Kafka 101

#mlsystemdesign_2022_ods

Mark Andreev
mark.andreev@gmail.com

Why we need queues?

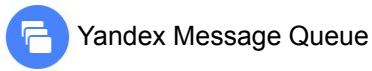
- One event producer for multiple consumers
- Asynchronous processing workflows
- Task complex evaluation rules
- Aggregation in streaming data

Why we need queues?

- One event producer for multiple consumers
- Asynchronous processing workflows
- Task complex evaluation rules
- Aggregation in streaming data

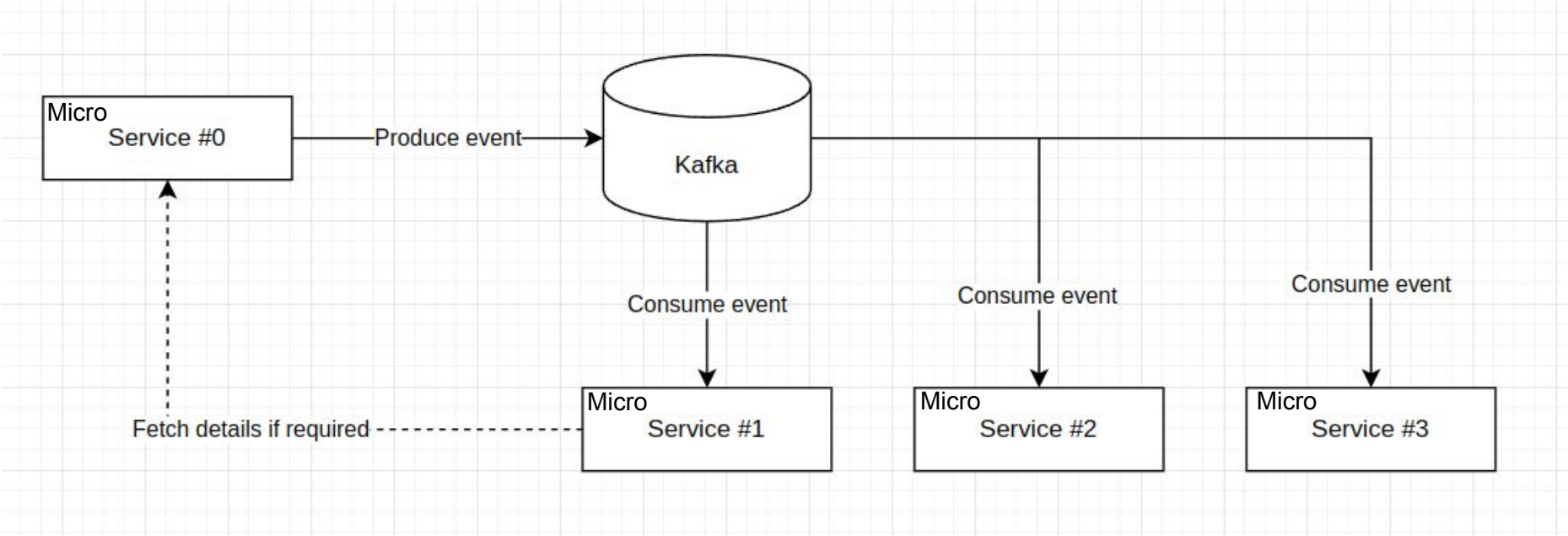
Event processing

Task processing

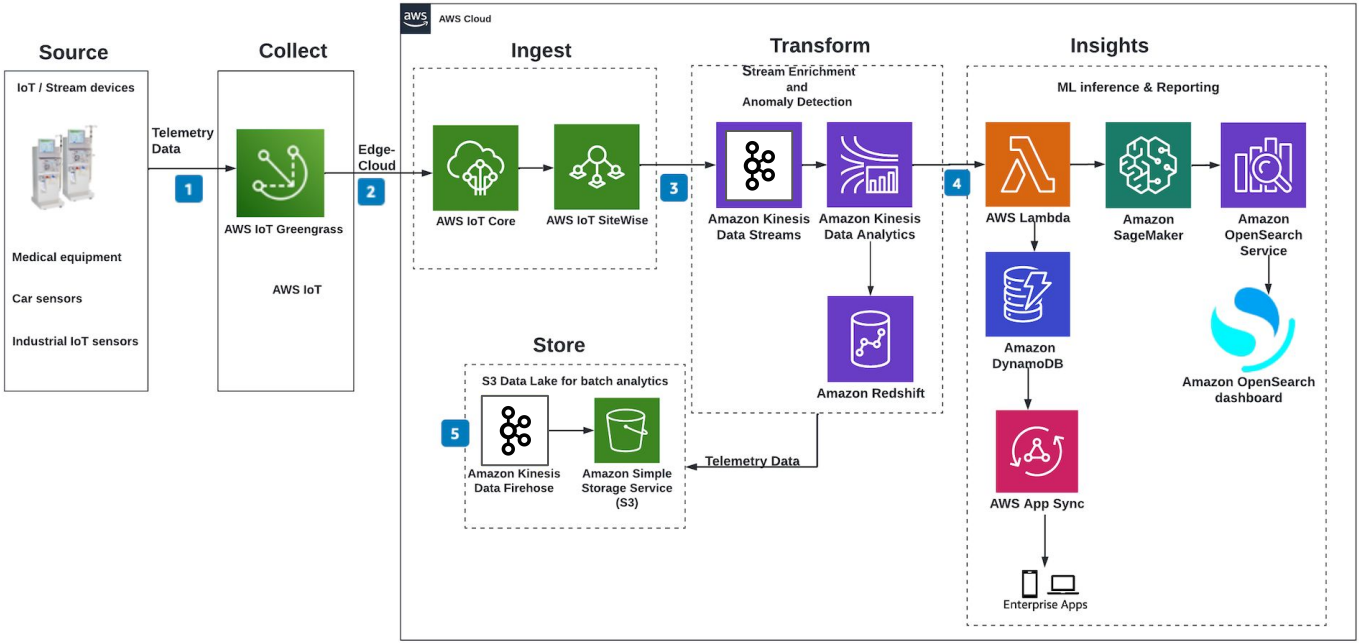


Use cases

Use case: Event-Driven Architecture

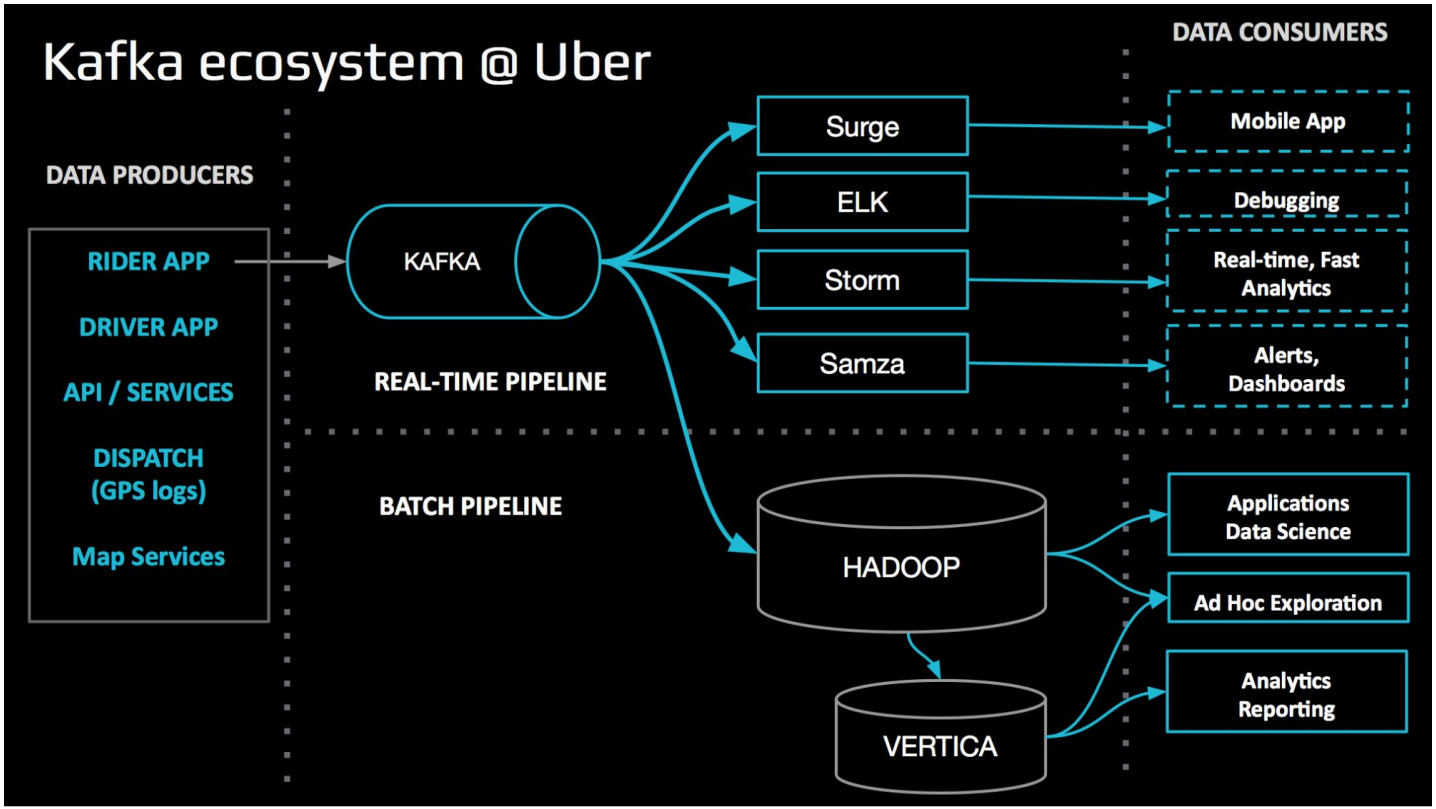


Building event-driven architectures with IoT sensor data

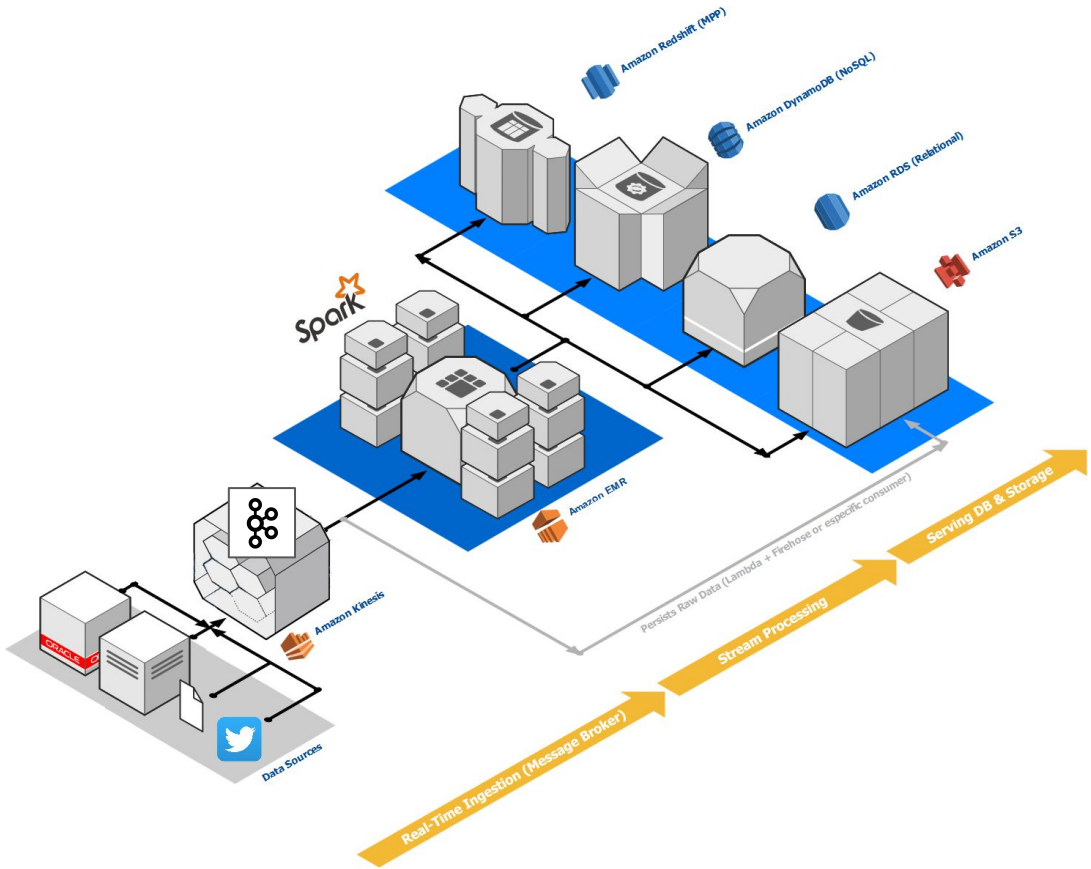


Ref: https://aws.amazon.com/ru/blogs/architecture/building-event-driven-architectures-with-iot-sensor-data/

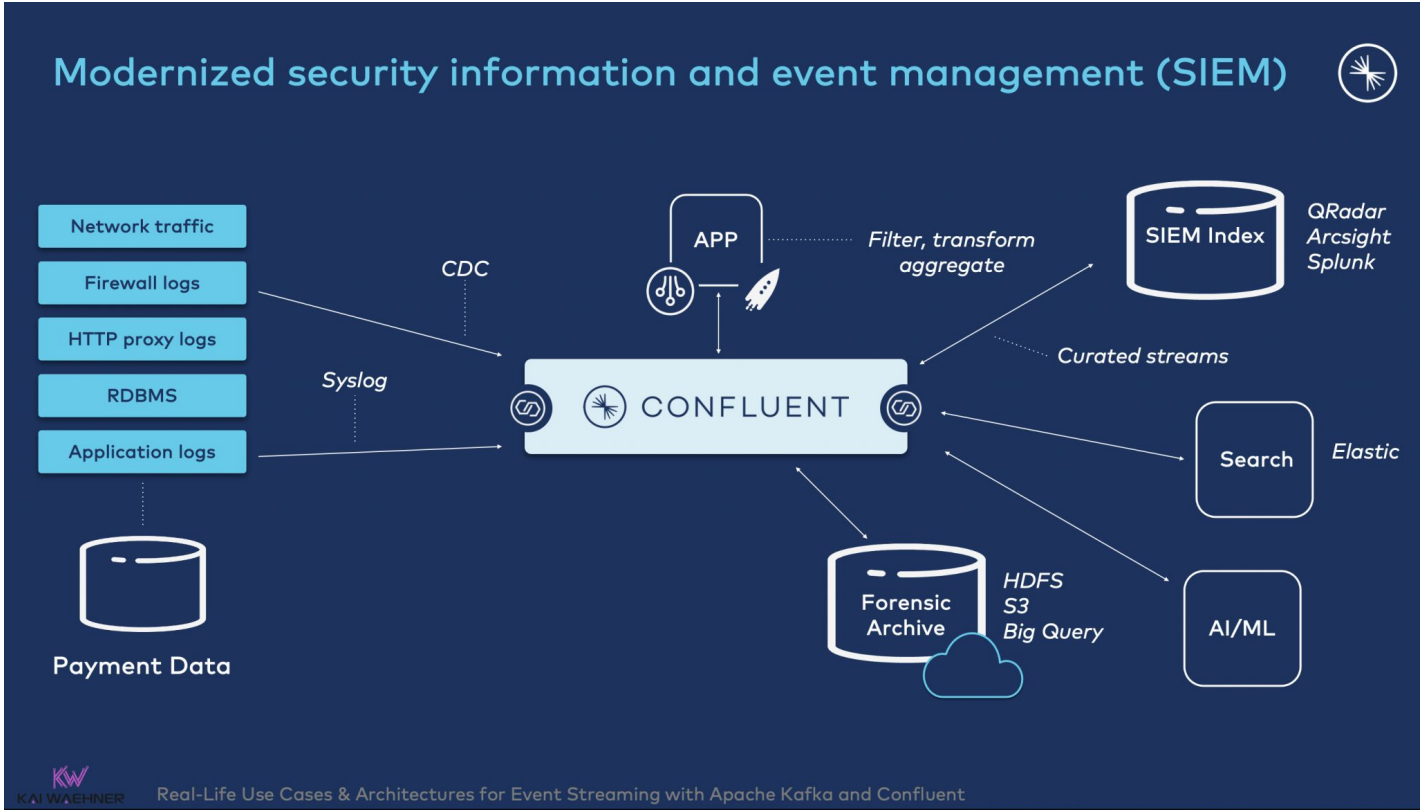
Use case: Event processing



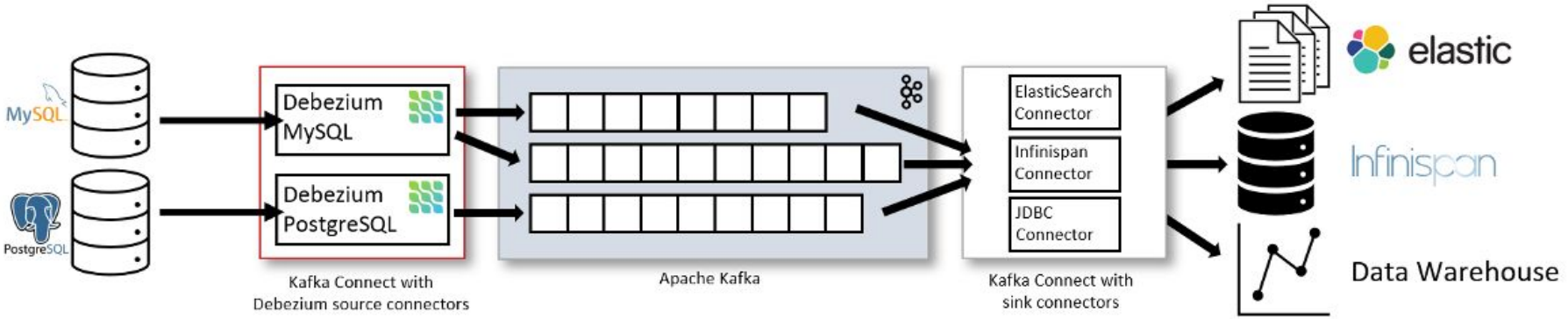
Use case: Event processing



Use case: SIEM. Security information and event management



CDC with Kafka



Use cases: Summary

1. Event driven architecture (share events)
2. Aggregate streaming data

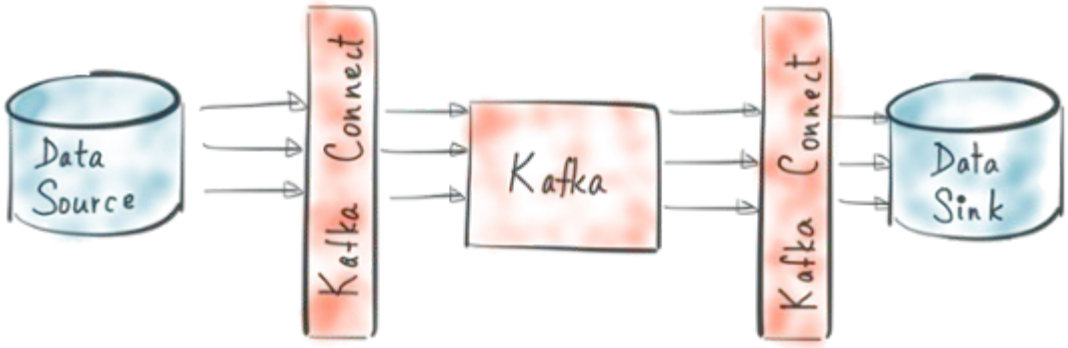
Tools for Kafka

Tools for Kafka

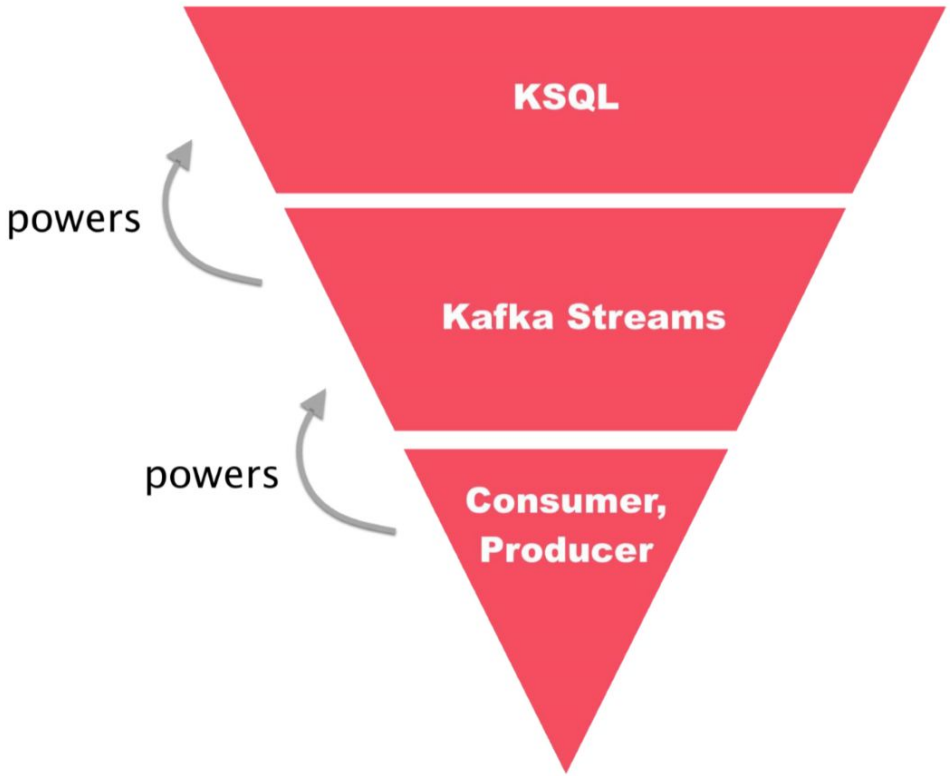
- Confluent schema registry
- Kafka Connect
- Kafka Streams
- KSQL
- Apache Spark
- Apache Flink
- Apache Beam

CONFLUENT

KAFKA CONNECT



Kafka Streams / KSQL



ease of use

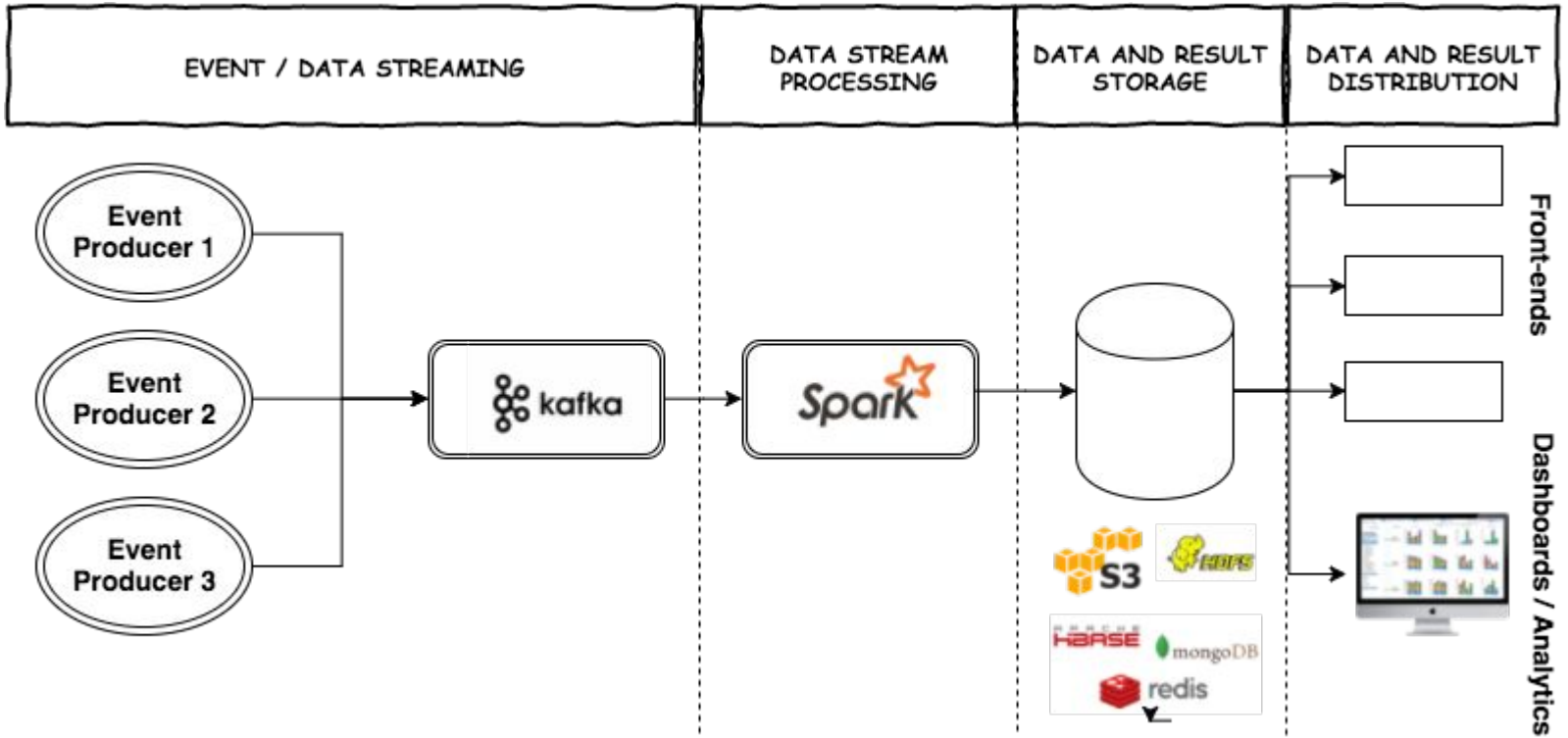
```
CREATE STREAM, CREATE TABLE,
SELECT, JOIN, GROUP BY, SUM, …
```

```
KStream, KTable,
filter(), map(), flatMap(),
join(), aggregate(), …
```

```
subscribe(), poll(), send(),
flush(), beginTransaction(), …
```

flexibility

Spark with Kafka



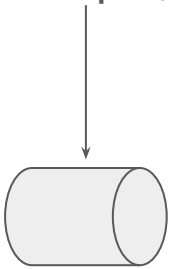
Kafka Inside

Message structure

<Topic, (Key, Value)>

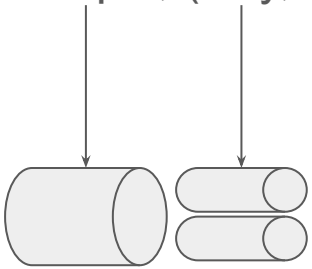
Message structure

<Topic, (Key, Value)>



Message structure

<Topic, (Key, Value)>



Specify partition = murmur2(key) % partition counts | round_robit if key is null

Message structure

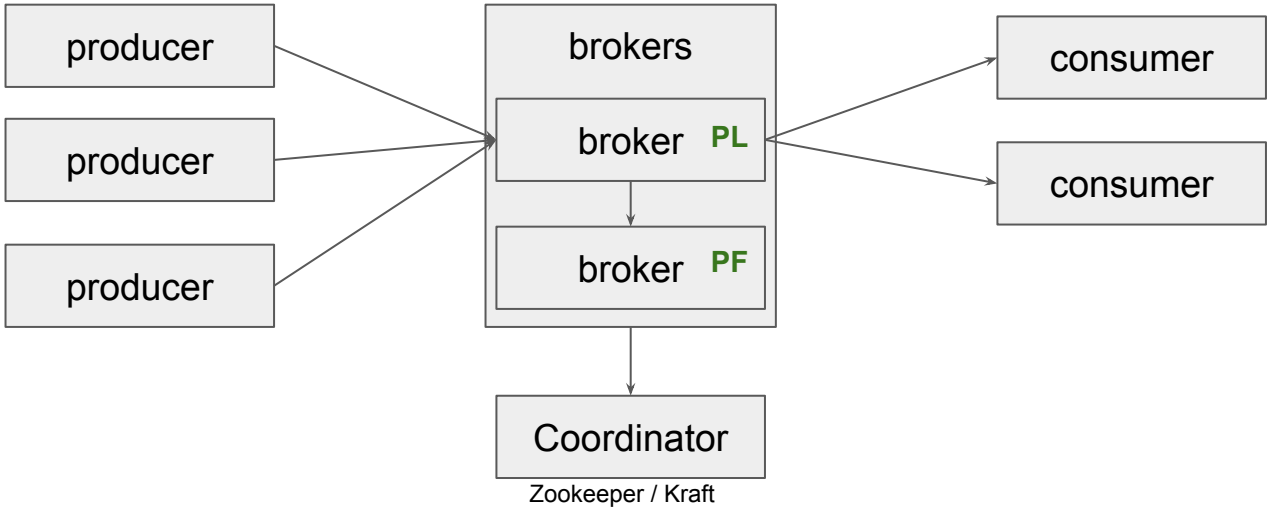
<Topic, (Key, Value)>

Max.request.size ~ 1MB

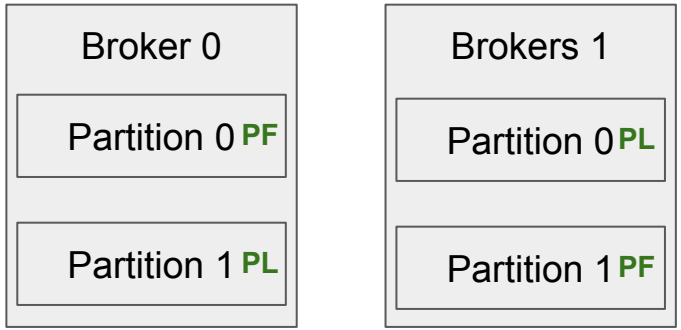
ACK

- 0
- 1
- all

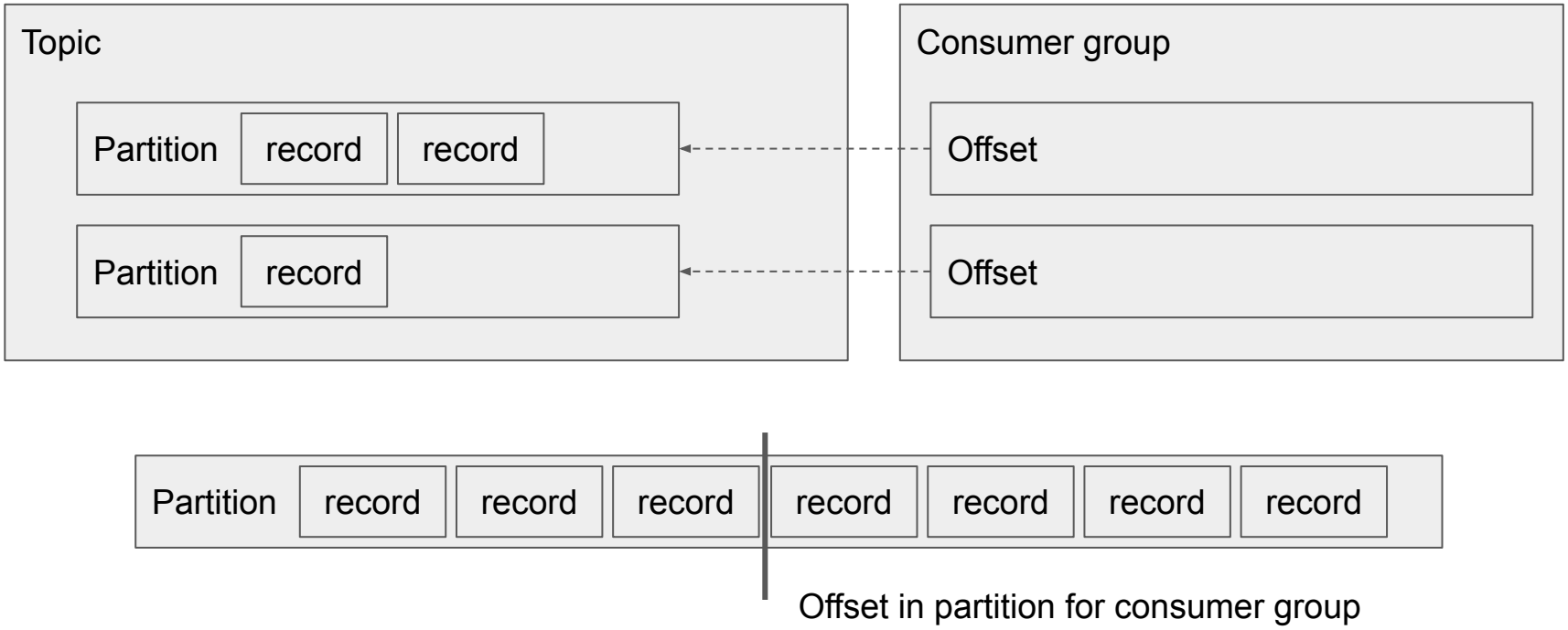
Producer / Consumer / Broker / Coordinator



Producer / Consumer / Broker / Coordinator



Topic / Consumer Group



Anti Patterns

- Consumer in consumer group more than partitions count*
- Request / Response
- Single partition topics
- 100500+ topics / partitions
- Topic full scan
- Topic auto create*

Example: Python

```python
from confluent_kafka import Consumer

consumer_conf = {}
consumer_conf['group.id'] = 'python_example_group_1'
consumer_conf['auto.offset.reset'] = 'earliest'
consumer = Consumer(consumer_conf)

consumer.subscribe(['topic'])

try:
    while True:
        msg = consumer.poll(1.0)
        if msg is None:
            print("Waiting for message or event/error in poll()")
            continue
        elif msg.error():
            print('error: {}'.format(msg.error()))
        else:
            record_key = msg.key()
            record_value = msg.value()
            print(record_key)
            print(record_value)
except KeyboardInterrupt:
    pass
finally:
    consumer.close()
```

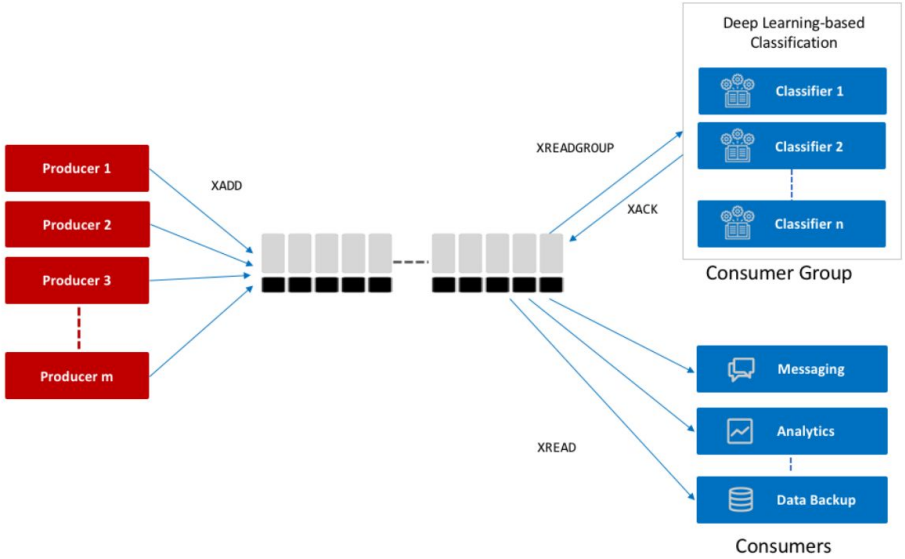
```python
import json

from confluent_kafka import Producer

producer_conf = {}
producer = Producer(producer_conf)


def acked(err, msg):
    """Delivery report handler called on
    successful or failed delivery of message
    """
    if err is not None:
        print("Failed to deliver message: {}".format(err))
    else:
        print("Produced record to topic {} partition [{}] @ offset {}"
              .format(msg.topic(), msg.partition(), msg.offset()))


for n in range(10):
    record_key = "alice"
    record_value = json.dumps({'count': n})
    print("Producing record: {}\t{}".format(record_key, record_value))
    producer.produce('topic', key=record_key, value=record_value, on_delivery=acked)

producer.flush()
```

Kafka Alternative

[ref]

Redis Streams

- In memory db for streams
- Easy to start
- No durability



Resources

- RabbitMQ 101, Pavel Filonov [ref]
- How to choose the right queue, Vladimir Perepelitsa [ref]
- Event-Driven Architecture, Nikolay Golov [ref]