Workshop Kafka Streams

Présentation

Titre

Staff Software Engineer @ManoMano

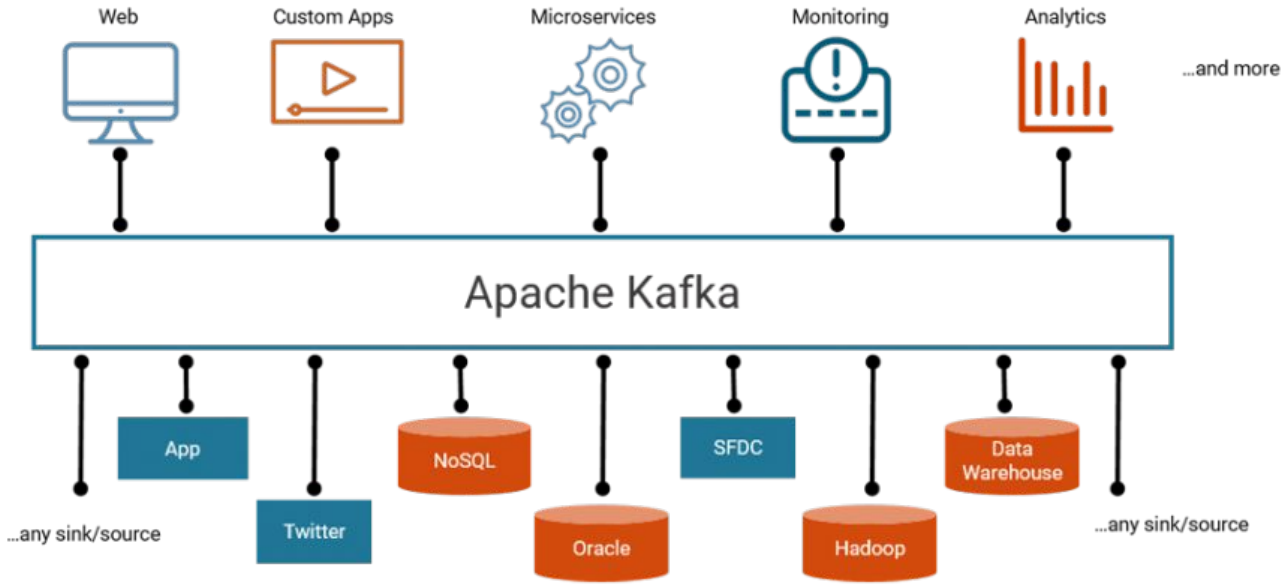
Confluent Certified Trainer

victor.gallet@manomano.com
https://vgallet.github.io/
@GalletVictor

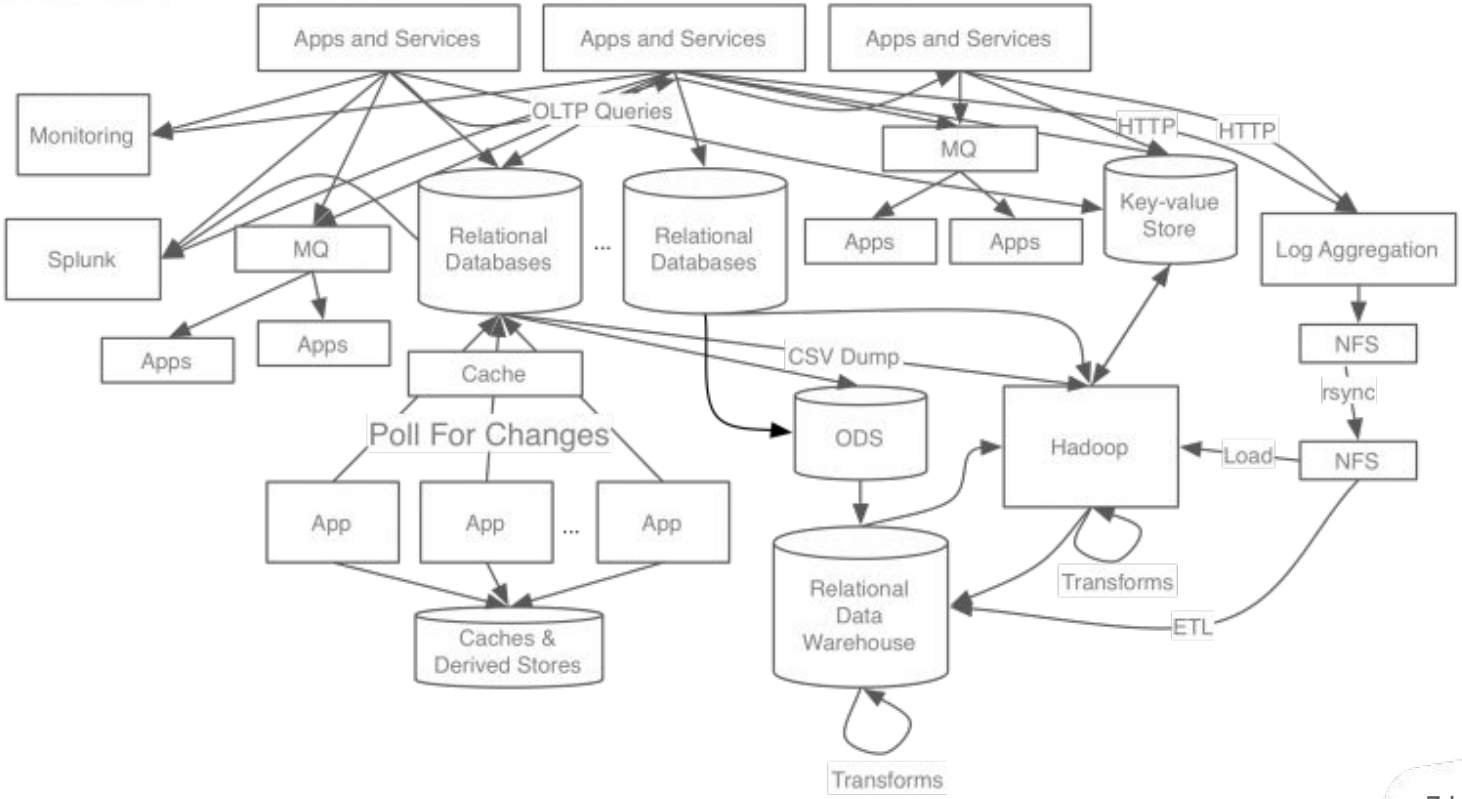
https://github.com/vgallet/workshop-kafka-streams

Kafka Fundamentals

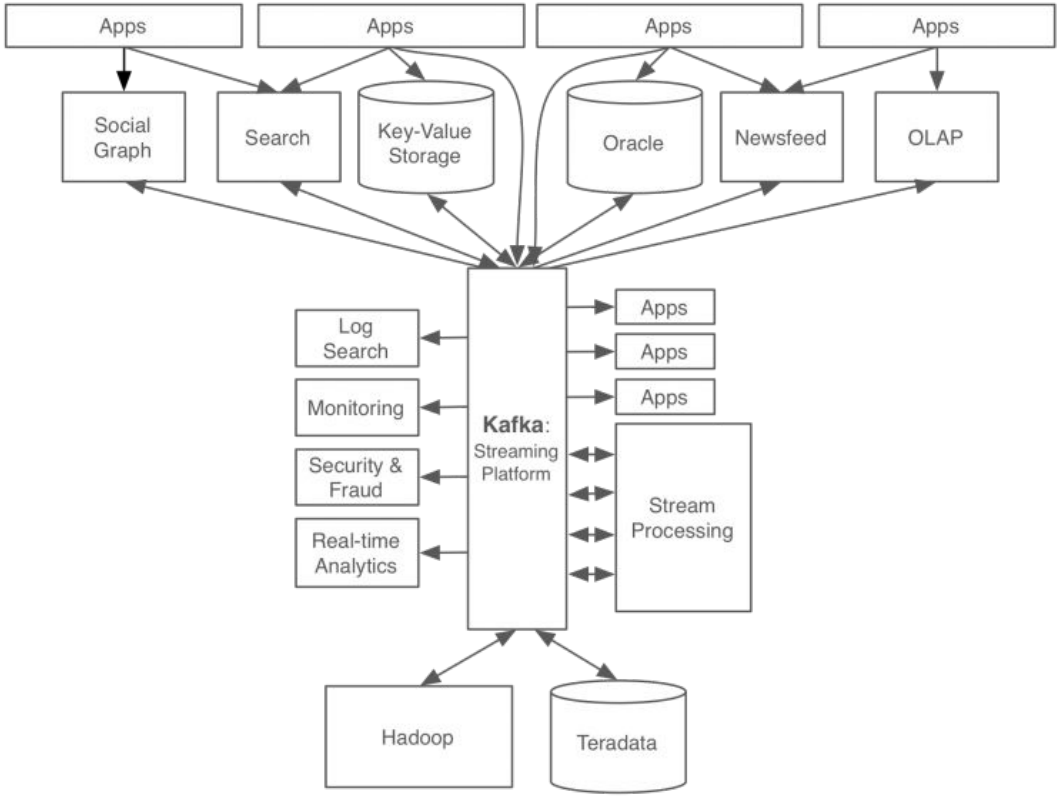
Streaming Platform



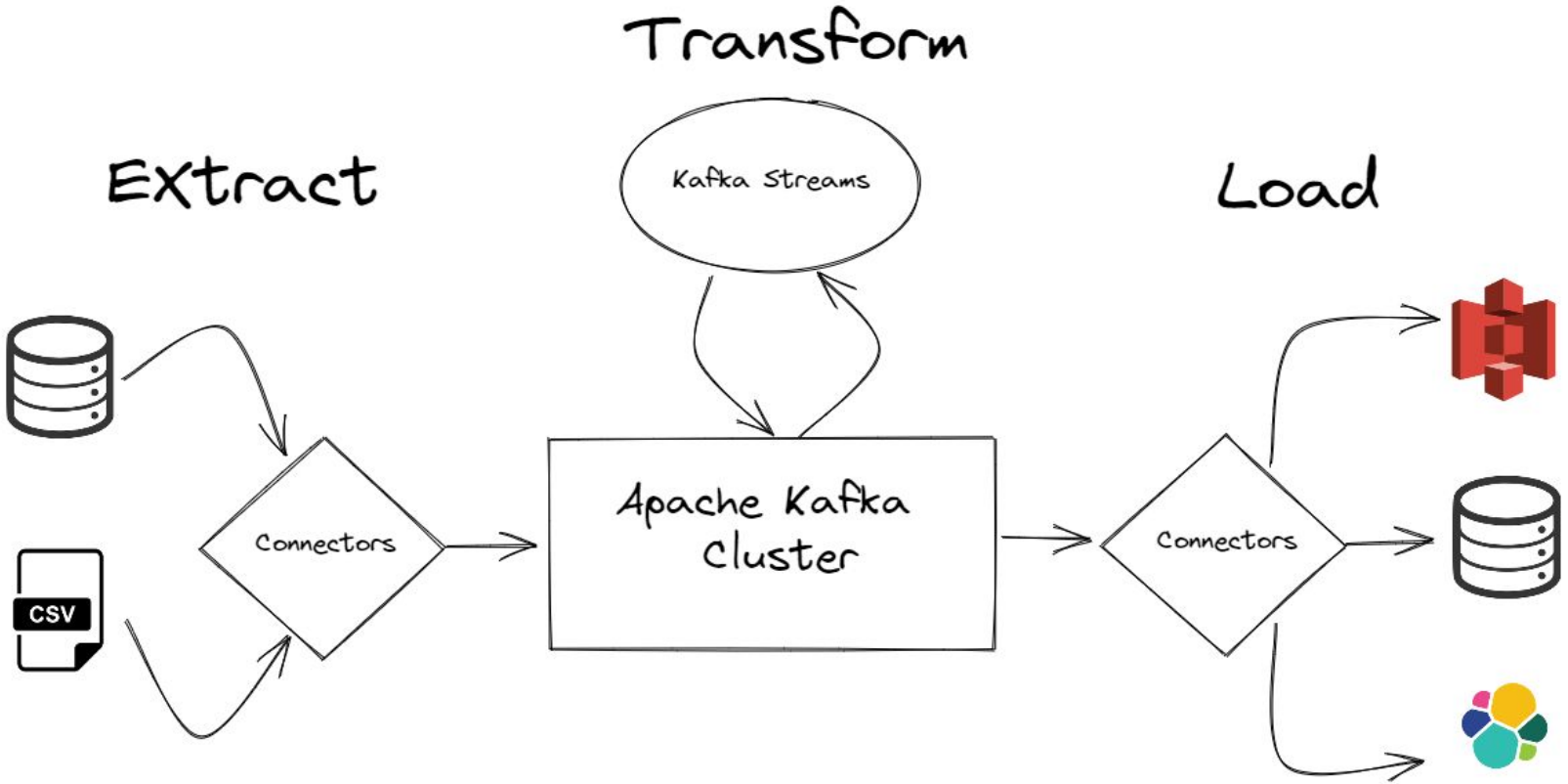
Streaming Platform



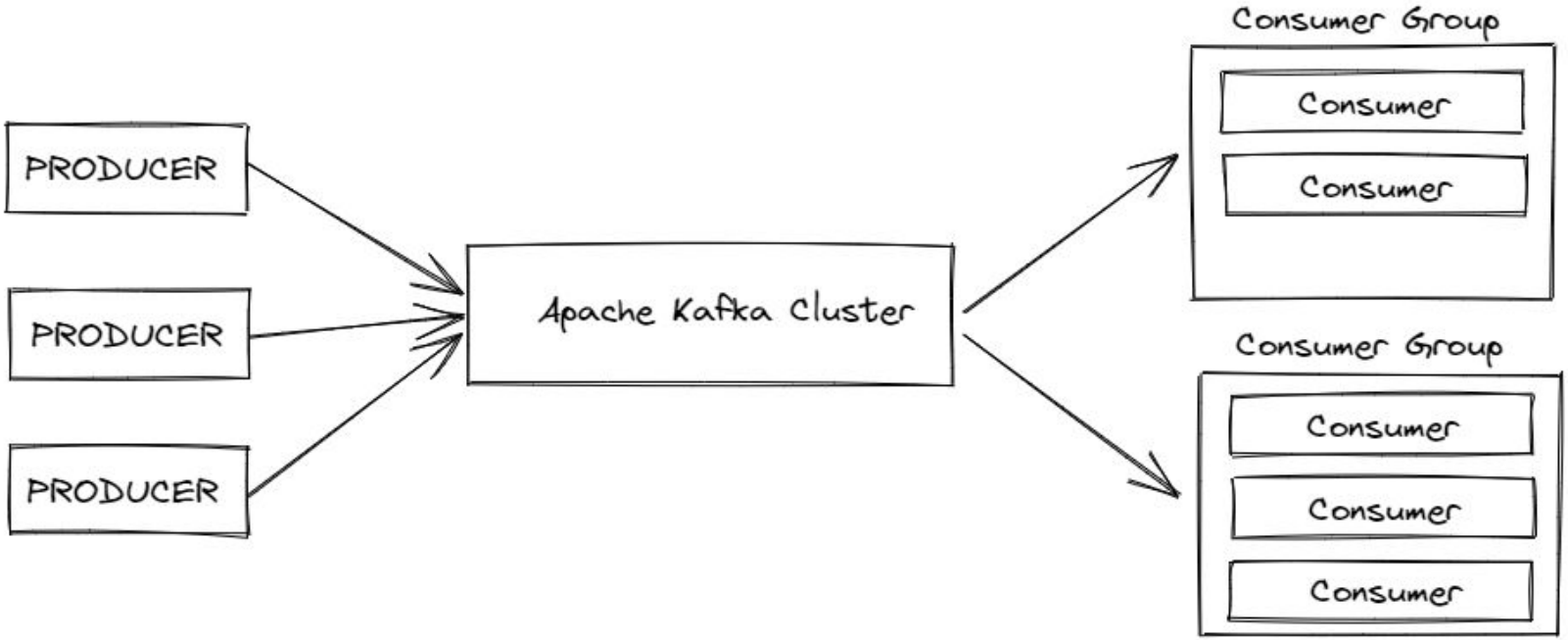
Streaming Platform



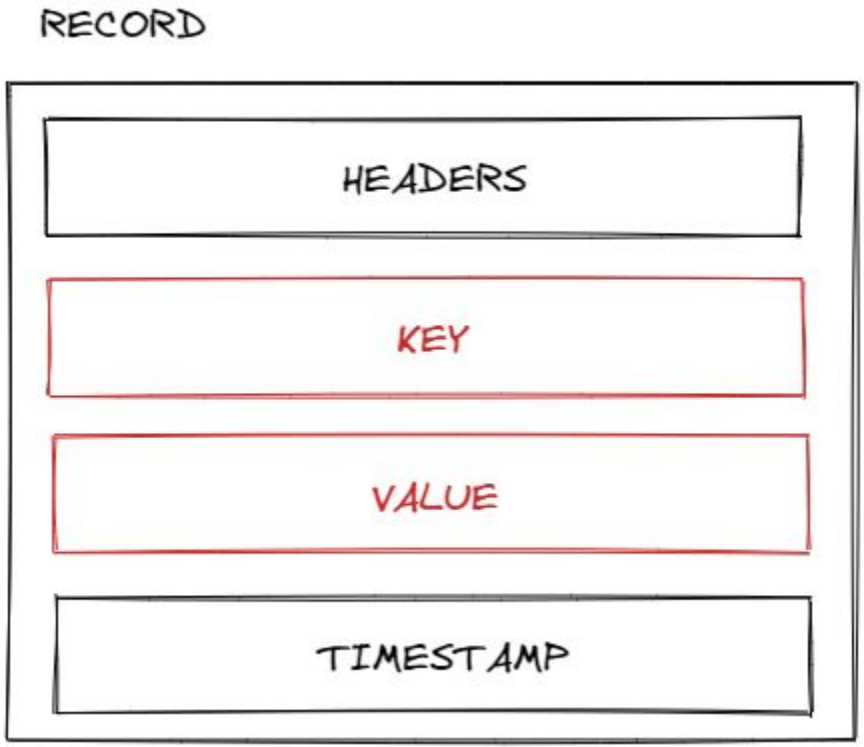
Streaming Platform



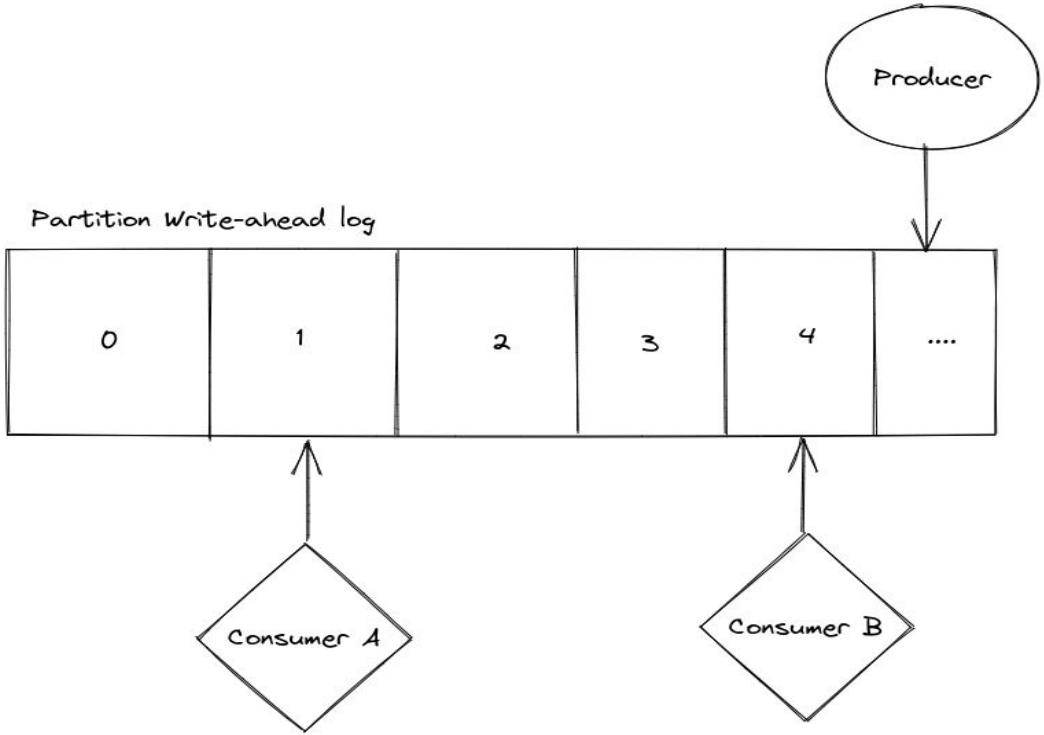
Clients - Producers & Consumers



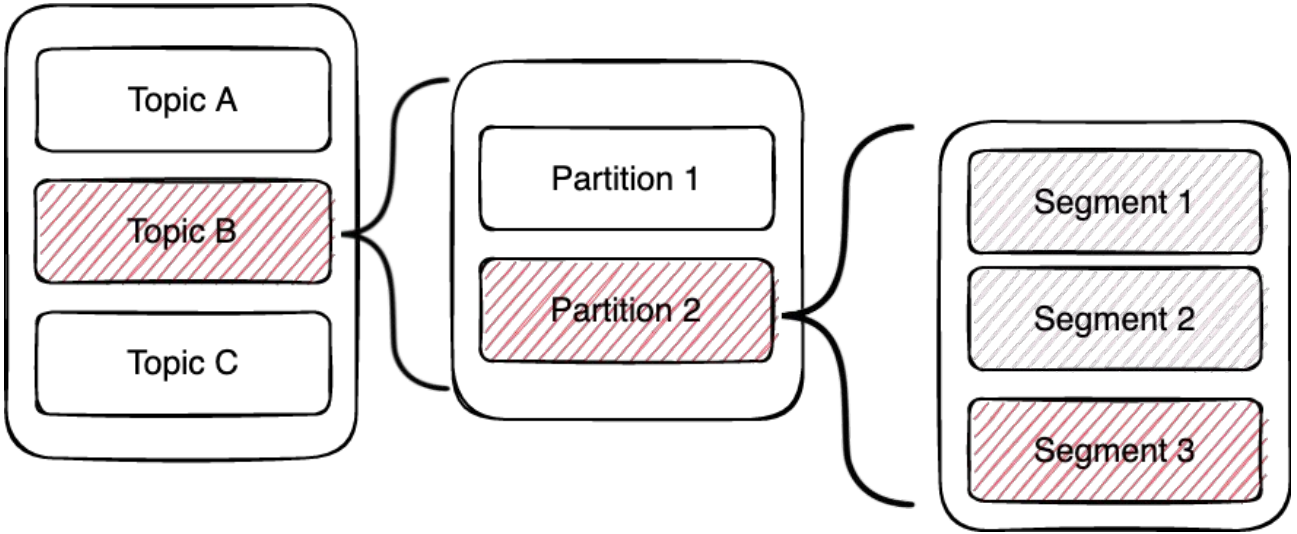
The Record



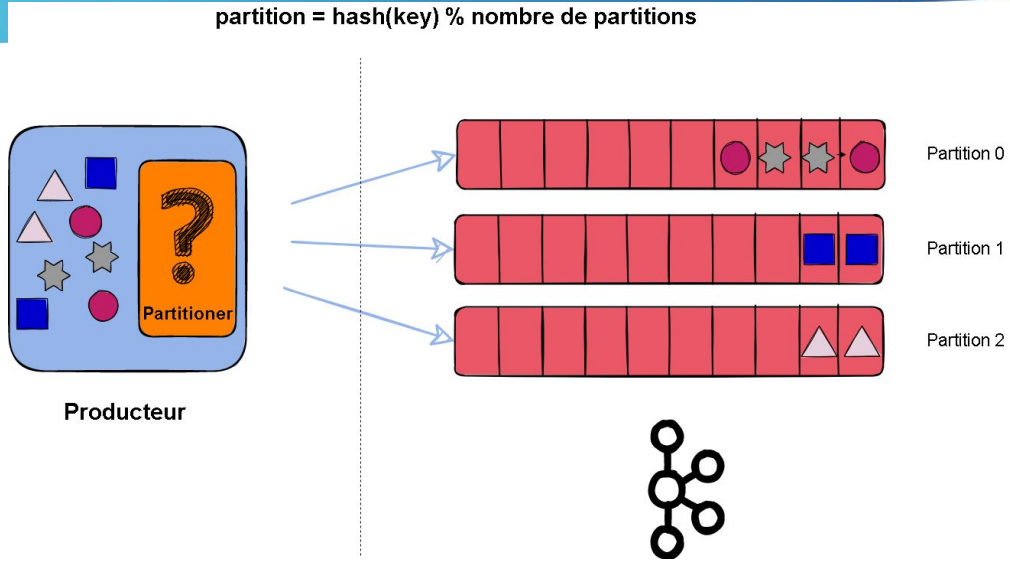
The Kafka Commit Log



Topic / Partition / Record



Partitioning

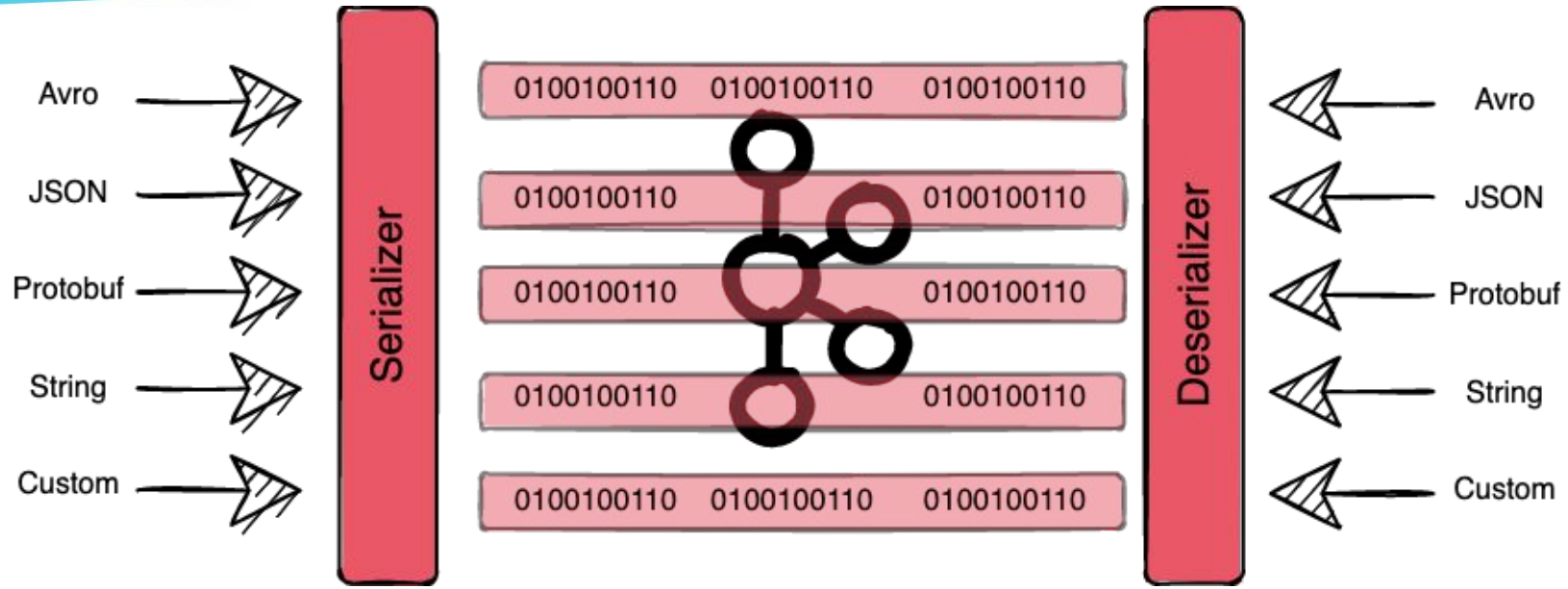


A message is delivered based on the partitioning key

The target partition is determined by the hash (key) modulo the number of partitions

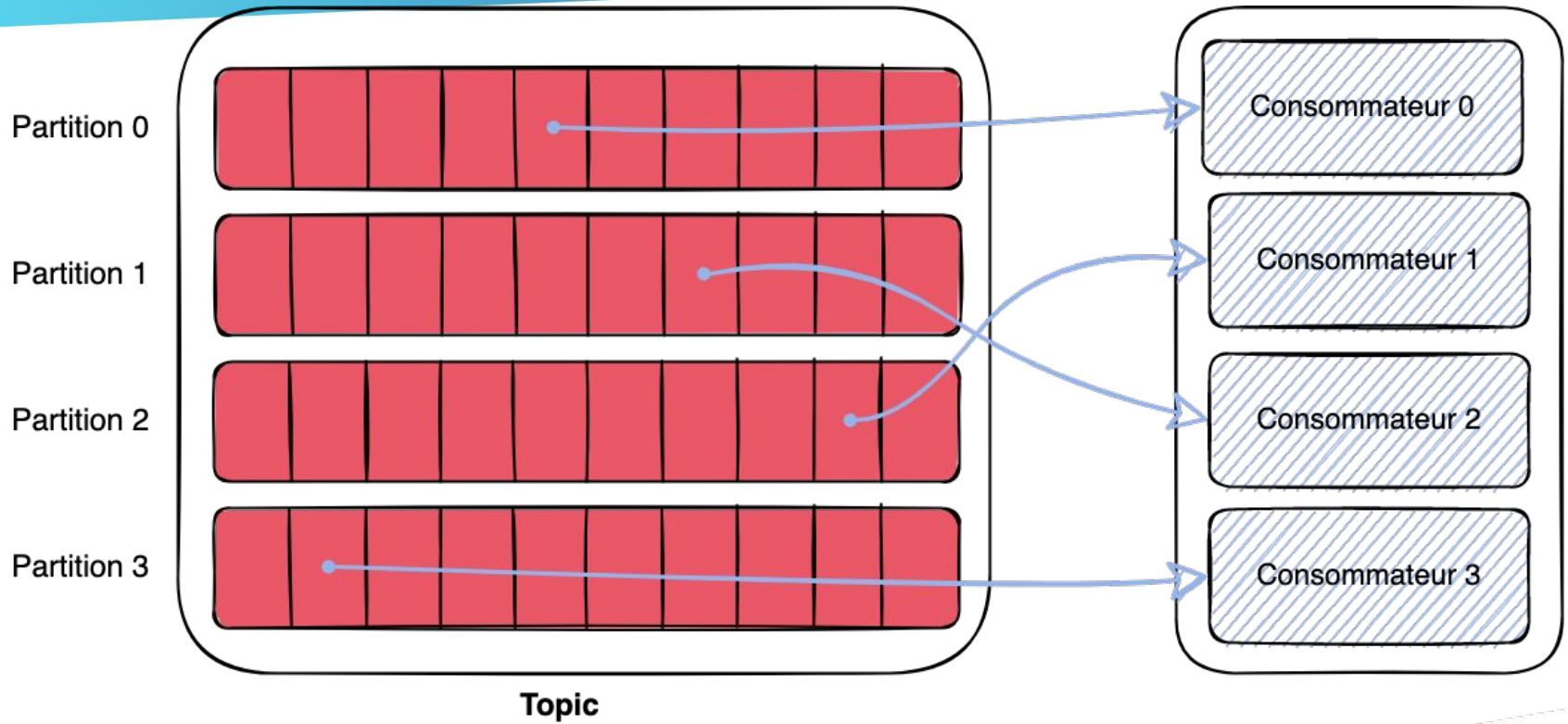
Default - Round Robin if no key is associated with a message

Serialization

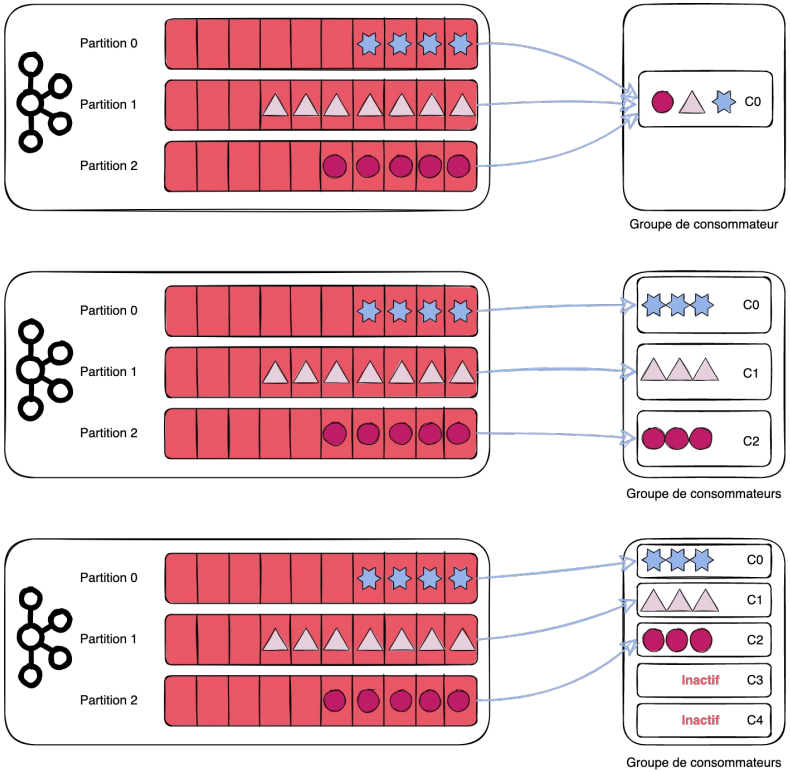


Kafka only knows binary!

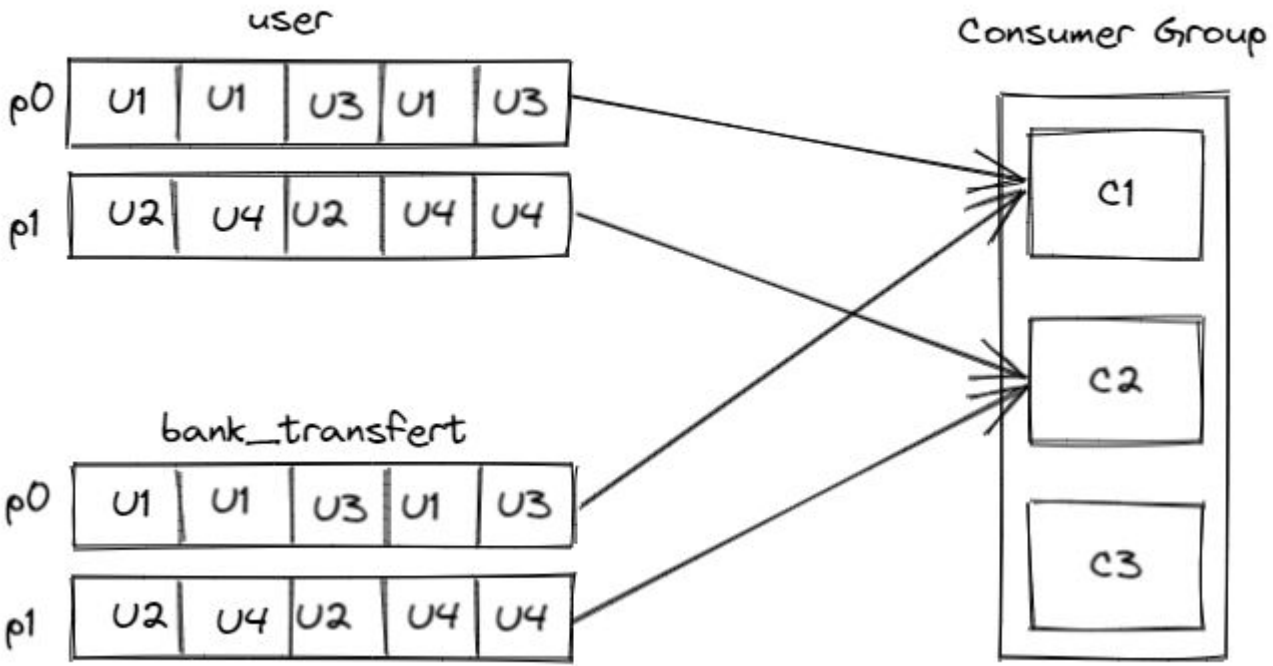
Consumer Group



Consumer Group



Partition Assignment Strategy



Partition Assignment Strategy

- Use **RangeAssignor** when joining data from multiple topics (this is the default):
  partition.assignment.strategy=org.apache.kafka.clients.consumer.RangeAssignor

- Use **RoundRobin** when performing stateless operations on records from many topics:
  partition.assignment.strategy=org.apache.kafka.clients.consumer.RoundRobinAssignor

- Sticky is **RoundRobin** with a best effort to maintain assignments across rebalances:
  partition.assignment.strategy=org.apache.kafka.clients.consumer.StickyAssignor

- **CooperativeSticky** is Sticky but it uses consecutive rebalances rather than the single stop-the-world used by Sticky (the next default):
  partition.assignment.strategy=org.apache.kafka.clients.consumer.CooperativeStickyAssignor

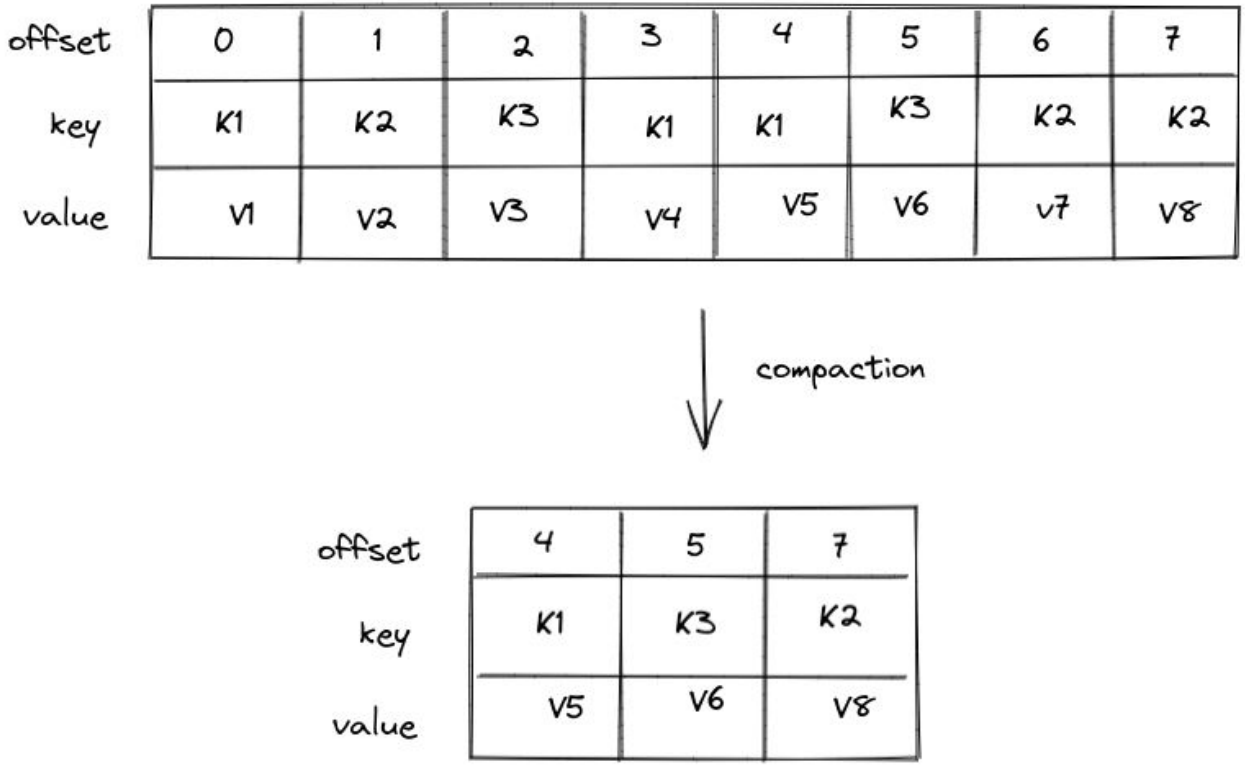
Topic Retention Policy

We don’t want topics to grow forever!

cleanup.policy

- *delete*
- *compact*
- both: *delete,compact*

Compacted Topic



Questions ?

Kafka Streams

Kafka Streams

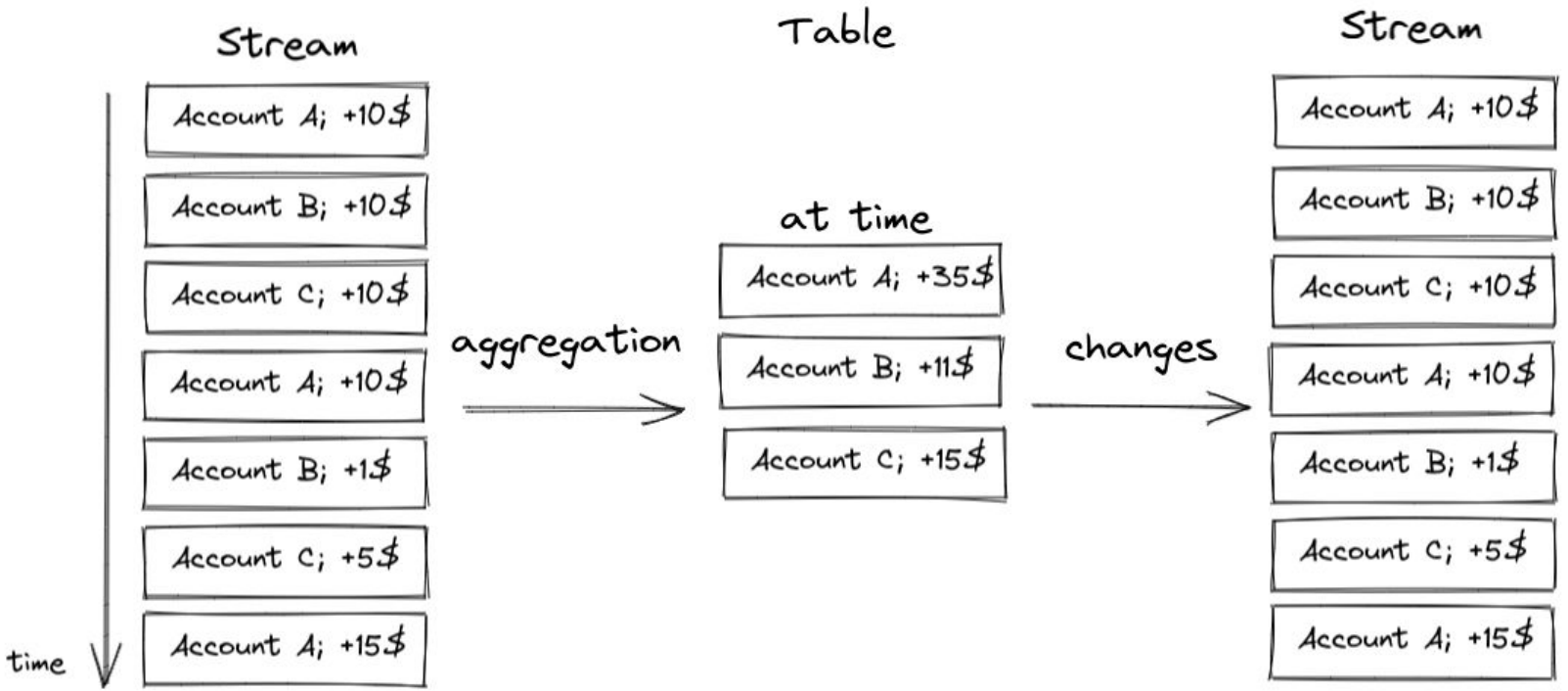
- Transforms and enriches data
  - per-record stream processing
  - millisecond latency
  - stateless & stateful processing
  - windowing operations

- Fault-tolerant and distributed processing
- Domain-Specific Language (DSL)
- High level operations: map, filter, count, etc.

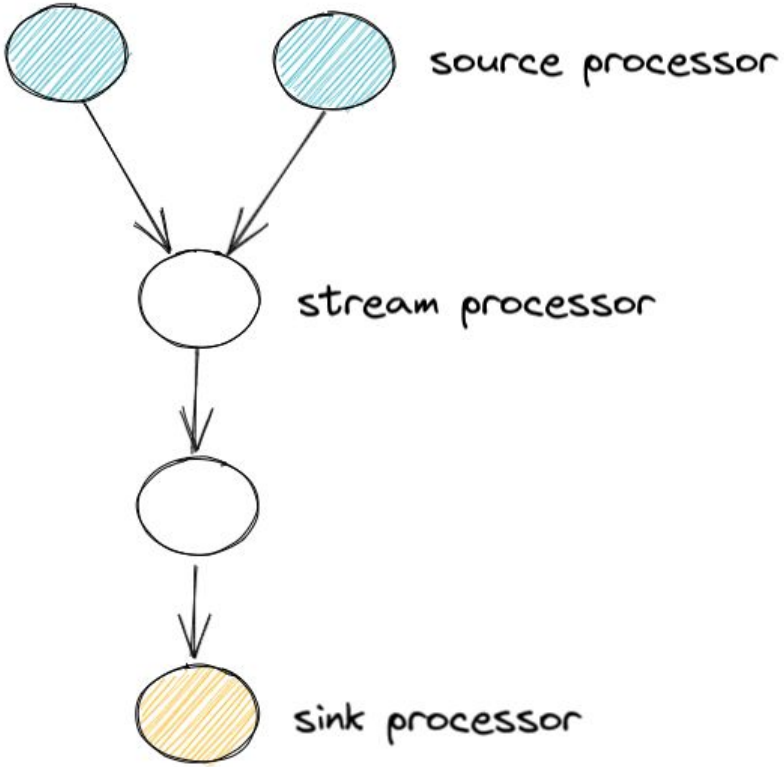
Kafka Streams

- not a framework, just a Java client library for building applications and microservices
- No separate resource management technology required
- Deploy to containers, VMs, bare metal, cloud
- Powered by Kafka: elastic, scalable, distributed, battle-tested
- Perfect for small, medium, large use cases
- Exactly-once processing semantics
- Part of the Apache Kafka project

Stream - Table Duality



Stream Topology



Stateless Processor

filter	KStream<K,V> smallPurchases = purchases.filter((key,value) -> value.amount < 50.0)
mapValues	KStream<K,V> upper = words.mapValues(value -> value.toUpperCase());
flatMapValues	KStream<byte[], String> words = textLines.flatMapValues(sentence -> Arrays.asList(pattern.split(sentence)));

Stateful Processor

count	Counts the number of instances of each key in the stream; KTable table = stream .groupByKey() .count()
reduce	Combines values of the stream using a supplied Reducer KTable table = stream .groupByKey() .reduce(...)

Windows, Aggregations, and Joins

Windows:

- Divide stream into "time buckets"
- Tumbling, Hopping, and Session windows

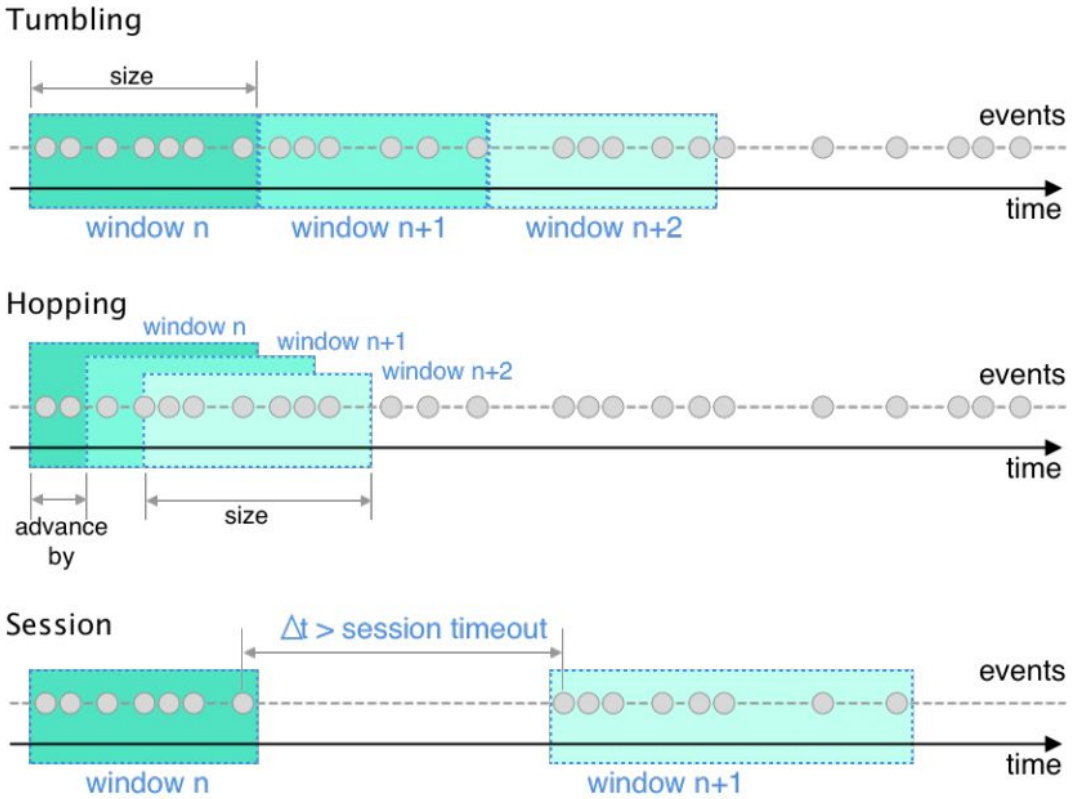
Aggregations

- Accumulate some value as new records come in
- Usually windowed
- Examples: sum, count, max, min

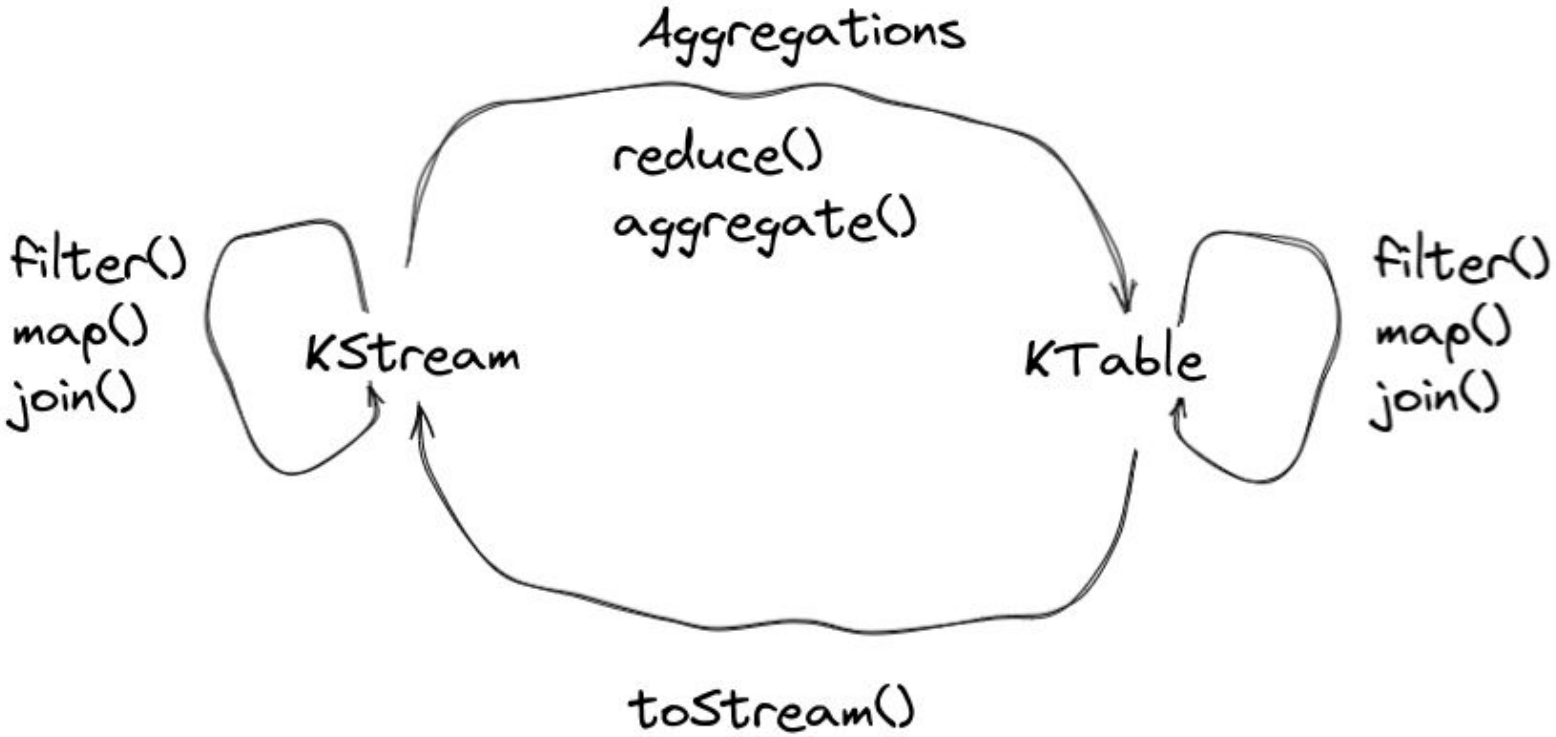
Joins:

- Combine different streams/tables together on a key
- Can be windowed with a "sliding window"

Windowing Types



Mixed Processing

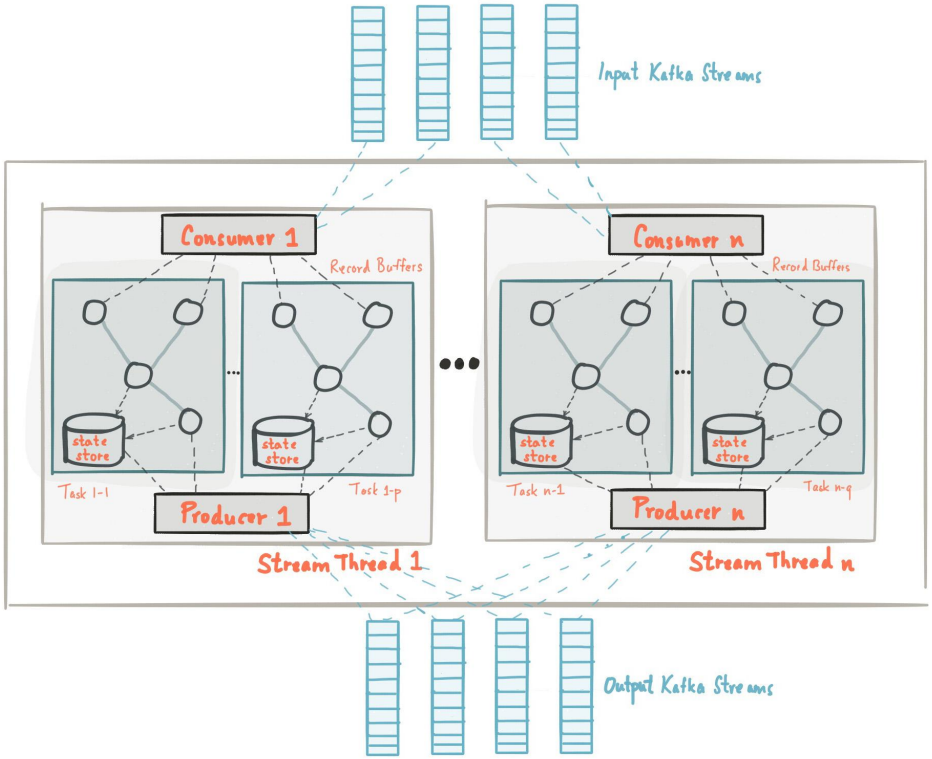


Configuring a Streams Application

```
import java.util.Properties;
import org.apache.kafka.streams.StreamsConfig;

Properties props = new Properties();
// Set a few key parameters
props.put(StreamsConfig.APPLICATION_ID_CONFIG, "my-first-streams-application");
props.put(StreamsConfig.BOOTSTRAP_SERVERS_CONFIG, "kafka-broker1:9092");
// Any further settings
props.put(... , ...);
```

Kafka Streams Architecture



Time To Code!

https://github.com/vgallet/workshop-kafka-streams

Thank you