```
                                    Date : 25-May-21
                                    Spring Boot 9AM
                                      Mr. RAGHU
                    ----------------------------------------------
                            Pub/Sub Communication using JMS

*) By default every Producer and Consumer app is P2P Communication
  as internally key: spring.jms.pub-sub-domain is set to false.

*) To convert P2P Application into Pub/Sub, then add key:
   spring.jms.pub-sub-domain with value true.

**) Producer/Consumer code is same for both P2P and Pub/Sub.
    Just define another conusmer Application.

=============================================================
*) Topic is created if spring.jms.pub-sub-domain=true
*) Based on no.of conusmers, actual Message cloning is done
   [Clone -Create a copy of actual message].
*) Consider no.of consumers are 10 then one message is delivered
   to 10 consumers from MOM.

---Advantages-----------------------------------------------------
a. MOM s/w will store data to avoid message lose
   until consumer receives it.

b. MOM S/w can support sending data as Gobal Format
    (JSON/XML)

c. It is good to use for simple and less data trasfer
   between multiple systems.

----Limitations------------------------------------------------------
a. In case of data is too large (like GBs) then MOM becomes
   slow even data lose may occure.

b. In case of no.of consumers got increased it may become very slow.

c. Here, MOM containes single broker instance. If it down (or) not
responding
   then data lose may occured. Not supporting Load Balancer concept.

d. It is fully Protocol dependent (TCP).

======================================================================
*) Apache Kafka :-

=> It is also called as Advanced Message Queue Protocol supporting
tool
   (AMQP)
=> It is from Apache with Scala 2.13.
=> It supports send/recive data between various systems.
   It is language indepenent and even protocol independent.
=> It contains only Topic Concept. Not supporting Queue.
*) We can use topic concept to send message to one or more consumers.
  So, queue concept is removed.
```

---Kafka S/w Overview--
a. One Message Broker is used for Read data from Topic Section, clone it
and send data to consumer based on topicName.

b. All mesage brokers and Topic section is fully controlled by R&D Server
Zookeeper. ie creating topic, store data, allocate Message broker to consumer
..etc

c. Full s/w is called as EcoSystem = Cluster + bootstrap Server + Topic Section

d. Cluster = Collection of Message Broker Instances.
When we start Kafka S/w it is created with one Instance.
On Demand (No.of consumers) Message Brokers are controller by Zookeeper.

e. Topic are memories which store data in partitions based.
Inputs: topicName, partition factor.
Based on replication factor no.of consumers even identified.
default is taken as 1 for both partition, replication.

==

Q) Kafka Full s/w is called as?
A) Eco System = Zookeeper + Topics + Cluster

Q) What is cluster? what is it default size on startup?
A)
Cluster it is a collection of Message brokers.
Message broker is a medaitor s/w sends data to consumer
Default is one.

Q) What is Topic Section? How will it store data?
A) Topic Section = It is memory that holds data.
Stores in K=V , topicName=Data using Partitions concept.
Index starts from zero. Default size is one.

Q) What is MR? when will it be executed?
A) MR = Message Replica creates colning objects
for actual data.
It gets executed before sending data to consumer.

===