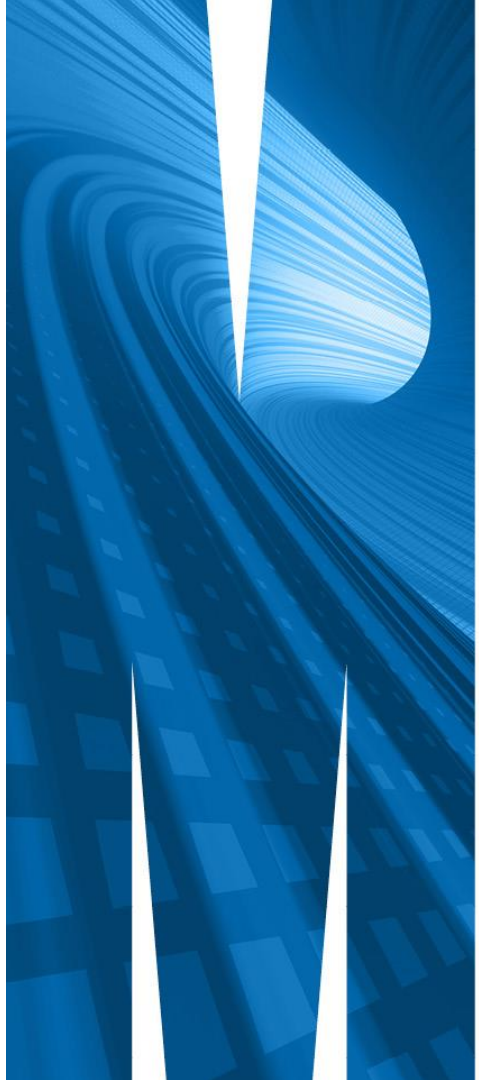
MONASH University

Week 9

FIT5202 Big Data Processing

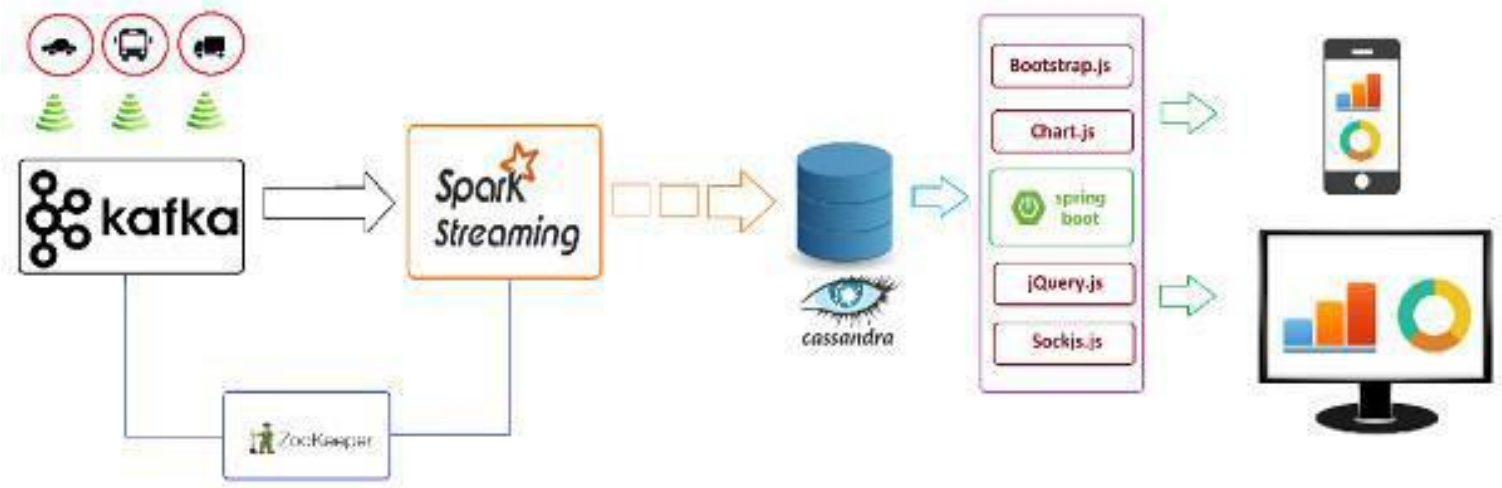
Data Streaming using Apache Kafka and Spark



Week 9 Agenda

- Week 8 Review
  - Implicit vs Explicit Data
  - Matrix Factorization
  - Collaborative Filtering with ALS
- Streaming using Apache Kafka
  - Kafka Producer
  - Kafka Consumer
    - Visualizing in real-time
  - **Use case : Click stream visualization**
- Spark Streaming Basics
  - Demo : word count
  - **Lab Task : Click Stream Analysis and Visualization**

Kafka Use Case (Traffic Data Monitoring)



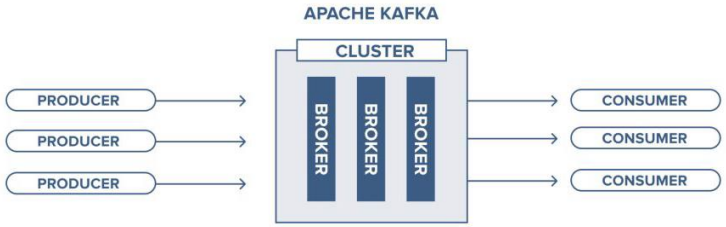
https://www.infoq.com/articles/traffic-data-monitoring-iot-kafka-and-spark-streaming/

What is Apache Kafka?

- Publish-subscribe messaging system
- Enables distributed applications

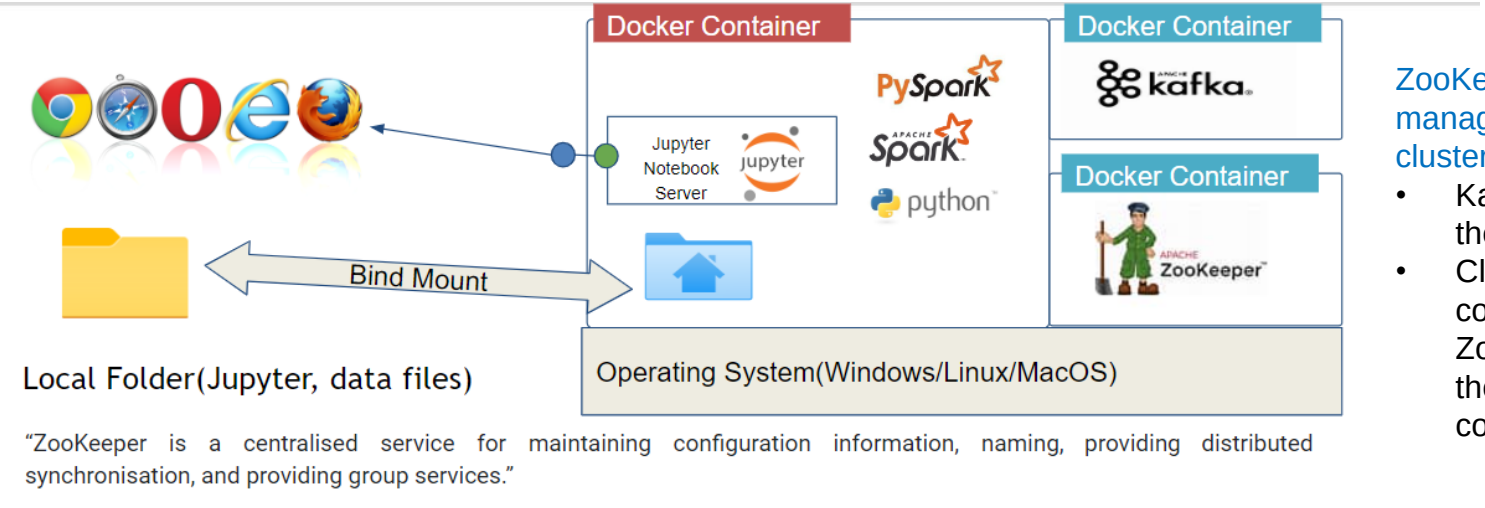
- Brokers utilize Apache ZooKeeper for management and coordination of the cluster
  - A Kafka broker receives messages from producers and stores them on disk keyed by unique offset.
  - A Kafka broker allows consumers to fetch messages by topic, partition and offset

- Each broker instance is capable of handling read and write quantities reaching to the hundreds of thousands each second (and terabytes of messages) without any impact on performance.



https://www.cloudkarafka.com/blog/2016-11-30-part1-kafka-for-beginners-what-is-apache-kafka.html
https://www.instaclustr.com/apache-kafka-architecture/

Environment



ZooKeeper helps Kafka in managing and coordinating its cluster

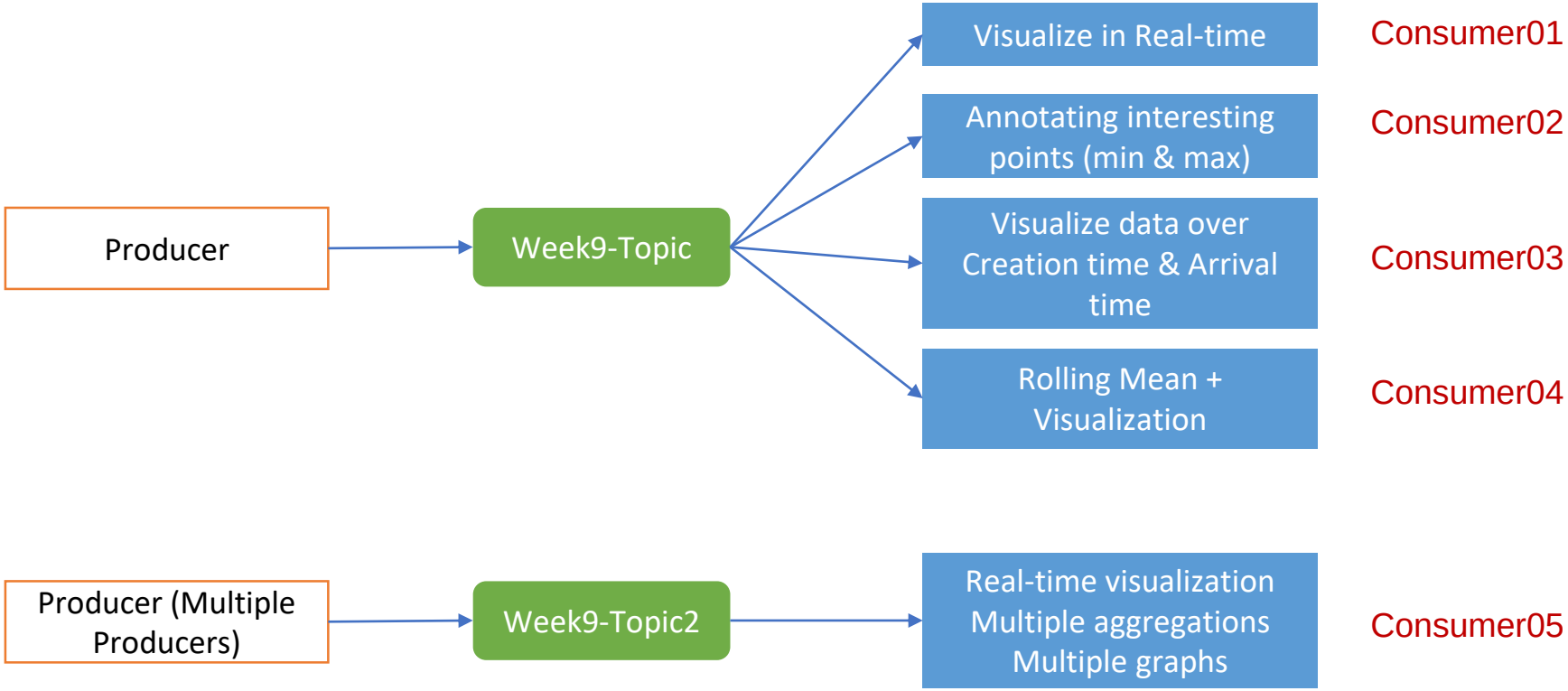
- Kafka servers register themselves in ZooKeeper
- Clients (producers and consumers) use ZooKeeper to discover the brokers and maintain connections to them

Step 1: **docker run -d --network fit5202 --name zookeeper -p 2181:2181 monashfit/fit5202-zookeeper**

Step 2:
docker run -d --network fit5202 --name kafka -e KAFKA_ZOOKEEPER_CONNECT=zookeeper:2181 -e KAFKA_ADVERTISED_HOST_NAME=kafka -p 9092:9092 monashfit/fit5202-kafka

All timezones in three containers are UTC. It can be configured with TZ environment variable(e.g. -e TZ=Australia/Melbourne).

DEMO Kafka Implementation Scenarios for Lab



Kafka Producer and Consumer Properties

- **KafkaProducer**
  - Bootstrap_servers (connect to brokers)
  - Value_serializer (convert data to byte arrays & encode with ascii)
  - Api_version

- **KafkaConsumer**
  - Consumer_timeout_ms
  - Auto_offset_reset
  - Bootstrap_servers
  - Value_deserializer
  - Api_version

```
_producer = KafkaProducer(bootstrap_servers=[f'{hostip}:9092'],
                          value_serializer=lambda x: dumps(x).encode('ascii'),
                          api_version=(0, 10))
```

```
_consumer = KafkaConsumer(topic,
                          consumer_timeout_ms=10000, # stop iteration if no messa
                          auto_offset_reset='latest', # comment this if you don't
                          bootstrap_servers=[f'{hostip}:9092'],
                          value_deserializer=lambda x: loads(x.decode('ascii')),
                          api_version=(0, 10))
```

Port 9092 is commonly associated with Apache Kafka

Serialization
- converting an object into a stream of bytes for the purpose of transmission

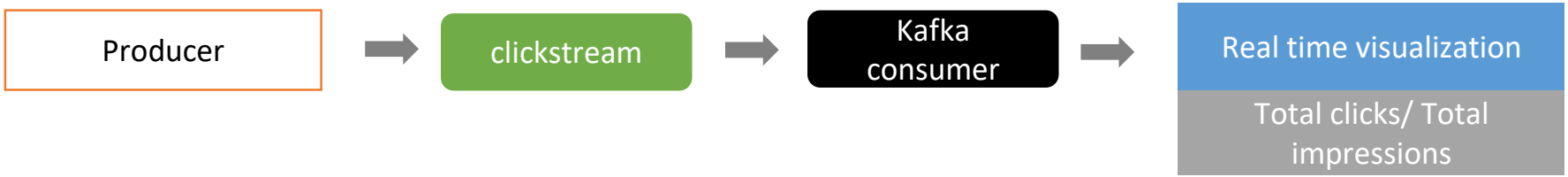
dumps(x) – convert python object into Jason object

https://kafka-python.readthedocs.io/en/master/apidoc/KafkaConsumer.html
https://kafka-python.readthedocs.io/en/master/apidoc/KafkaProducer.html

Lab Task for Kafka

Clickstream.csv

Impressions - the number of times digital ad has been viewed.
Clicks - number of times users have clicked on a ad



Example output of Producer:

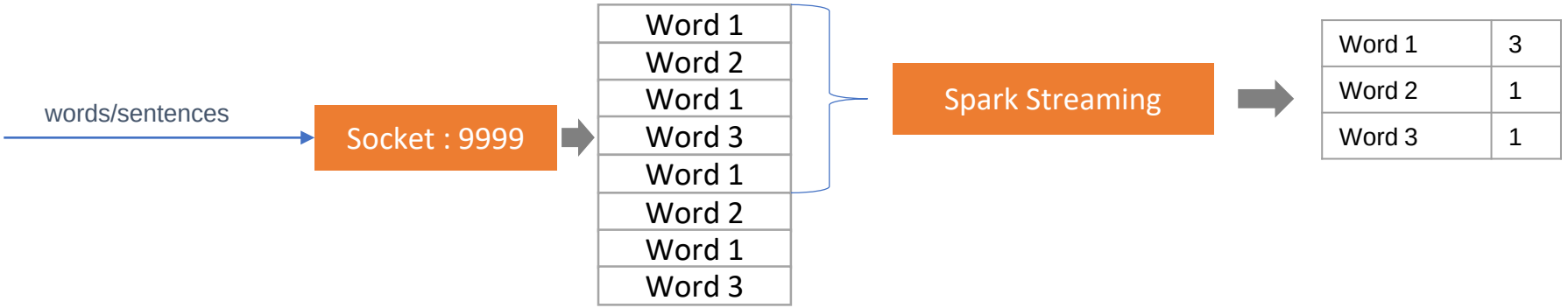

```
Message published successfully. Data: {'data': [{'Age': '36', 'Gender': '0', 'Impressions': '3', 'Clicks': '0', 'Signed_In':
'1'}, {'Age': '73', 'Gender': '1', 'Impressions': '3', 'Clicks': '0', 'Signed_In': '1'}, {'Age': '30', 'Gender': '0', 'Impres
sions': '3', 'Clicks': '0', 'Signed_In': '1'}, {'Age': '49', 'Gender': '1', 'Impressions': '3', 'Clicks': '0', 'Signed_In':
'1'}, {'Age': '47', 'Gender': '1', 'Impressions': '11', 'Clicks': '0', 'Signed_In': '1'}, {'Age': '47', 'Gender': '0', 'Impre
ssions': '11', 'Clicks': '1', 'Signed_In': '1'}, {'Age': '0', 'Gender': '0', 'Impressions': '7', 'Clicks': '1', 'Signed_In':
'0'}], 'ts': 1745816344}
Message published successfully. Data: {'data': [{'Age': '46', 'Gender': '0', 'Impressions': '5', 'Clicks': '0', 'Signed_In':
'1'}, {'Age': '16', 'Gender': '0', 'Impressions': '3', 'Clicks': '0', 'Signed_In': '1'}, {'Age': '52', 'Gender': '0', 'Impres
sions': '4', 'Clicks': '0', 'Signed_In': '1'}, {'Age': '0', 'Gender': '0', 'Impressions': '8', 'Clicks': '1', 'Signed_In':
'0'}, {'Age': '21', 'Gender': '0', 'Impressions': '3', 'Clicks': '0', 'Signed_In': '1'}, {'Age': '0', 'Gender': '0', 'Impress
ions': '4', 'Clicks': '0', 'Signed_In': '0'}, {'Age': '57', 'Gender': '0', 'Impressions': '6', 'Clicks': '0', 'Signed_In':
'1'}], 'ts': 1745816349}
```

Spark Structured Streaming

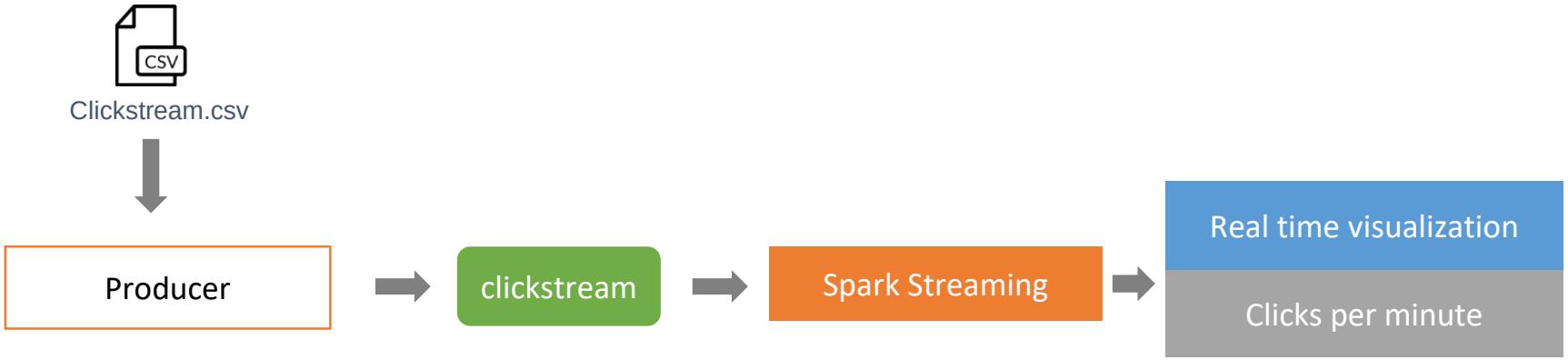
To be covered in Week 10 Lecture

DEMO Spark Structured Streaming

Word Count Demo



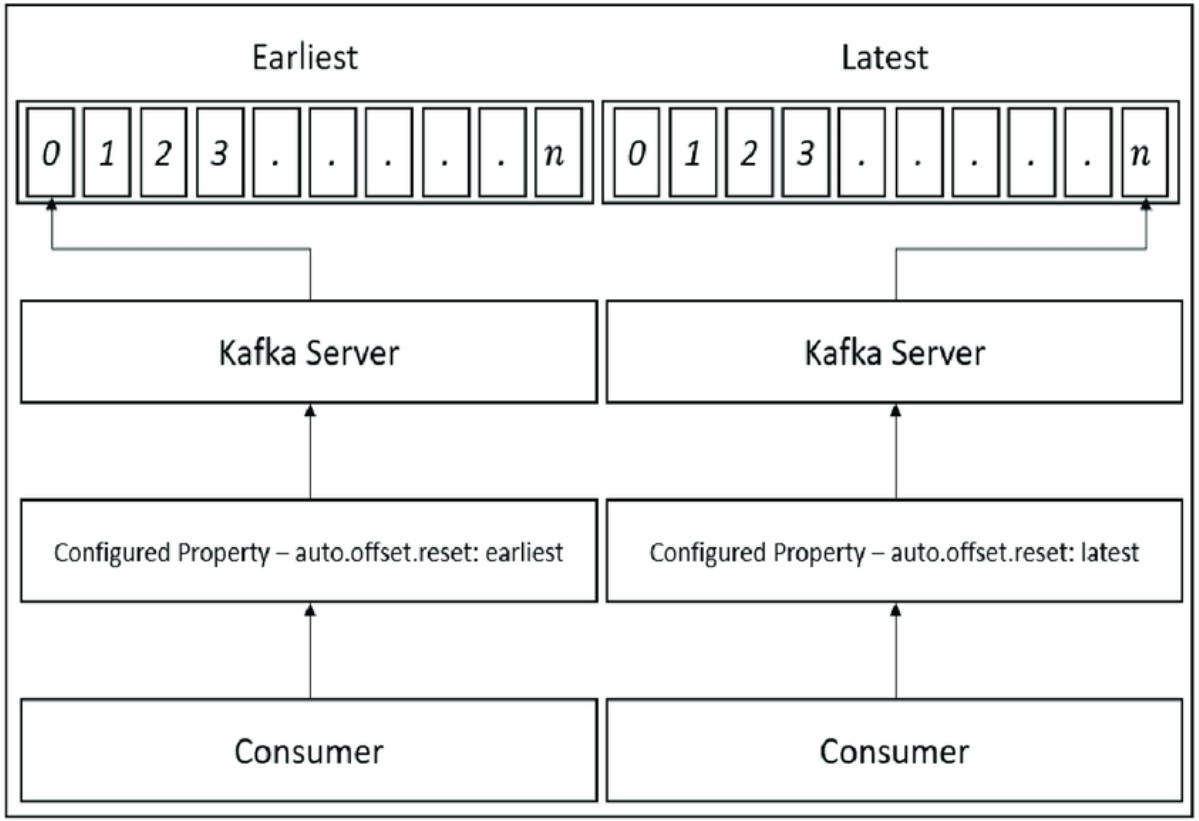
Next Week for Spark Structured Streaming



See Demo files (Week 10):
Clickstream-Producer DEMO
Spark Streaming - ClickStream-Analysis DEMO [V 1.1]

LT2-Producer
Clickstream Spark Streaming - Handling Json Array DEMO

Thank You!



https://www.youtube.com/watch?v=r5mZ74N997o