#08 practical class

Apache Kafka

COMPUTING SYSTEMS AND INFRASTRUCTURES

(SISTEMAS E INFRAESTRUTURAS DE COMPUTAÇÃO)

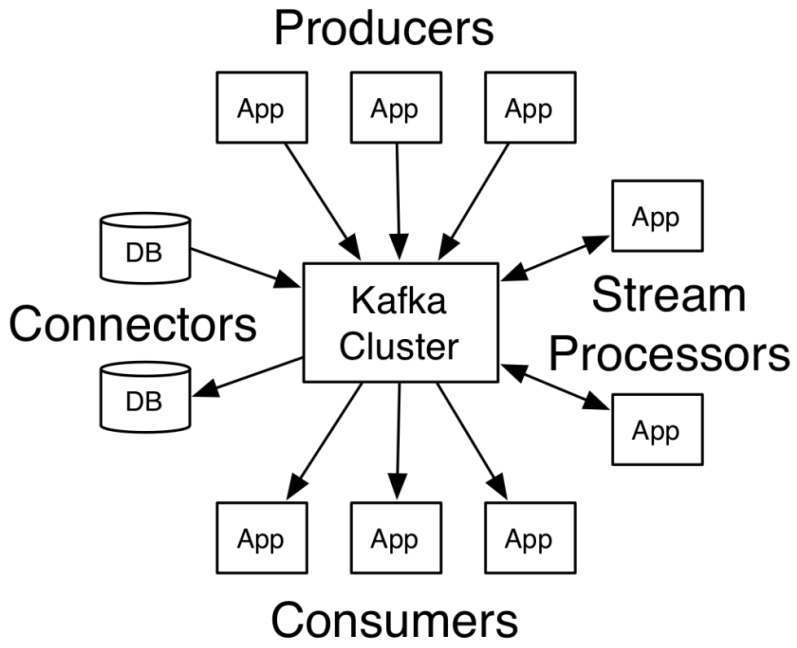
Overview

- Apache Kafka

Kafka

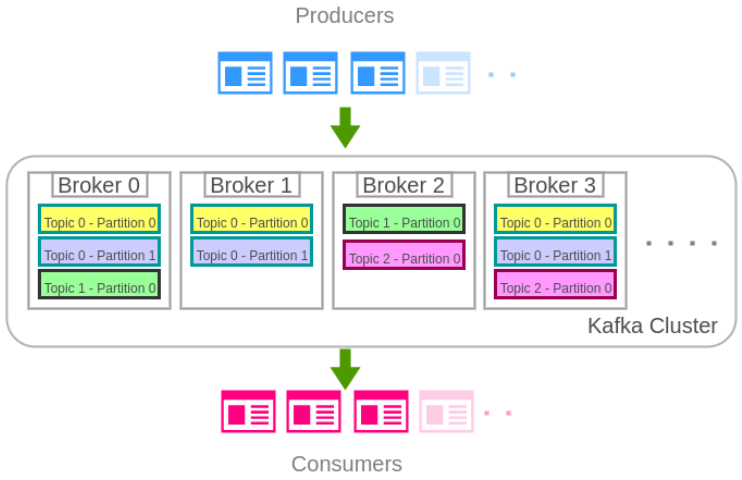
https://kafka.apache.org

- Kafka is a distributed system consisting of servers and clients that communicate via a high-performance TCP network protocol
- One or more servers that can span multiple datacenters or cloud regions

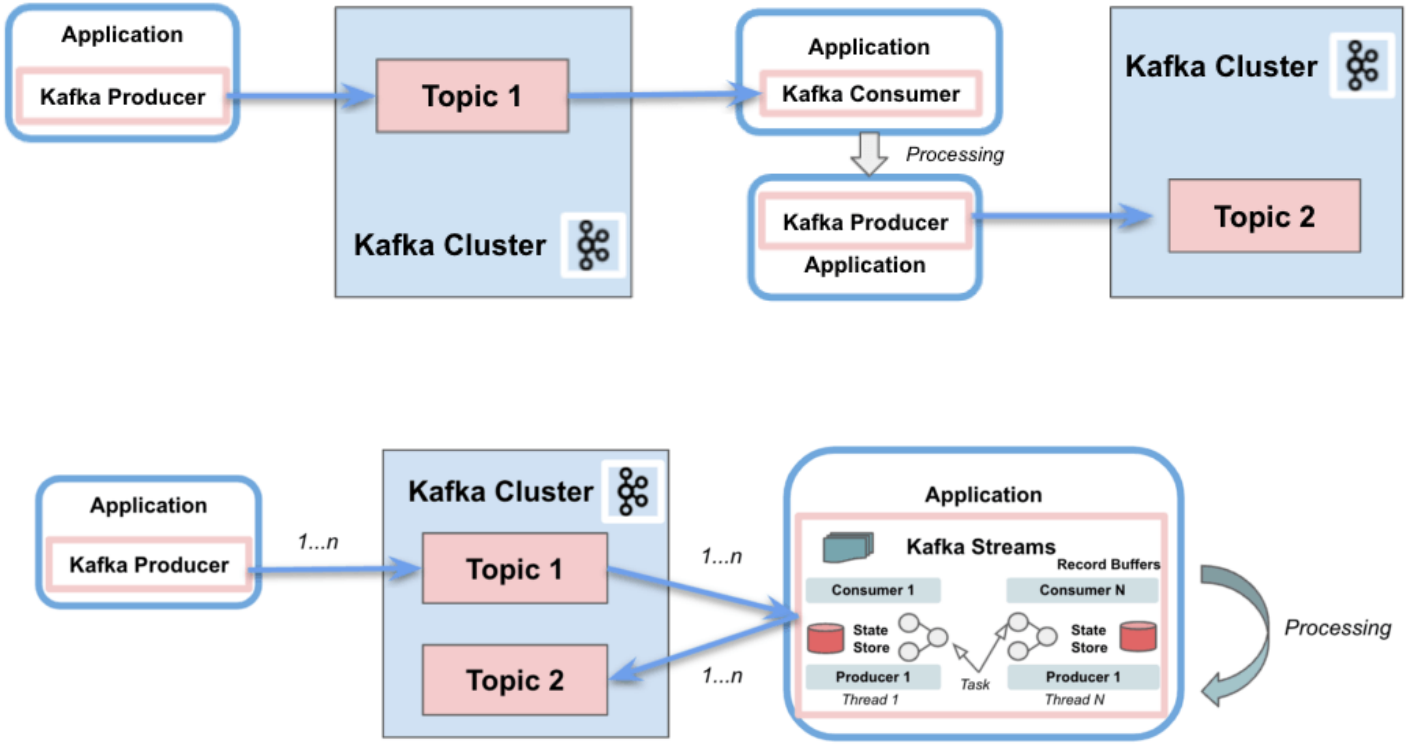


Main characteristics

- Fault Tolerance and High Availability
- High Throughput and Low-latency
- Scalability and Reliability
- Real-time Data Streaming and Processing
- Ease of Integration through Multiple Connections



Consumer and Stream



From https://www.baeldung.com/java-kafka-streams-vs-kafka-consumer

Deploy Kafka

- Create network for internal Kafka communication
- Deploy Zookeeper
- Deploy Kafka server (enabling external access)

```
docker network create kafka-network

docker run -d --name kafka-zookeeper \
    --network kafka-network \
    -e ALLOW_ANONYMOUS_LOGIN=yes \
    bitnami/zookeeper:latest

docker run -d --name kafka \
    --network kafka-network \
    -e ALLOW_PLAINTEXT_LISTENER=yes \
    -e KAFKA_CFG_ZOOKEEPER_CONNECT=kafka-zookeeper:2181 \
    -e KAFKA_CFG_LISTENER_SECURITY_PROTOCOL_MAP=CLIENT:PLAINTEXT,EXTERNAL:PLAINTEXT \
    -e KAFKA_CFG_LISTENERS=CLIENT://:9092,EXTERNAL://:9093 \
    -e KAFKA_CFG_ADVERTISED_LISTENERS=CLIENT://kafka:9092,EXTERNAL://192.168.23.130:9093 \
    -e KAFKA_CFG_INTER_BROKER_LISTENER_NAME=CLIENT \
    -p 9092:9092 \
    -p 9093:9093 \
    bitnami/kafka:latest
```

Example of python Kafka producer

```
#!/usr/bin/env python
import time

from kafka import KafkaAdminClient, KafkaConsumer, KafkaProducer
from kafka.admin import NewTopic

if __name__ == "__main__":
    # Create 'my-topic' Kafka topic
    print("Starting producer script...")
    try:
        admin = KafkaAdminClient(bootstrap_servers='192.168.23.130:9093')
        print("KafkaAdminClient passed!")

        topic = NewTopic(name='sic-topic',
                         num_partitions=1,
                         replication_factor=1)
        admin.create_topics([topic])
        print("New topic created!")
    except Exception:
        pass

    producer = KafkaProducer(bootstrap_servers='192.168.23.130:9093')
    print("KafkaProducer passed!")

    for n in range(1,100):
        print("N: "+str(n))
        time.sleep(1)
        producer.send('sic-topic', b'number %d' % n)
        print("Message sent!")
        producer.flush()

    print("Ending script!")
```

Example of python Kafka consumer

```
#!/usr/bin/env python
from kafka import KafkaConsumer

if __name__ == '__main__':
    print("Starting consumer script...")

    consumer = KafkaConsumer(bootstrap_servers='192.168.23.130:9093')
    print("KafkaConsumer passed!")
    consumer.subscribe(['sic-topic'])
    print("Subscribed to the topic!")

    print("Waiting for message...")
    for msg in consumer:
        print("Msg: " + msg.value.decode('utf-8'))
        print("Waiting for message...")

    print("Ending script!")
```

Activities

- Deploy Kafka, Zookeeper, the producer, and the consumer using Docker compose
- To test Kafka, you can use the *.sh inside the container → /opt/…
  - kafka-topics.sh --create --topic sic-topic --bootstrap-server localhost:9092 --replication-factor 1 --partitions 1
  - kafka-console-producer.sh --topic sic-topic --bootstrap-server localhost:9092
  - kafka-console-consumer.sh --topic sic-topic --from-beginning --bootstrap-server localhost:9092
- For the producer and the consumer
  - Use python:3.11