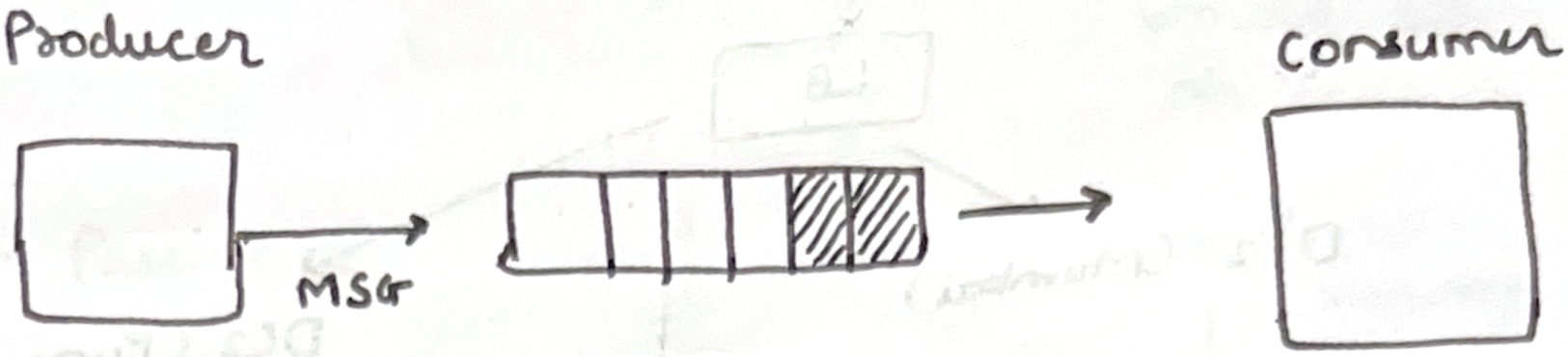
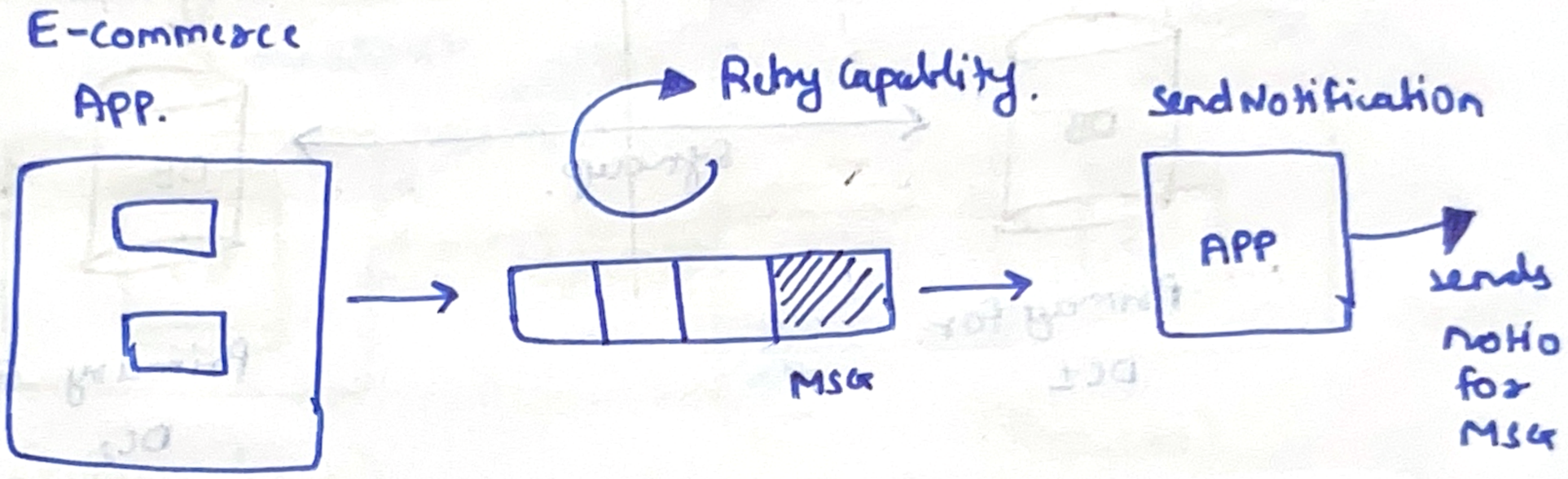
Messaging Queues :

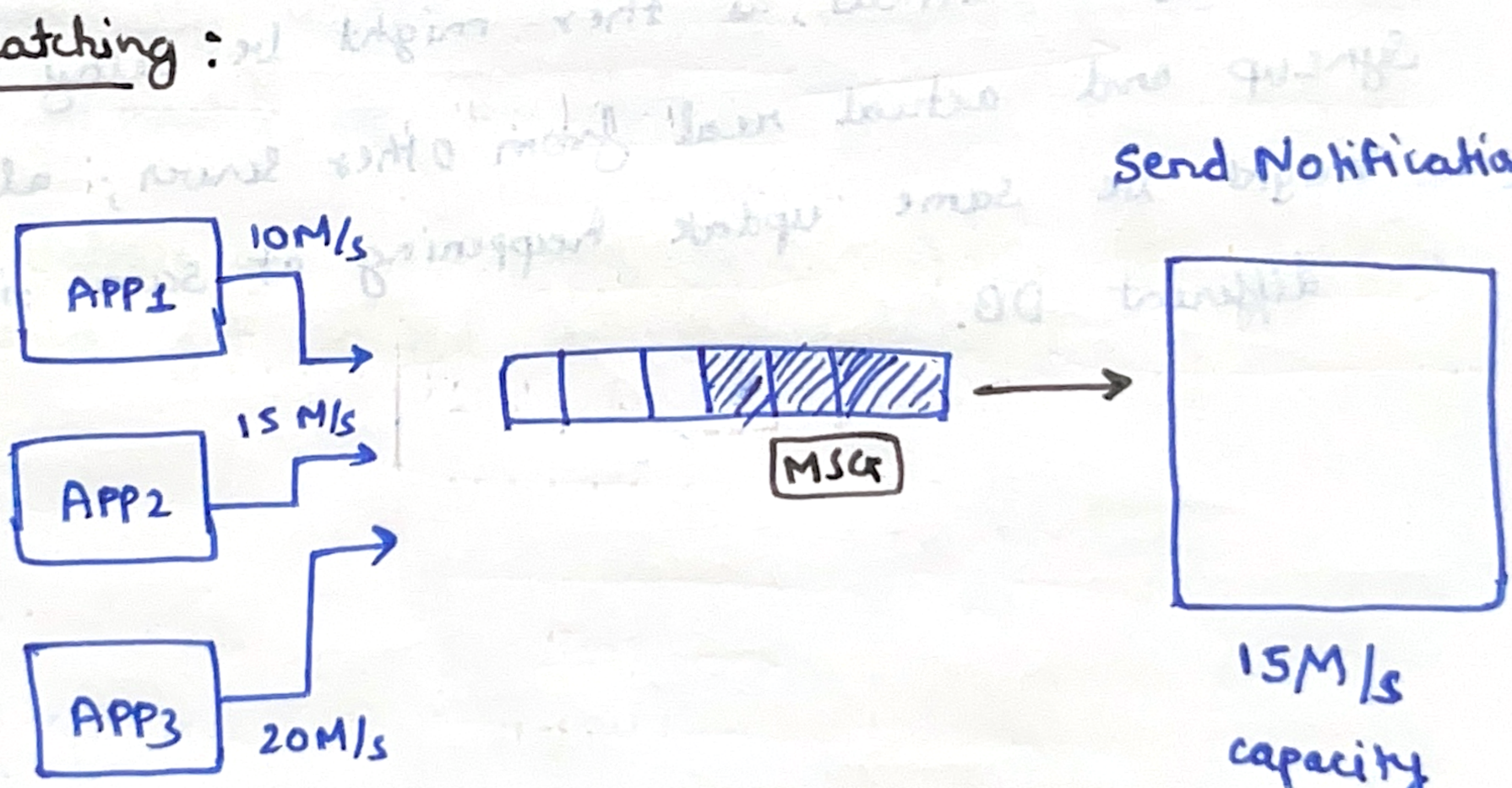


USE CASE :

① Async nature :



② PACE matching :

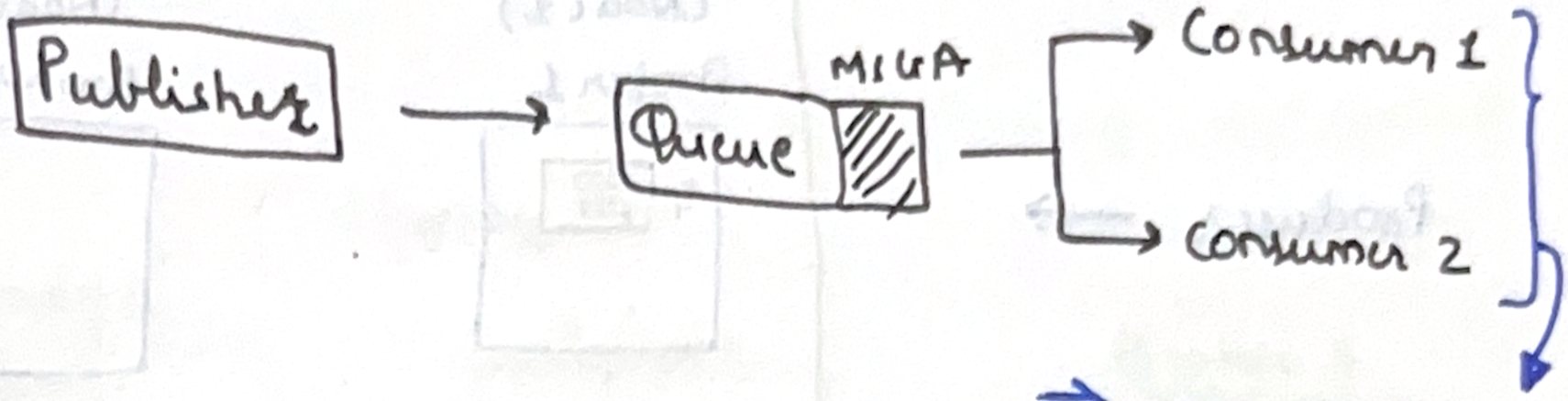


also helps when number of Producers are very high.

Consumer will consume at its own PACE.

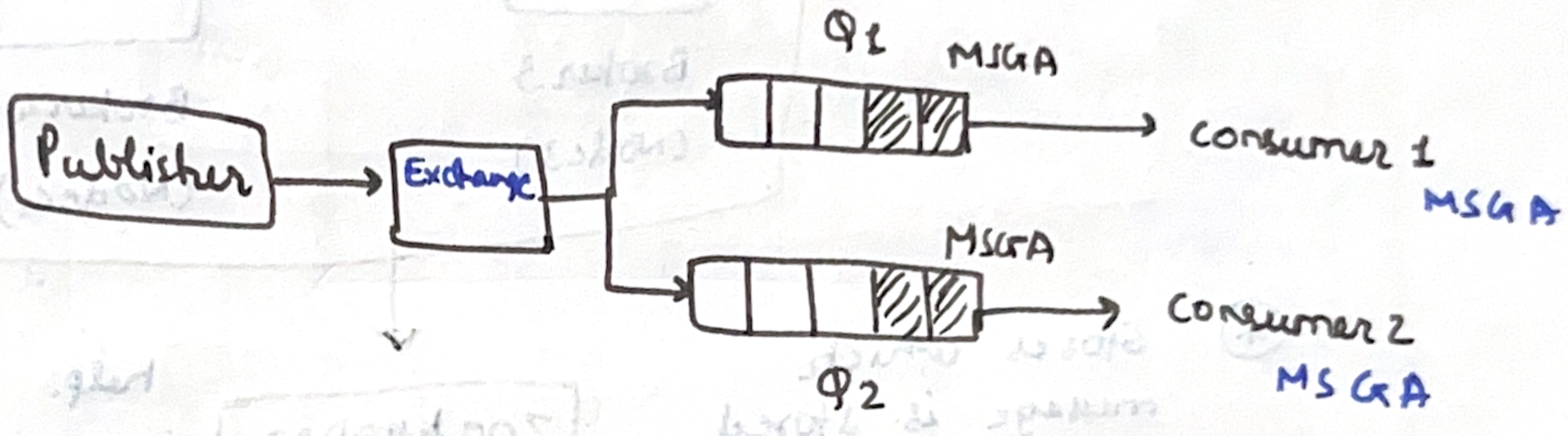
P2P and Pub/sub :

P2P:



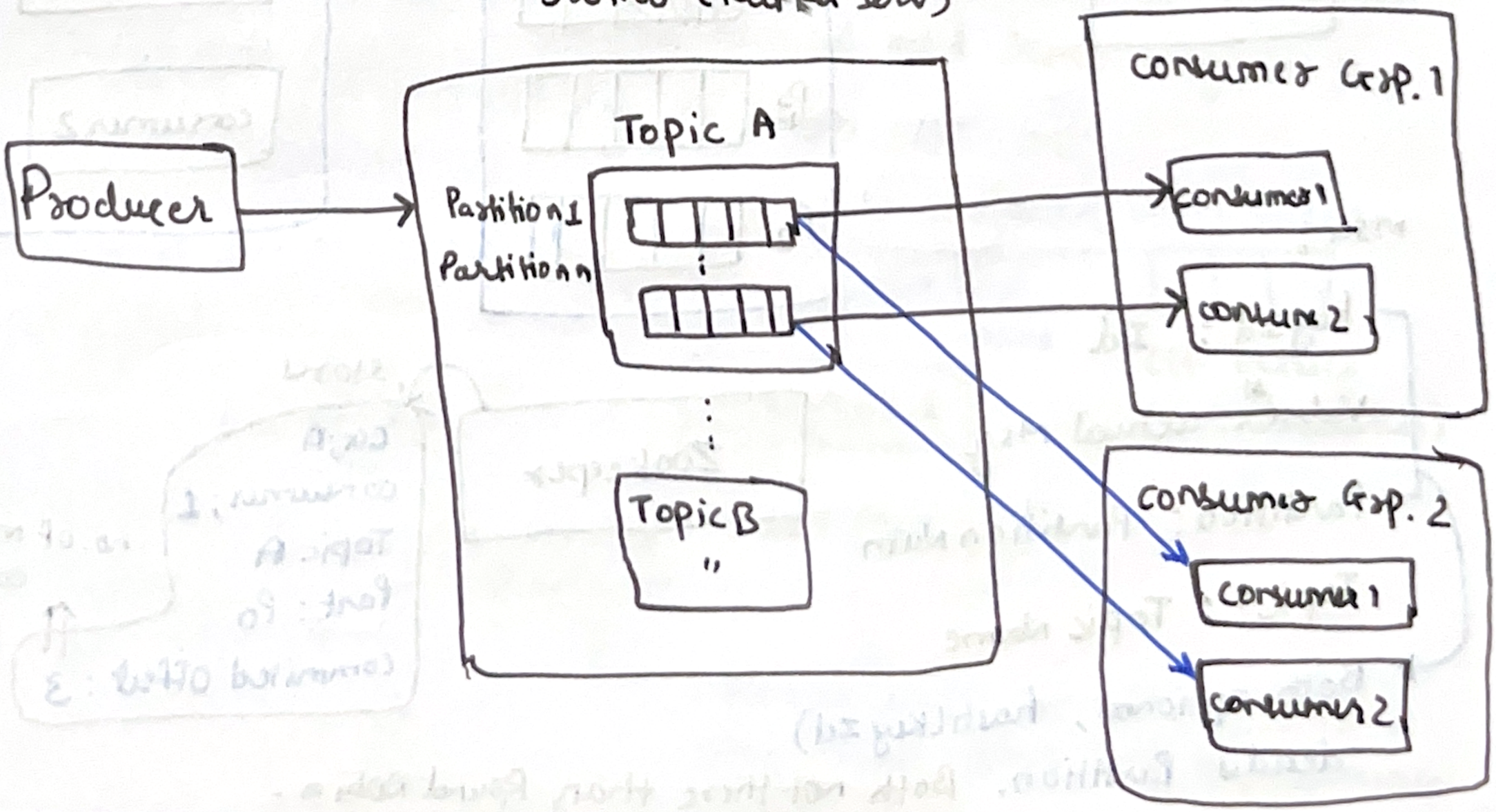
⇒ anyone of the two will consume the message.

Pub/sub:

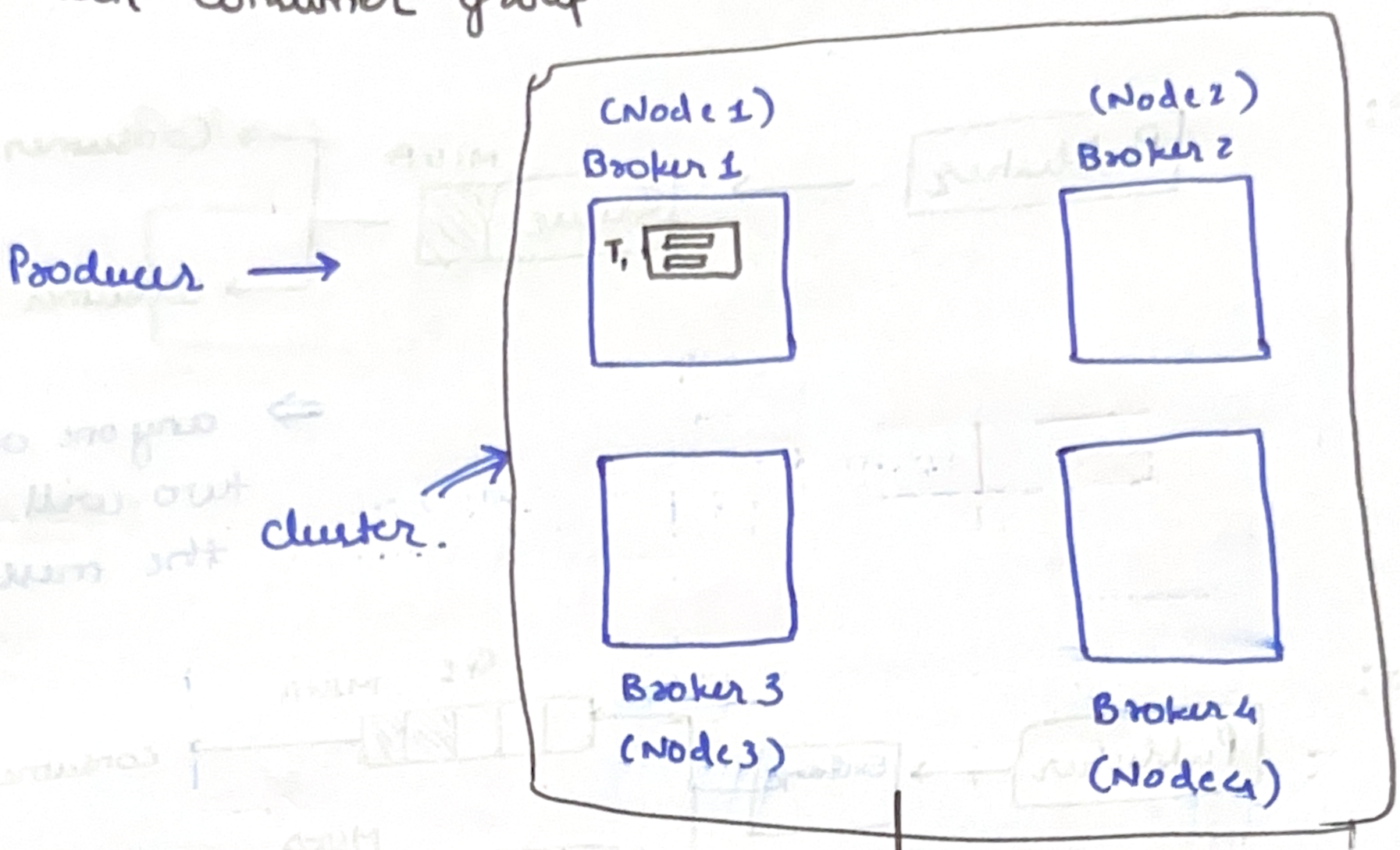


How Kafka works ?

- Producer
- Consumer
- Consumer Group
- Topic
- Partition
- offset
- Broker
- cluster
- Zookeeper



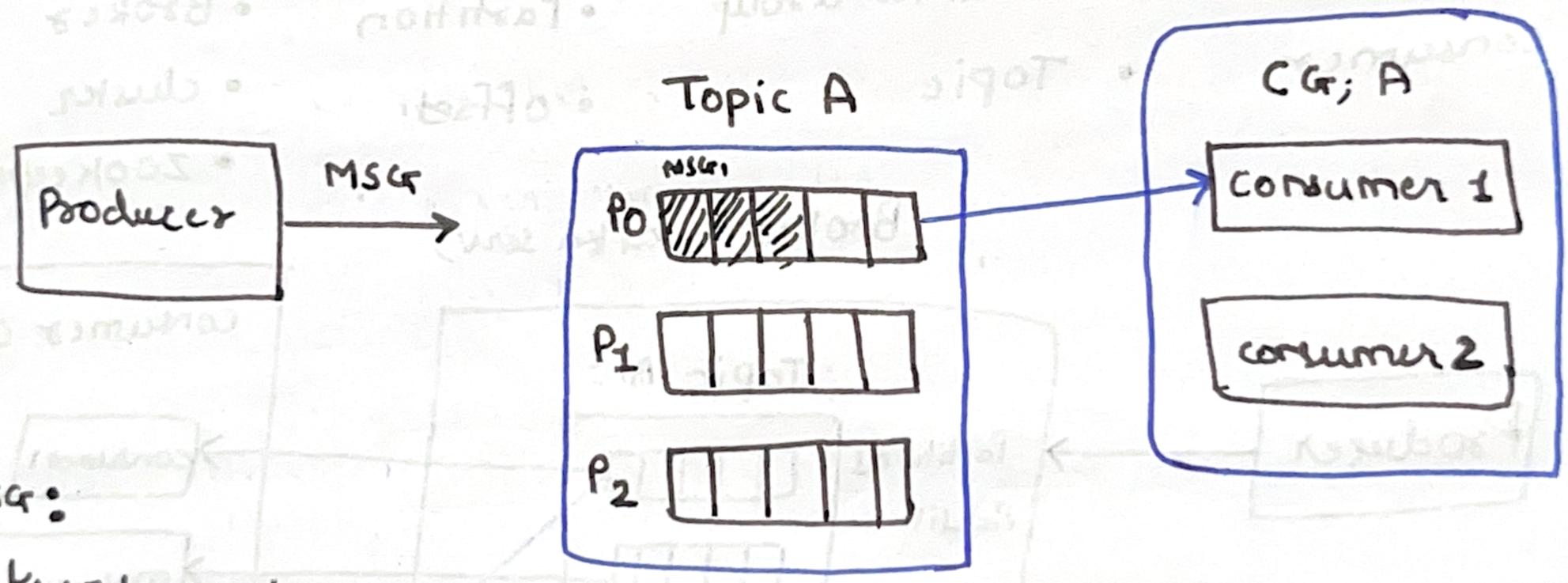
* each partition is accessed by one consumer per each consumer group



* Stores which message is stored by which Broker's which Topic's which Partition

Zookeeper

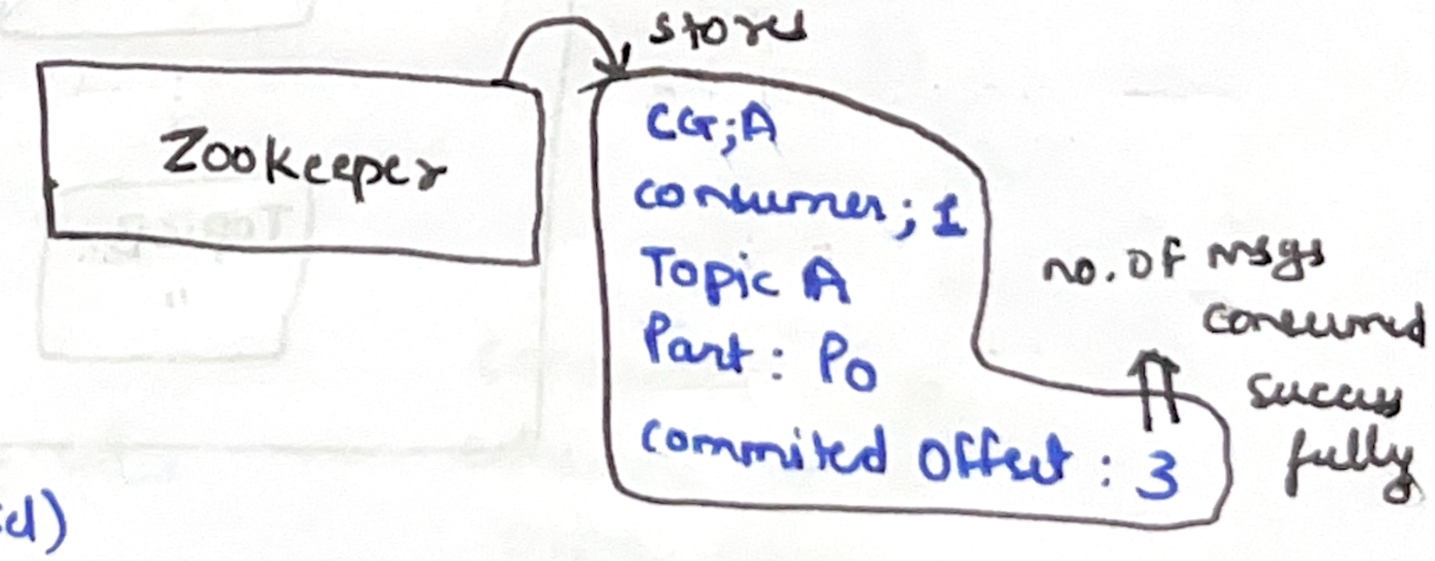
help. internal communication



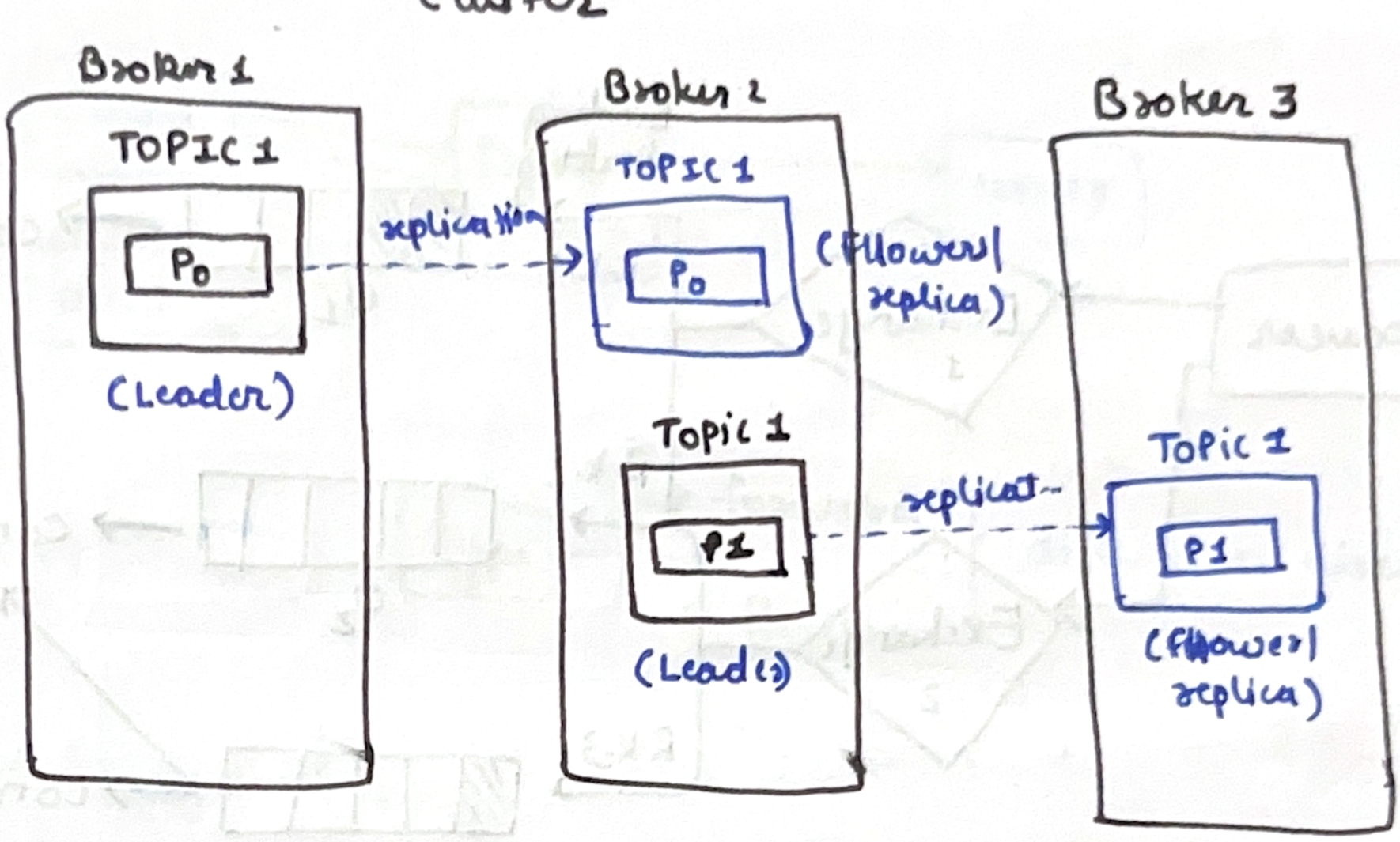
MSG:

- KeyId : Id
- Value* : actual Msg
- Partition : PartitionNum
- Topic* : Topic Name

Both optional, hash(KeyId) decides Partition. Both not there than Round Robin.



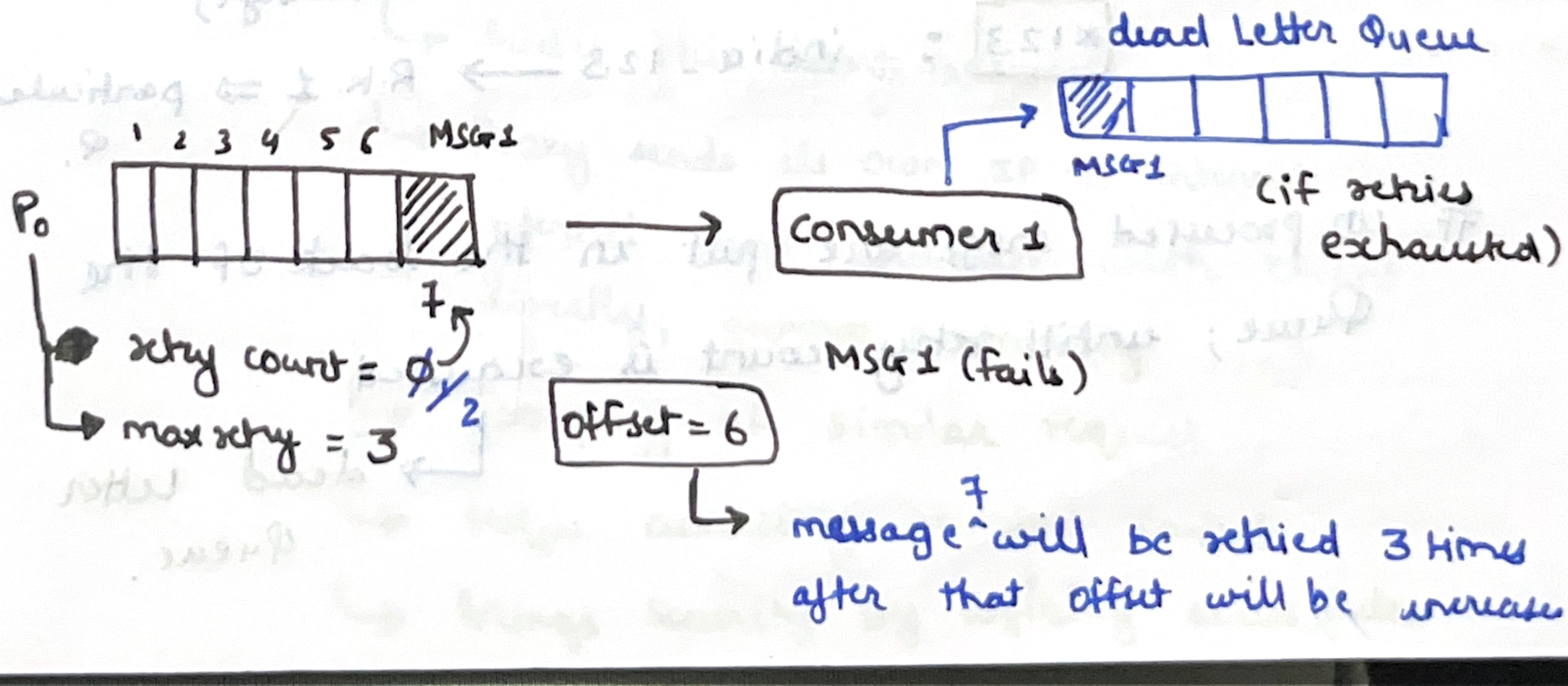
in above case if consumer 1 goes down, kafka picks another consumer and uses offset to replace older (failed) consumers place.



Read/write → Leader

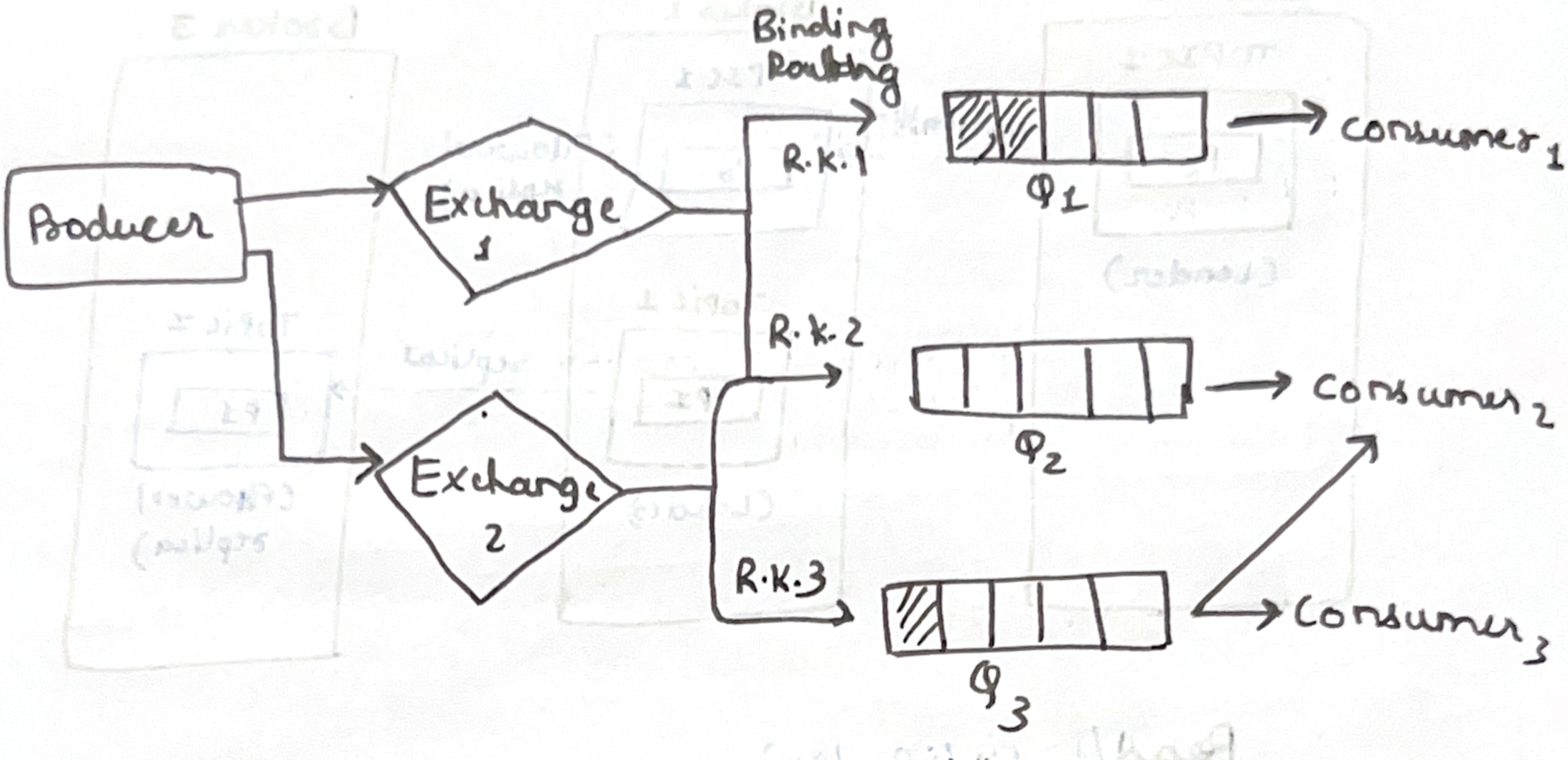
if leader goes down, follower becomes leader

⇒ Syncing would be continous, if msg comes to Leader, replica would add same msg in the corresponding partition.



* Kafka is pullbased; consumer pull the message out of Queue and processes it
  ↳ keeps on checking

* Rabbit MQ: push based approach;



Exchange:

① Fan Out type (Broadcast to all associated Qs)

② direct type (R.K. is sent with Msg to put msg in particular Q)

③ Topic Exchange (wildcard exchange)

*123 : india_123 ⟶ R.K. 1 ⇒ particular Q.

unprocessed msgs are put in the back of the Queue; untill retry count is exausted
↳ dead letter Queue.