# Kafka streams

Data lake is stationery

Stream – flow of data, or data in motion

When we are dealing with data which is in motion then it is called stream processing

Definitions: - event is an outcome of an action performed by an actor

Real time use cases:-

1. Flipkart orders - Ordering event , each order in kafka when u place an order and reject an order everything is an event and placed in kafka
2. Stock market trades – real time rates
3. Advertisement click events- monitors when an user clicked on an add or not , if the user clicked it may go as an event and will be placed in kafka
4. All POS machine UPI transactions – when any person transacted on a pos machine it may put a message to kafka
5. Ordering food in our mobile app
6. Booking a cab- once we book it may send a message to kafka
7. In facebook reading a post and creating a post which is a continuous flow data

Read from a partition and write into another partition – writing a post- like a post, all these activities are generating some content

1. Healthcare monitoring
2. ­Traffic monitoring
3. Finding fraud transactions- monitor the transaction on bank account and identify if it is fraud or not in real time and stop the transaction if it is fraud
4. Personalized customer experience- flipkart and myntra
5. Real time bidding

Data coming into my feed, If someone liked by post, commented and tagged me and who commented and liked our post and someone tagged you or if someone tagged ur friend all those events may flow into ur database partition

Misc use cases

1. Infrastructure management –monitor cpu health ram, rom
2. Campaign management- whether recent offers in myntra is driving traffic or not

## Design considerations for data integration solutions

2 applications should be integrated by following points

1. Time sensitivity 2) Decoupling 3)Data format evolution 4) Reliability 5) scalability 6)D
2. Decoupling

Says we should minimize the dependency between source and destination systems within milliseconds we should consume it should be very fast

1. We cant constraint the data format , source system may not send messages to target system in desired formats we should have flexibility to convert the messages

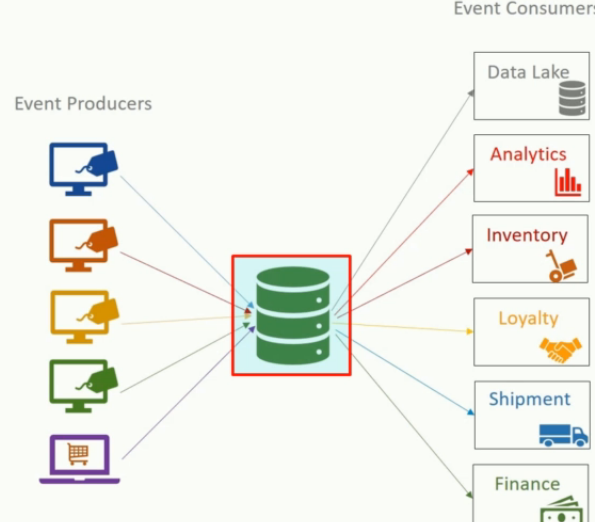
Src may send json and target may need only xml

1. Reliability- even if the consumer went down offline for some times messages should be stored until customer comes online – kafka does that, all messages will be stored in broker partition’s file segment means messages are written to files so those are reliable
2. Scalability – we want horizontal scaling this data integration solution must support at least millions of messages per second

Ways of communication

1. Shared database
2. File transfer – here will transfer file by file
3. Messaging
4. RPC remote procedure call – Its is by rest calls – worst for scalable application

### **1) Shared database**

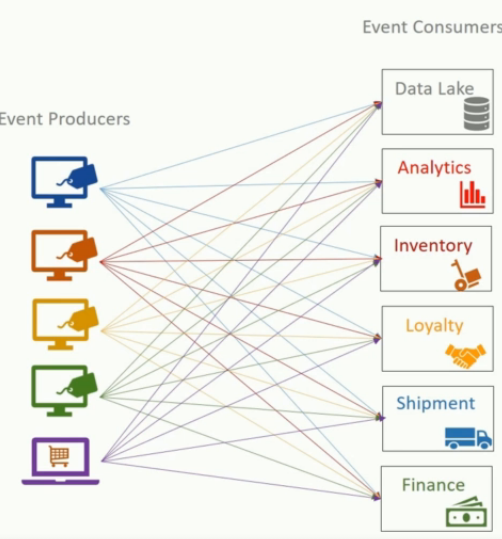
1) This is a good approach for storing now and query later not good to deal with the data in motion and not good for realtime and to deal in milliseconds

2) reg decoupling – all consumers may not support RDBMS and 3) not good for data format evolution because producer may send json and consumer may need xml

5) and this is not horizontally scalable not but the problem is

### **RMI- RPC**

Fails on decoupling (data exchange formats) and scalability , for million messages should we do million rest calls



**1) consumer offline :-** remote method invocation this is like REST service call, we cant send millions of data over REST, main issue – what if the consumer is offline so is chain broken

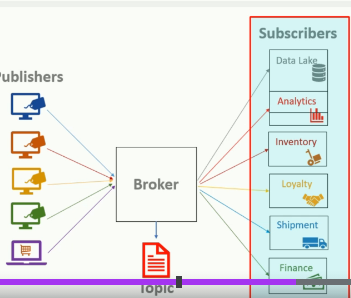
, in kafka all messages will be stored in topic-folder-files when ever he is online he can consume all the messages from the beginning

1. **Resend same message to all consumers –** un necessary REST calls if we want to send millions of same messages to all 5-6 consumers its terrible to make 6 REST calls to 6 services if one is down how can the producer send the same data again

### **Files**

They are not good at time sensitivity

### **Messaging**

Here all publishers will send message to the broker , consumer will poll for the messages