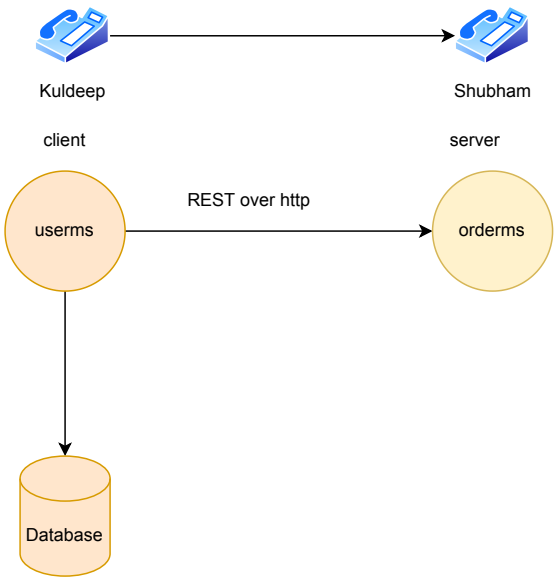
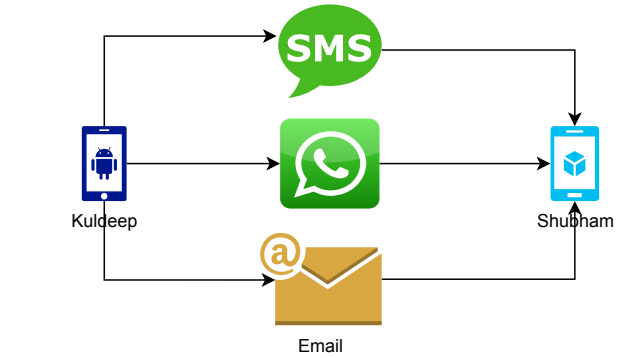
REST vs Messaging

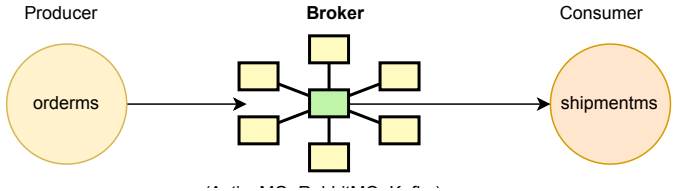
RESTful

Synchronous Communication



Messaging



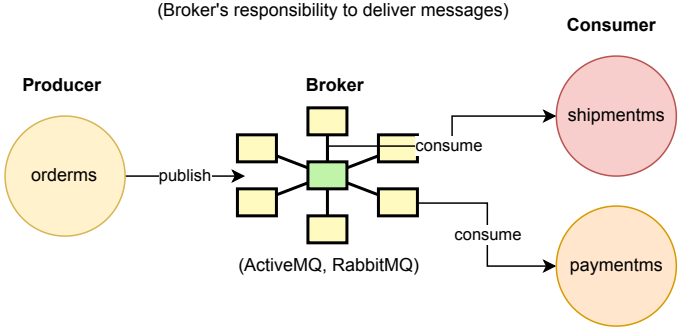


(ActiveMQ, RabbitMQ, Kafka)

Messaging (Contd.)

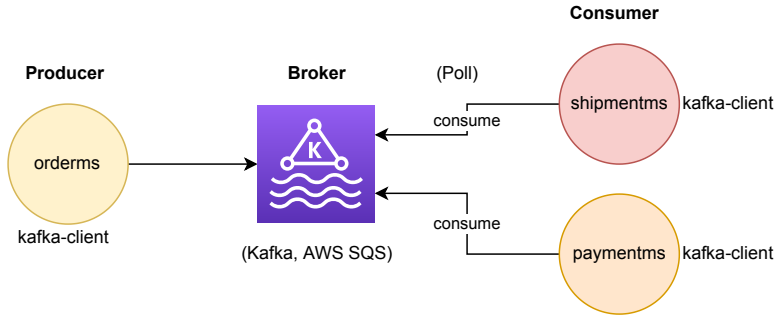
Push-Based

(Broker's responsibility to deliver messages)



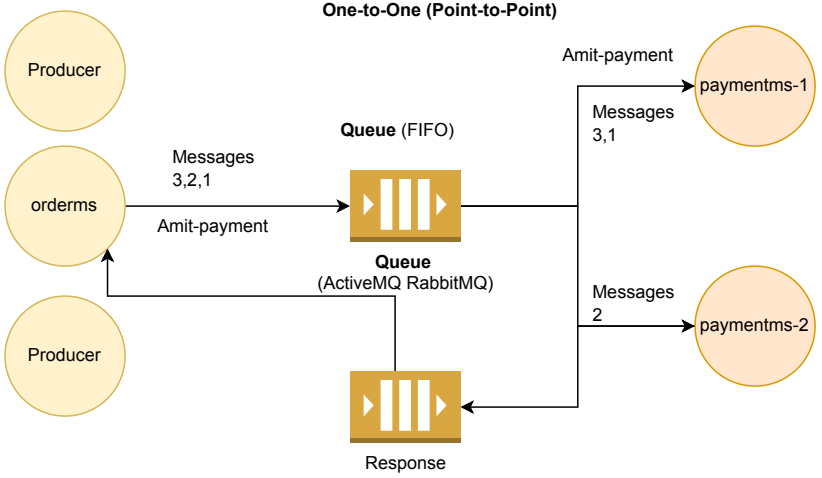
Pull-Based

(Consumer's responsibility to consume messages)

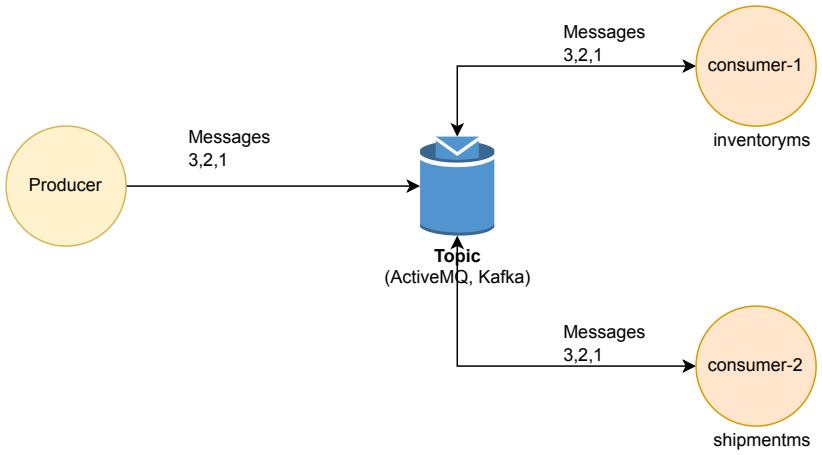


Messaging (Contd.)

One-to-One (Point-to-Point)



One-To-Many (Publisher/Subscriber or Pub/Sub)



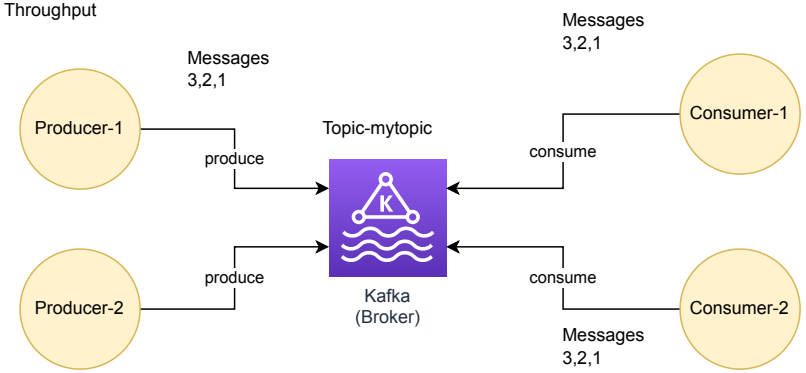
Kafka

1. Apache Kafka
2. Confluent Kafka

Initially developed at LinkedIn
Later came under Apache Open Source Licence

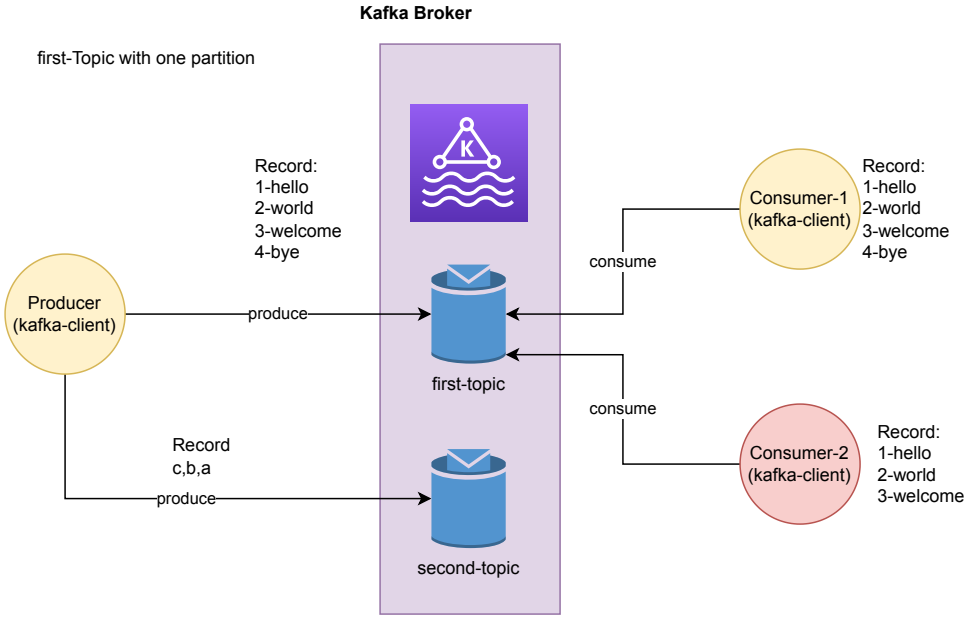
Characteristics of Kafka:

1. Highly Available
2. Scalable
3. Resilient
4. Fault Tolerant
5. Distributed
6. Low Latency
7. High Throughput

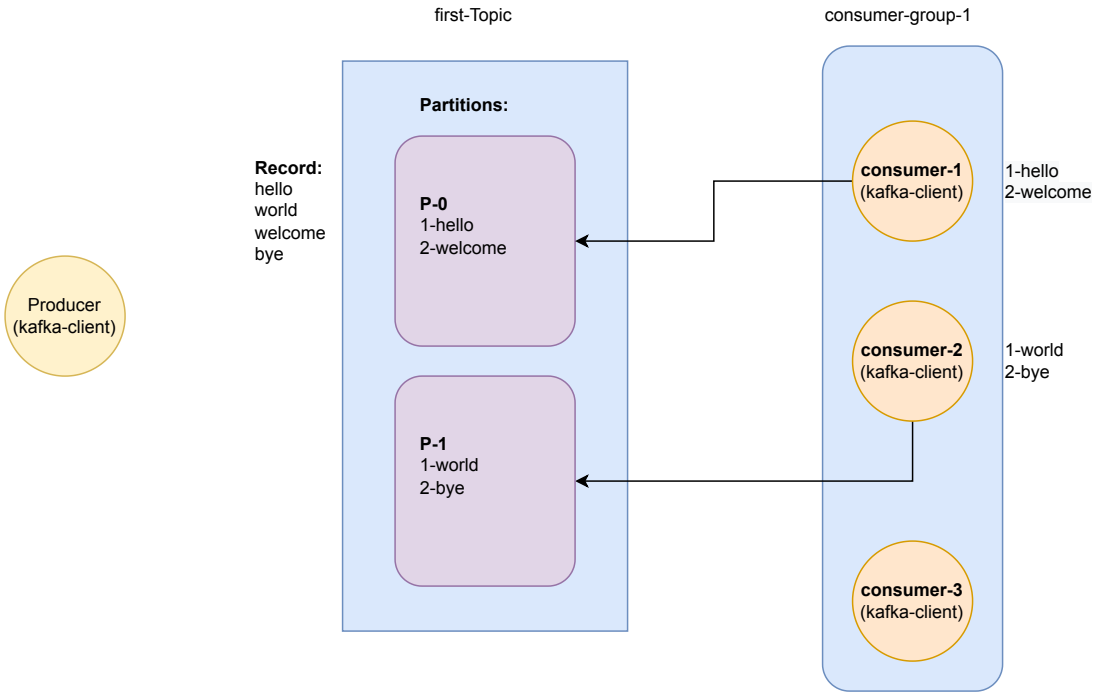


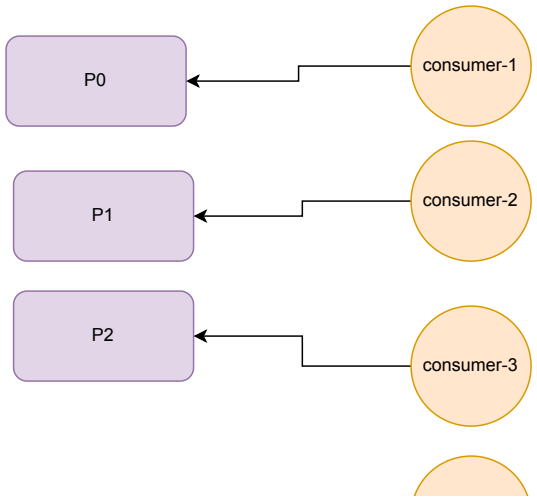
Kafka Terminologies

1. Broker
2. Topic
3. Record(message)
4. Producer
5. Consumer
6. Consumer Group
7. Kafka Client
8. Partition (Scalable)
9. Replication Factor (Highly Available)
10. Offset
11. Zookeeper

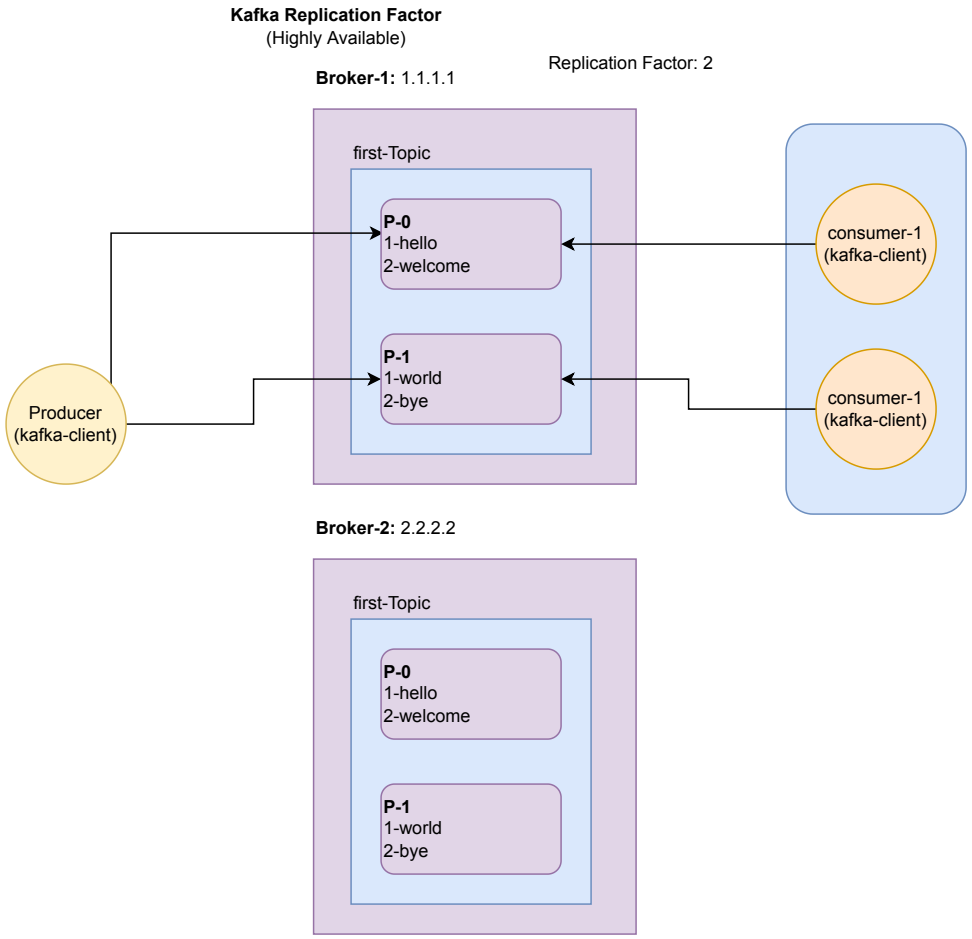


Kafka Partitions
(Scalable)

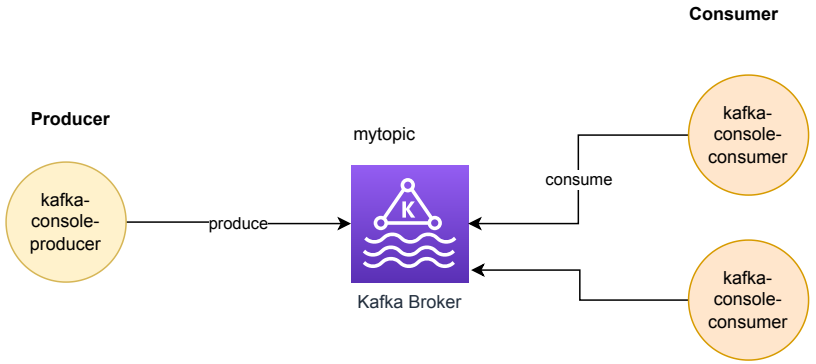





consumer-4


Kafka Replication Factor
(Highly Available)
Replication Factor: 2
Broker-1: 1.1.1.1
first-Topic
P-0
1-hello
2-welcome
P-1
1-world
2-bye
Producer
(kafka-client)
consumer-1
(kafka-client)
consumer-1
(kafka-client)
Broker-2: 2.2.2.2
first-Topic
P-0
1-hello
2-welcome
P-1
1-world
2-bye

Kafka Console Producer
Kafka Console Consumer



Use Case# 1
(Sequential Processing)

Positive Scenario:

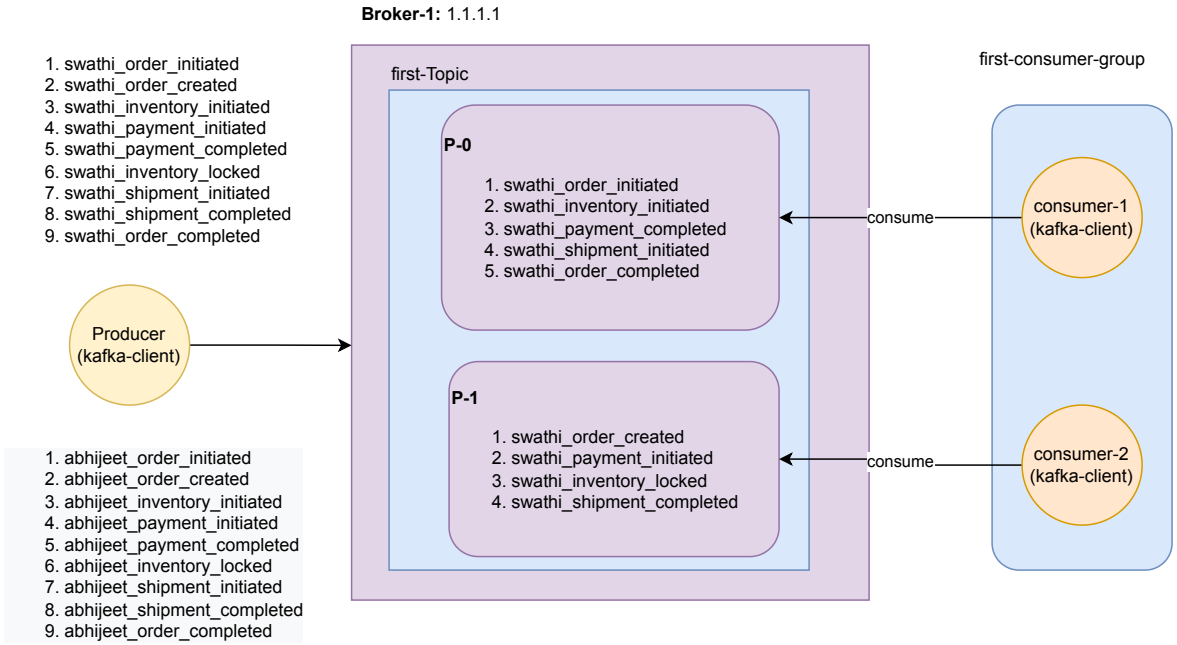
1. order_initiated
2. order_created
3. inventory_initiated
4. payment_initiated
5. payment_completed
6. inventory_locked
7. shipment_initiated
8. shipment_completed
9. order_completed

Negative Scenario:

1. order_initiated
2. order_created
3. inventory_initiated
4. payment_initiated
5. payment_failed
6. inventory_released
7. order_failed

1. swathi_order_initiated
2. swathi_order_created
3. swathi_inventory_initiated
4. swathi_payment_initiated
5. swathi_payment_completed
6. swathi_inventory_locked
7. swathi_shipment_initiated
8. swathi_shipment_completed
9. swathi_order_completed

1. abhijeet_order_initiated
2. abhijeet_order_created
3. abhijeet_inventory_initiated
4. abhijeet_payment_initiated
5. abhijeet_payment_completed
6. abhijeet_inventory_locked
7. abhijeet_shipment_initiated
8. abhijeet_shipment_completed
9. abhijeet_order_completed



Use Case# 1
(Sequential Processing)

Positive Scenario:

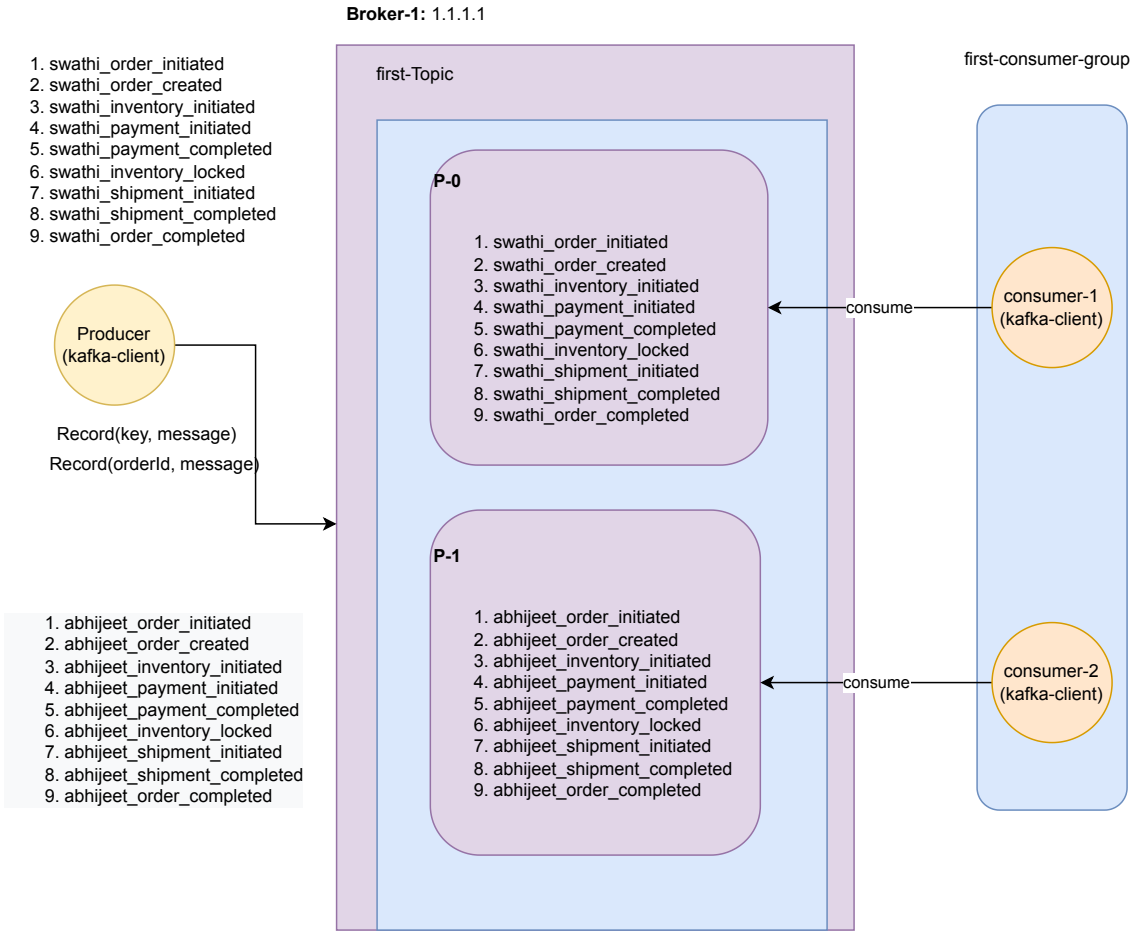
1. order_initiated
2. order_created
3. inventory_initiated
4. payment_initiated
5. payment_completed
6. inventory_locked
7. shipment_initiated
8. shipment_completed
9. order_completed

Negative Scenario:

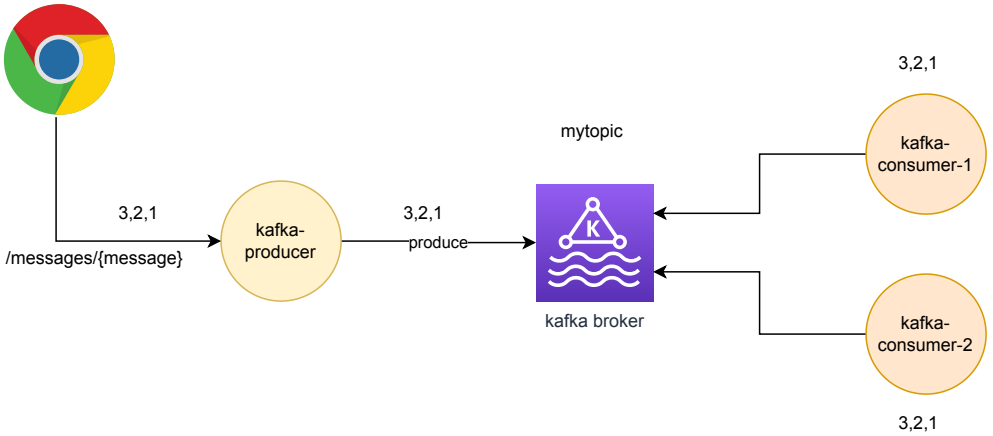
1. order_initiated
2. order_created
3. inventory_initiated
4. payment_initiated
5. payment_failed
6. inventory_released
7. order_failed

1. swathi_order_initiated
2. swathi_order_created
3. swathi_inventory_initiated
4. swathi_payment_initiated
5. swathi_payment_completed
6. swathi_inventory_locked
7. swathi_shipment_initiated
8. swathi_shipment_completed
9. swathi_order_completed

1. abhijeet_order_initiated
2. abhijeet_order_created
3. abhijeet_inventory_initiated
4. abhijeet_payment_initiated
5. abhijeet_payment_completed
6. abhijeet_inventory_locked
7. abhijeet_shipment_initiated
8. abhijeet_shipment_completed
9. abhijeet_order_completed



Day-2

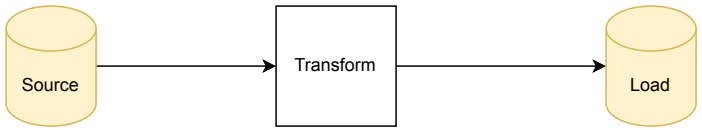

3,2,1
/messages/{message}
kafka-
producer
3,2,1
produce
mytopic
kafka broker
3,2,1
kafka-
consumer-1
kafka-
consumer-2
3,2,1

Confluent Kafka Products

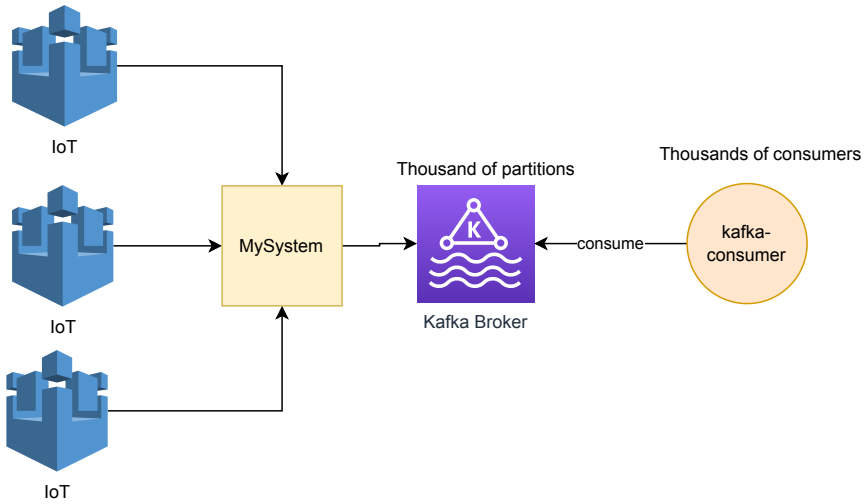
1. Kafka Connector
2. Kafka Stream
3. KSQL
4. Kafka GUI
5. Confluent Kafka Cloud (Managed Service)
6. AWS MSK (Managed Streaming for Apache Kafka)
7. Confluent CLI

Batch vs Stream

IMPS or NEFT

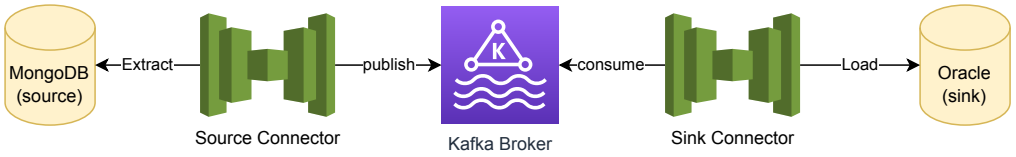


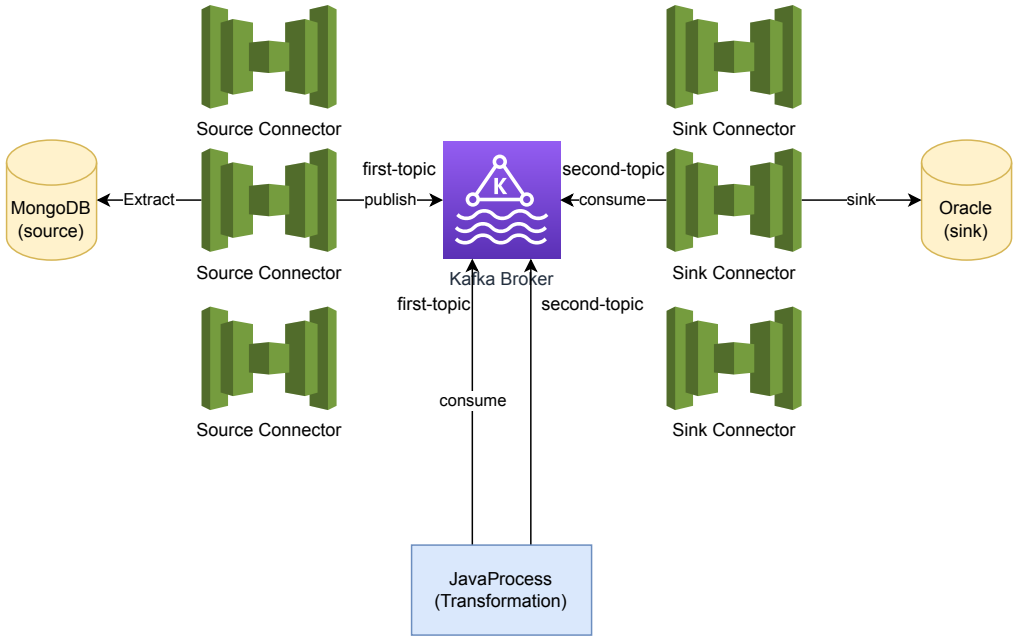
6 times/minute x 1000000 = 6 million events / minute



ETL
(Extract Transform Load)

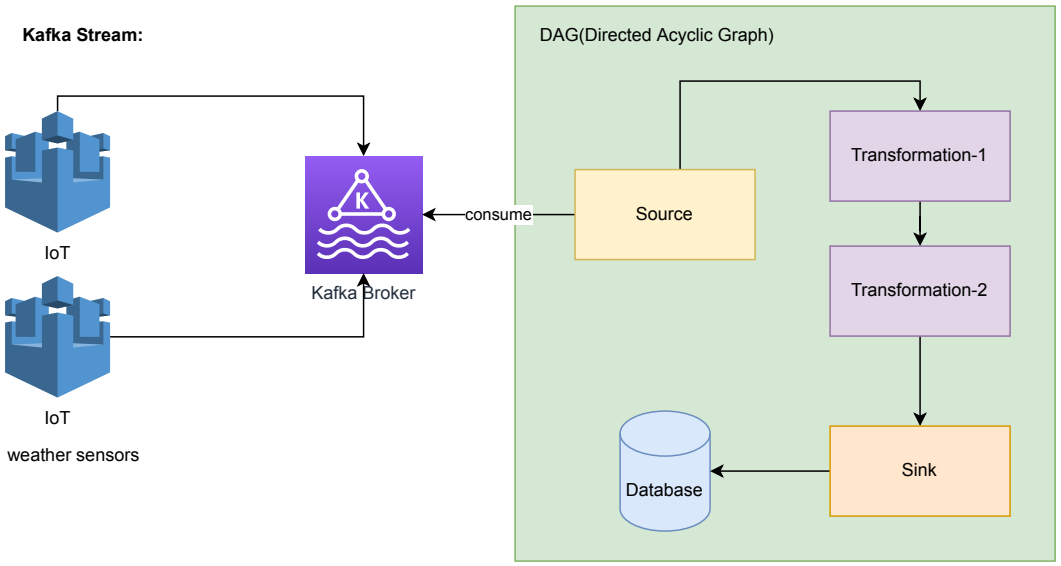
1. Kafka Connector



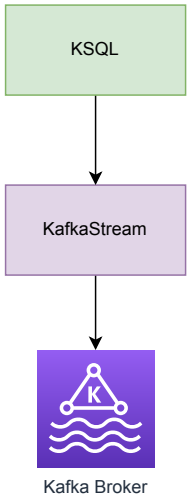


Confluent Kafka Products **Kafka Stream**

Kafka Stream:



KSQL

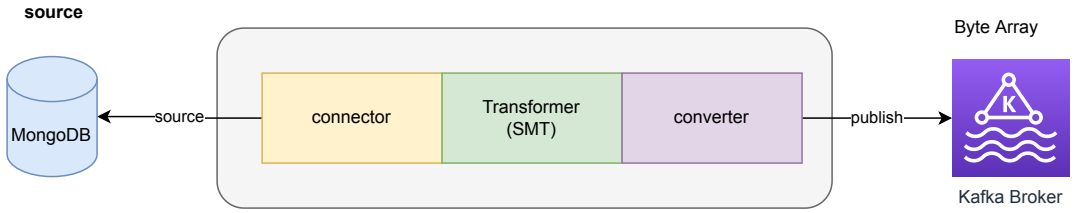


Day-3

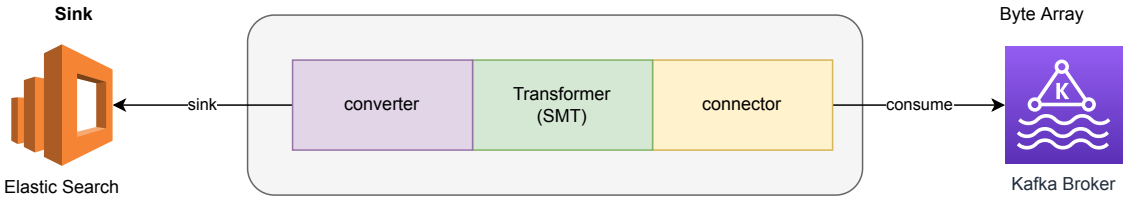
Kafka Connect

SMT: Single Message Transform

Source Connector

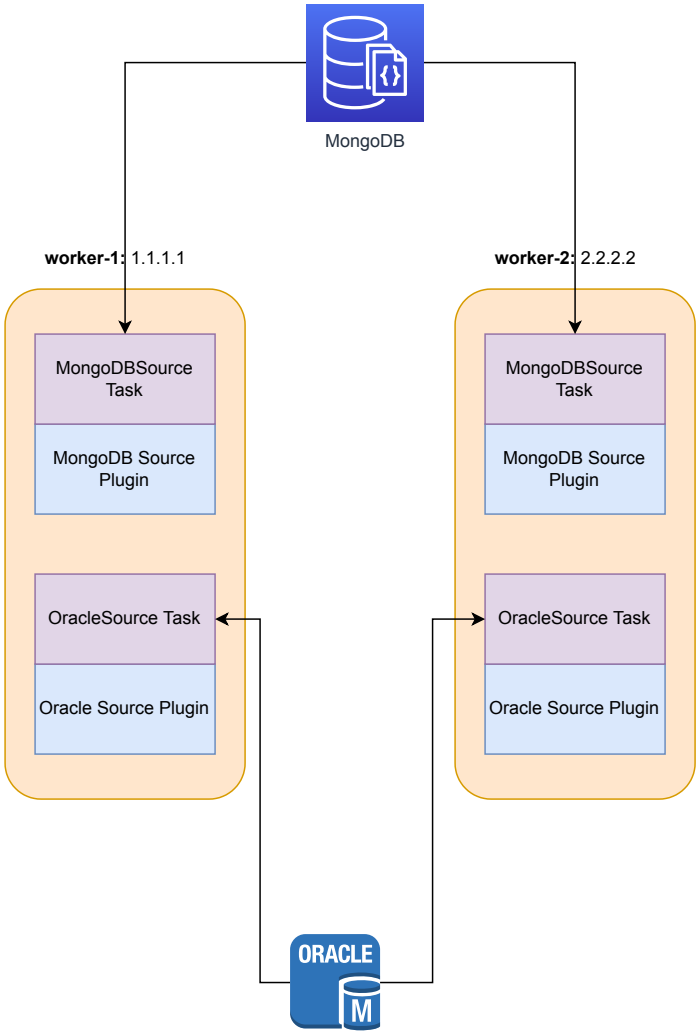


Sink Connector



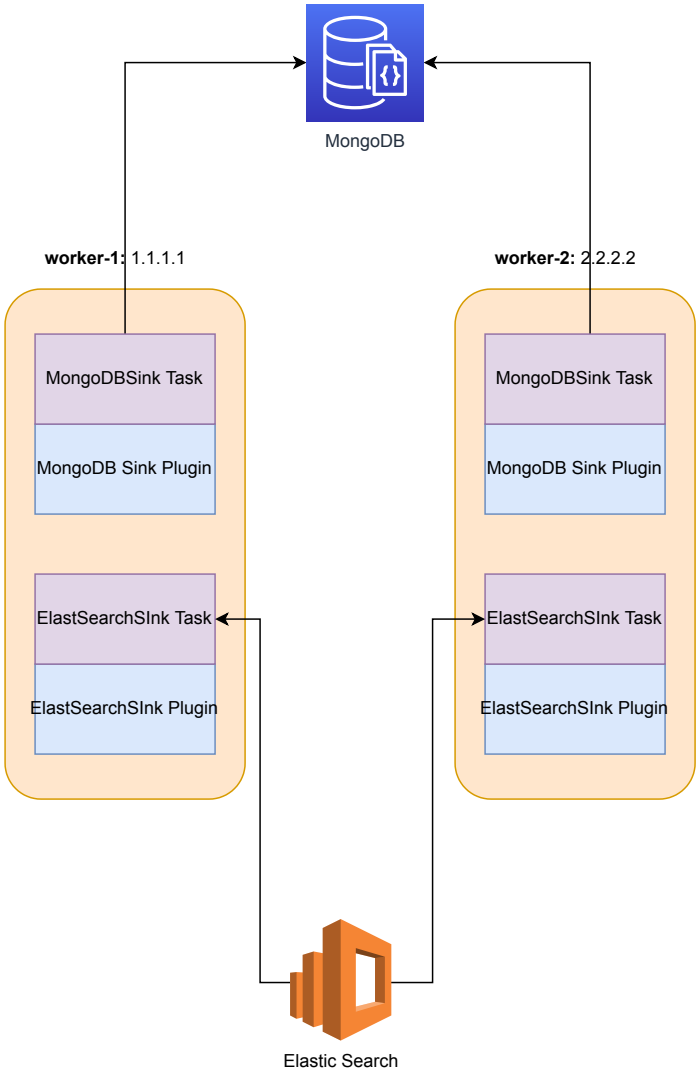
Kafka Connect

Worker and Tasks for Source



Kafka Connect

Worker and Tasks for Sink

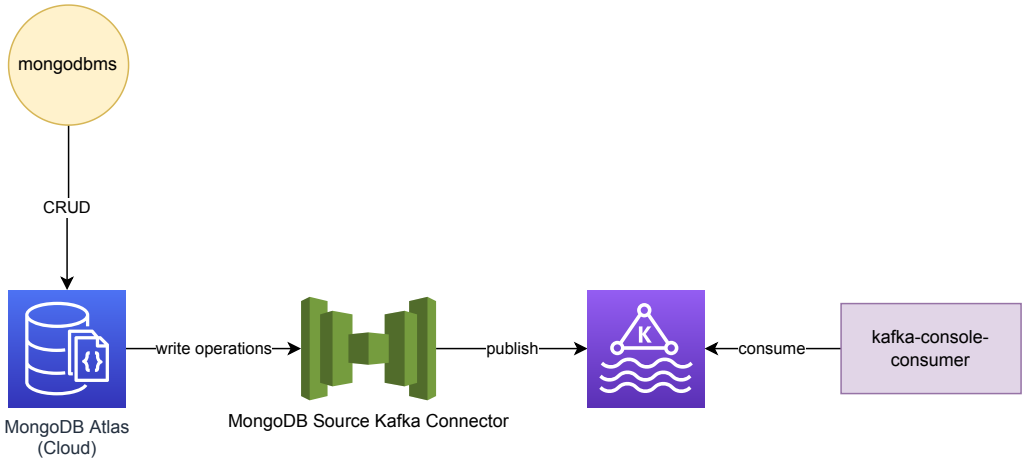


Relational vs MongoDB

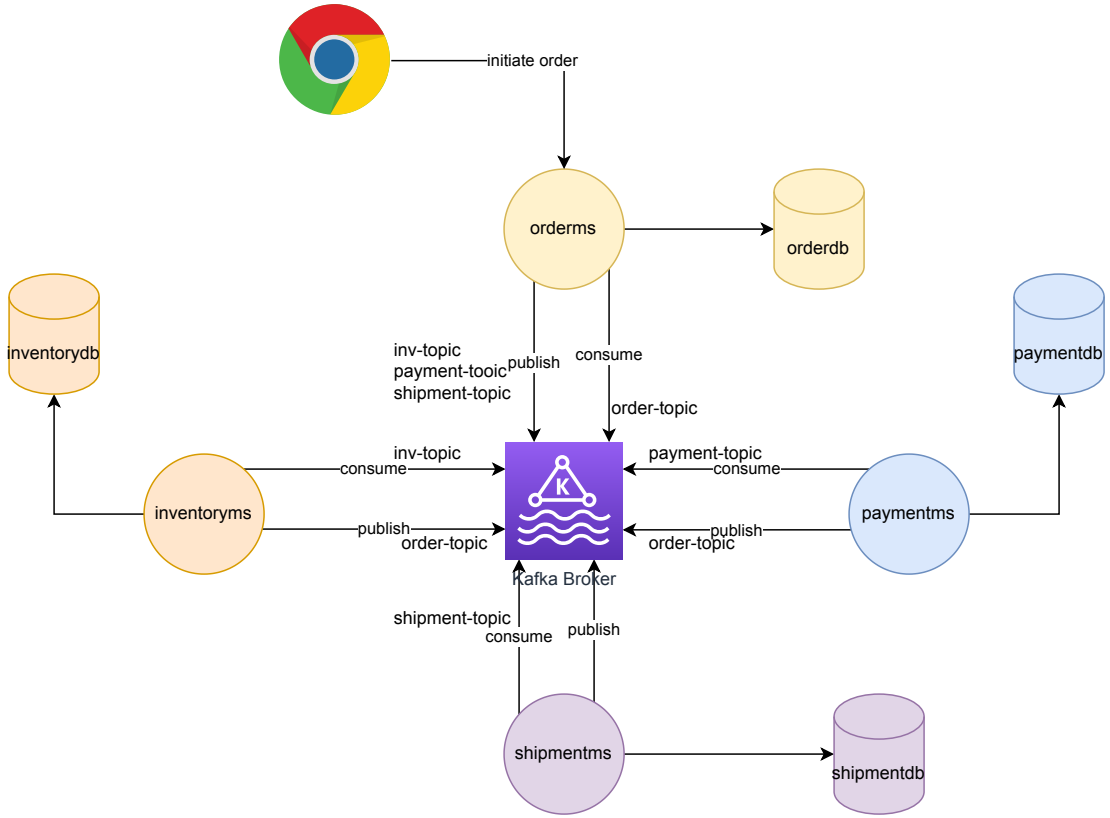
Relational	**No-SQL**	(BSON: Binary JSON)
1. Database	1. Database	
2. Table	2. Collection	
3. Rows	3. Document	
4. Colums	4. Field	

{"_id":{"$oid":"6207d6f9c023b166b109c861"},"amount":"1050","_class":"com.mongdbms.Payment"}
{"_id":{"$oid":"6207d6f9c023b166b109c861"},"amount":"1050","_class":"com.mongdbms.Payment"}

HandsOn



EDA/MDA
(Event Driven Architecture / Message Driven Architecture)



EDA/MDA - Spring Cloud Stream