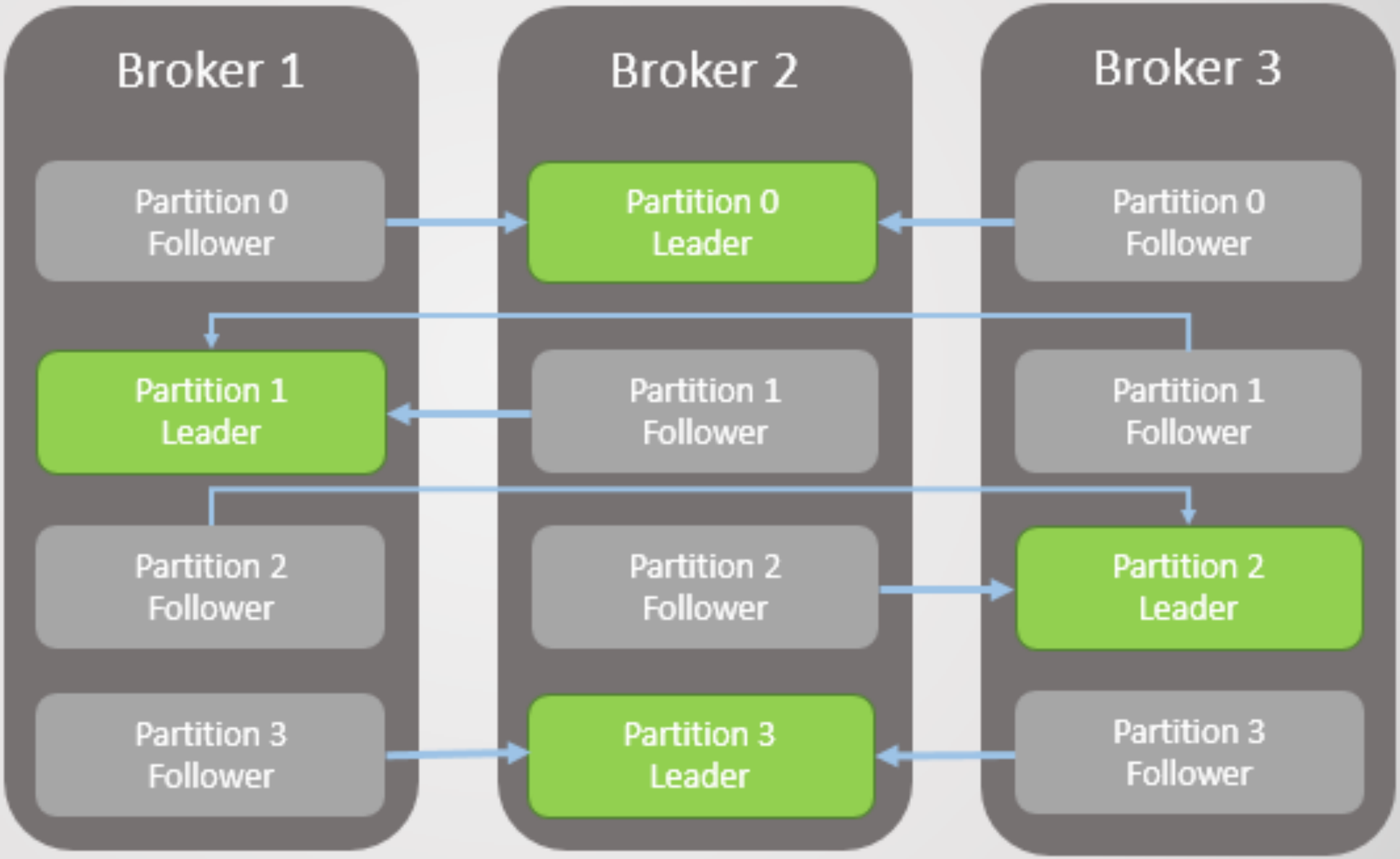
github: @skynyrd | twitter, medium: @surmelianil

Topics: Stream of data. (Mongo collection, SQL table, Elastic index)


Partition 0
0
1
2
3
4
...
<linuxhint/>
Partition 1
0
1
2
3
4
...
Writes
Partition 2
0
1
2
3
4
...

Brokers: Cluster Nodes. Connecting to any broker means connecting to cluster.

Topic with:
- 4 partitions
- Replication factor = 3



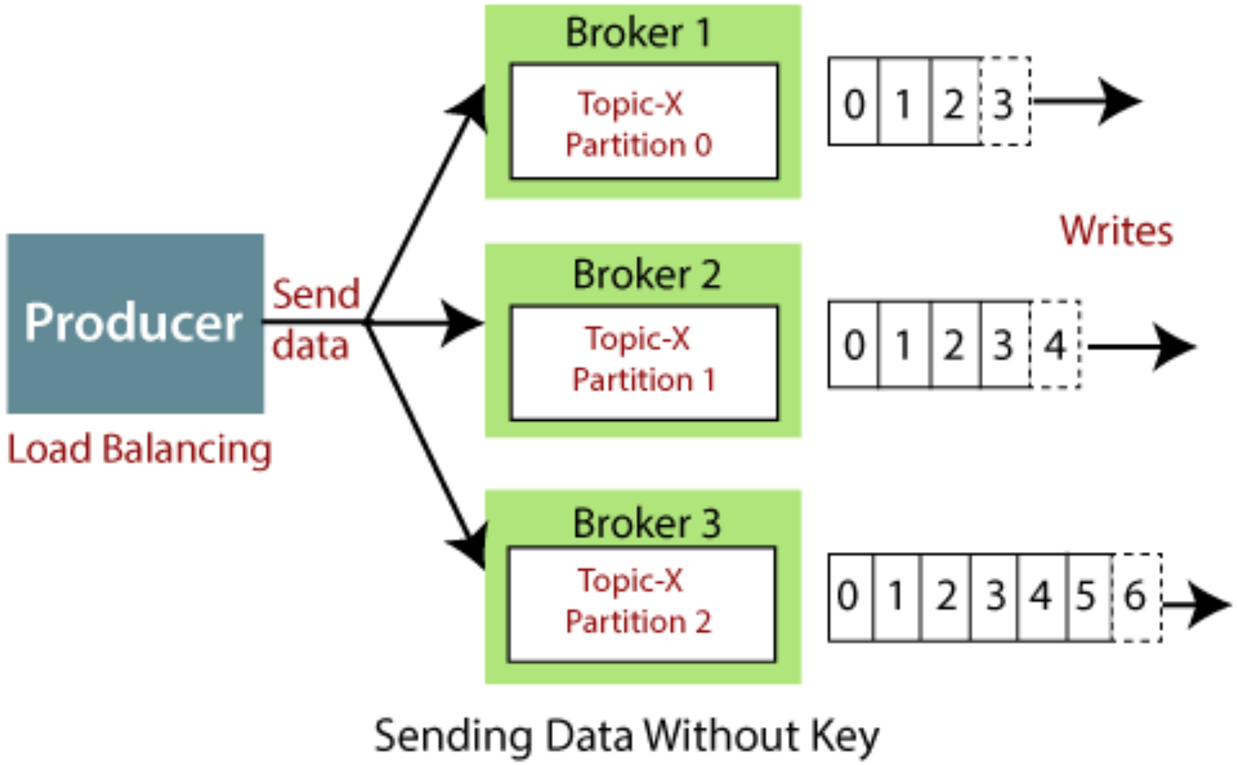
Zookeeper

- Apache project
- Enables reliable distributed coordination
- Brokers are managed by Zookeeper
- Working in a seperate cluster
- No Zookeeper, no Kafka

APACHE ZooKeeper™

Producers

- Services that sends data to Kafka
- When broker fails, produces can automatically recover it
- All major languages provide the API



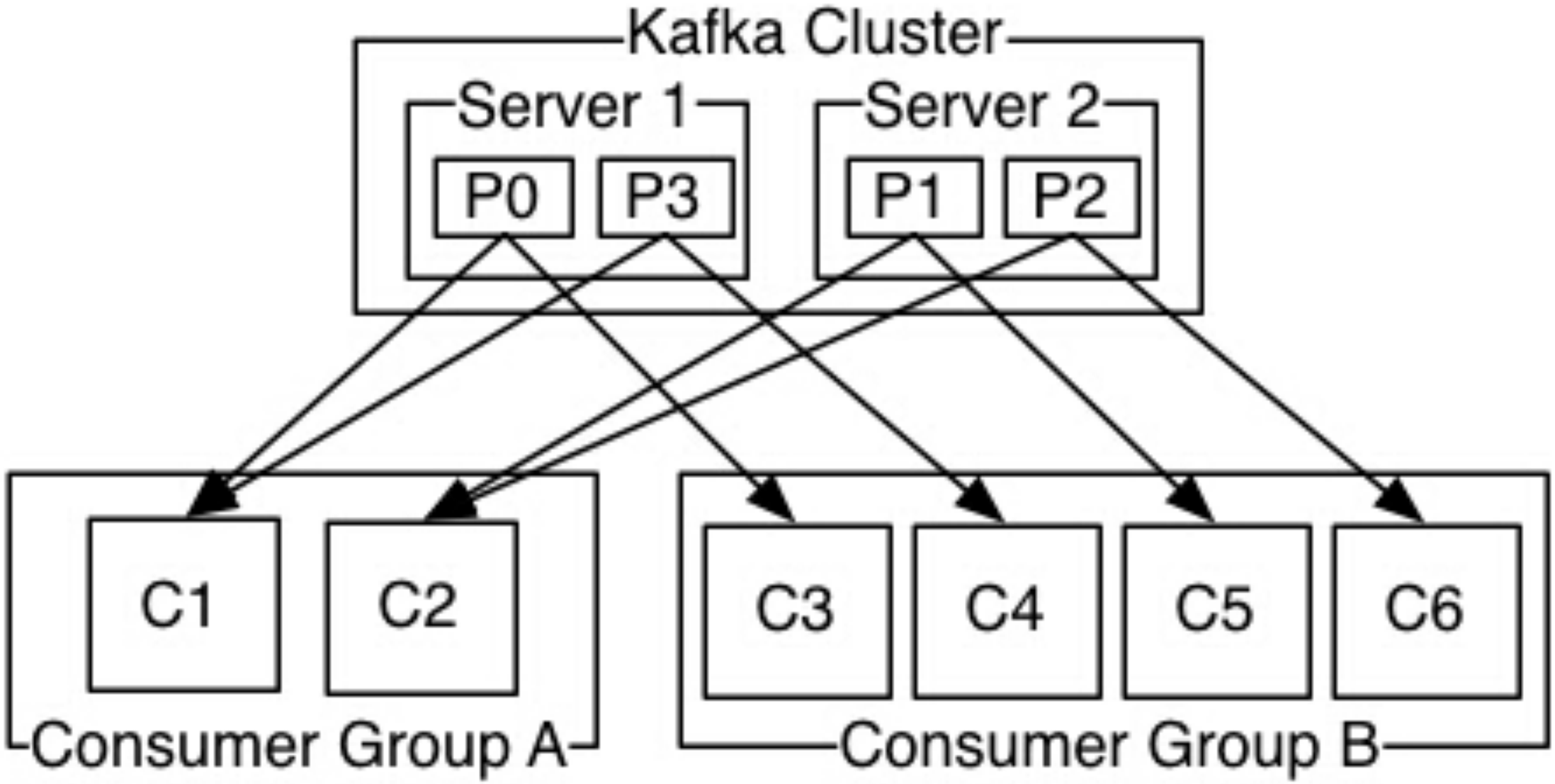
Sending Data Without Key

Acknowledgement Strategies

- acks=0 : Fire and forget
- acks=1 : Get ack message from leader partition
- acks=all: Get ack messages from leader and replica partitions

Consumers

- Services that reads data from Kafka
- When broker fails, consumers can automatically recover it
- Works in a consumer group
- Consumer group offsets are stored as a state (in `__consumer_offsets` internal topic)


Kafka Cluster
Server 1
Server 2
P0
P3
P1
P2
C1
C2
C3
C4
C5
C6
Consumer Group A
Consumer Group B

Delivery Semantics

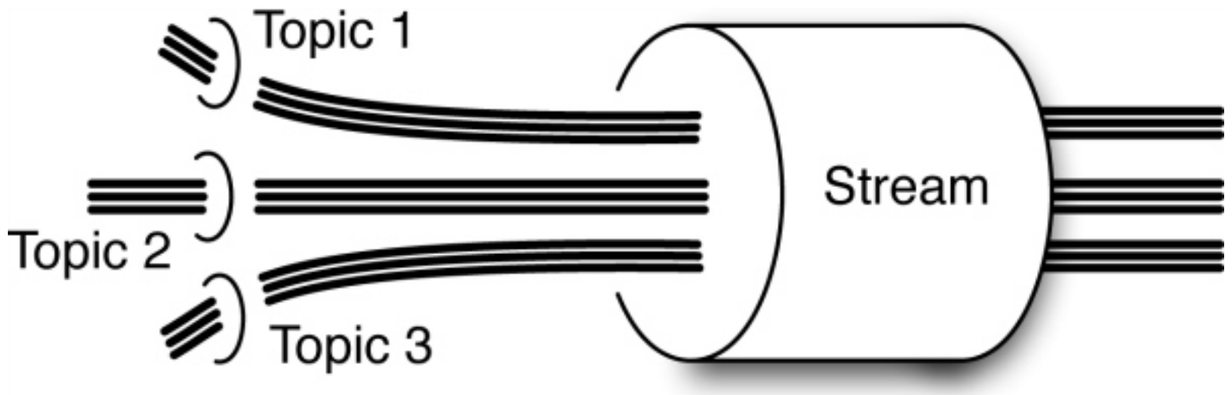
- At least once: offsets are commited when the message is received.
- At most once: offsets are commited after message is received and processed
- Exactly once: Only available for Kafka streams

Kafka Streams

Client library for building apps and services, where the input and output data is stored in Kafka Cluster. (Only for Java and Scala)

github: @skynyrd | twitter, medium: @surmelianil

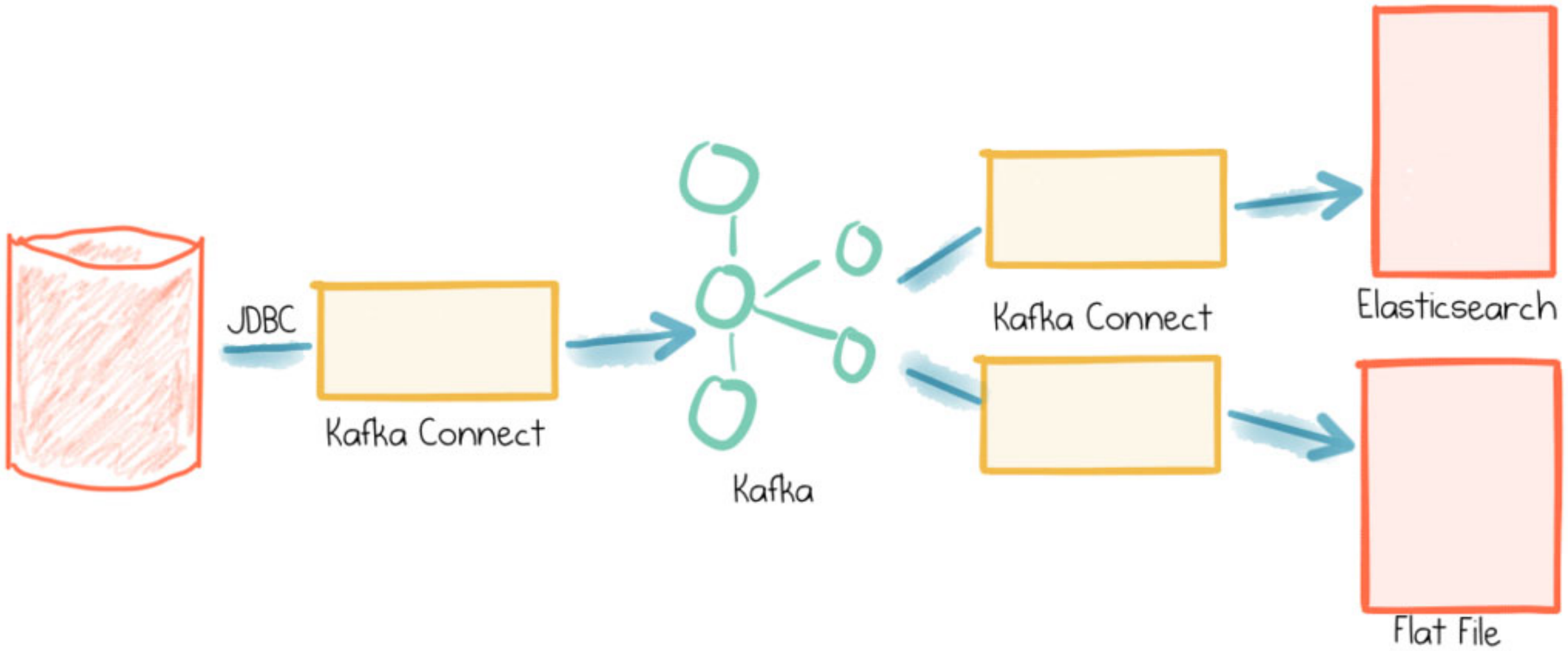
- Encapsulates reading and writing process (Producer + Consumer)
- ”T” in ETL, Transforming the objects
- Reads from a topic/topics, processes it and stores it in a seperate topic.



Kafka Connect

github: @skynyrd | twitter, medium: @surmelianil

- Open source component
- Framework to connect external systems such as databases, file systems, twitter?
- `Source` for reading, `Sink` for writing
- Manages the offsets for you, uses Kafka Producers/ Consumers internally
- Main advantage: You can use existing implementations



- May be used instead of Consumers and Producers

The image from here.

Writing Your Own Sink Connector for Your Kafka Stack

Arjun V

Anil Selim Surmeli

Sep 17, 2017 · 5 min read

Thiên Toa

There can be no Kafka Connector for your system, or available ones may not meet your requirements. On both cases, you have to write your own Kafka Connector and there are not many online resources about it. I'll try to write my adventure to help others suffering with the same pain.

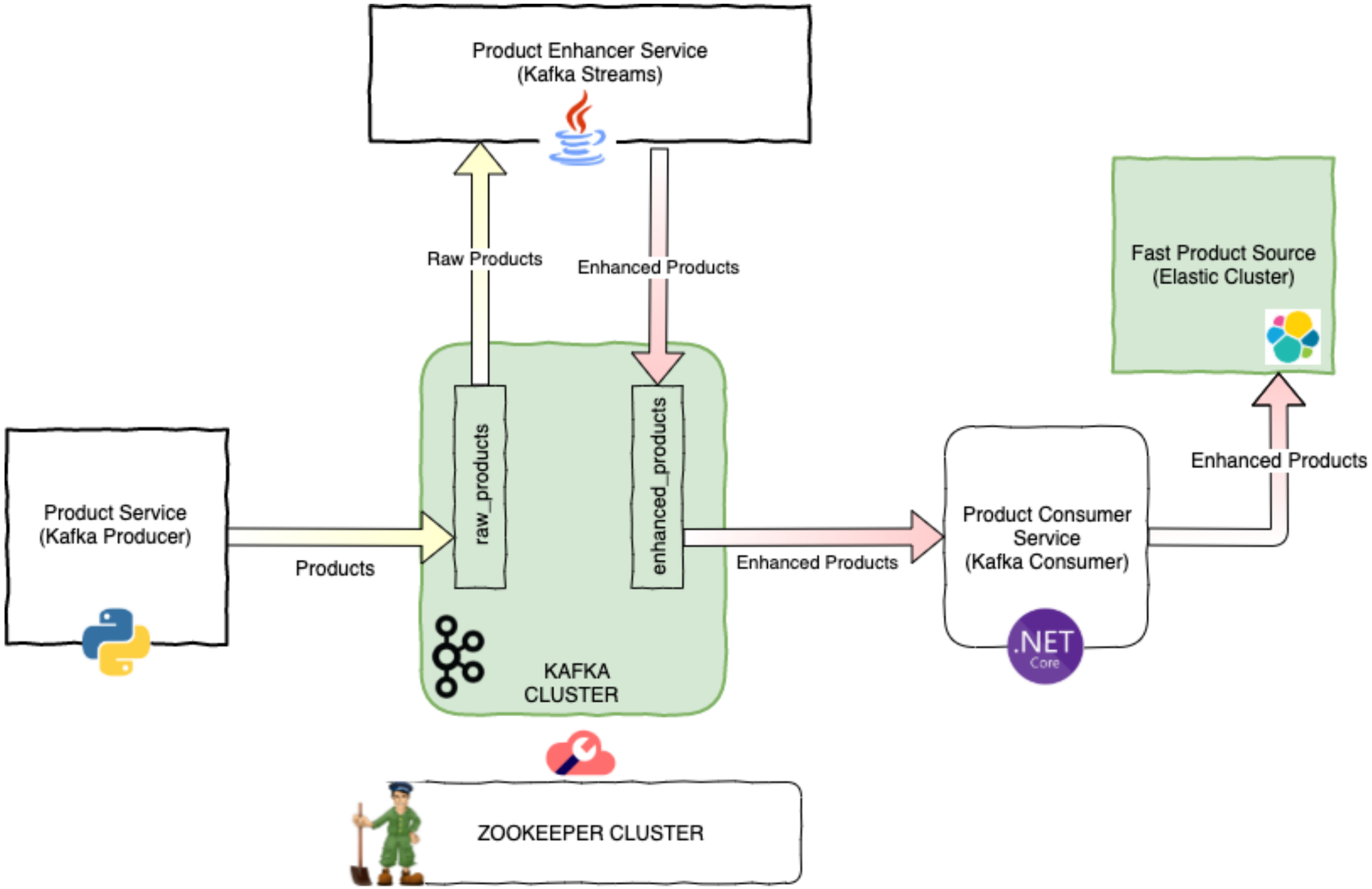
Why We Replaced Our Kafka Connector with a Kafka Consumer

Anil Selim Surmeli Follow
Dec 16, 2017 · 3 min read

A few months ago, I wrote about creating your own sink connector after we started using ours. Surprisingly, we replaced it with Kafka Consumers last week. I am going to review our experience and try to write the advantages and disadvantages of both technologies in this short article.

github: @skynyrd | twitter, medium: @surmelianil

Demo Time!


Product Enhancer Service
(Kafka Streams)
Raw Products
Enhanced Products
Fast Product Source
(Elastic Cluster)
Product Service
(Kafka Producer)
Products
raw_products
enhanced_products
Enhanced Products
Product Consumer
Service
(Kafka Consumer)
Enhanced Products
KAFKA
CLUSTER
.NET
Core
ZOOKEEPER CLUSTER

References & Image URLs

* Stephane Maarek Courses and Articles
* https://linuxhint.com/apache-kafka-partitioning/
* https://jack-vanlightly.com/blog/2018/9/2/rabbitmq-vs-kafka-part-6-fault-tolerance-and-high-availability-with-kafka
* https://www.javatpoint.com/apache-kafka-producer
* http://javabender.blogspot.com/2017/03/kafka-basics-producer-consumer.html
* https://www.talend.com/resources/what-is-etl/
* https://mapr.com/blog/apache-kafka-and-mapr-streams-terms-techniques-and-new-designs/assets/streaming-post1.jpg
* https://www.confluent.io/blog/simplest-useful-kafka-connect-data-pipeline-world-thereabouts-part-1/
* http://valleymustangsunlimited.com/

Many Thanks!

github: @skynyrd | twitter, medium: @surmelianil