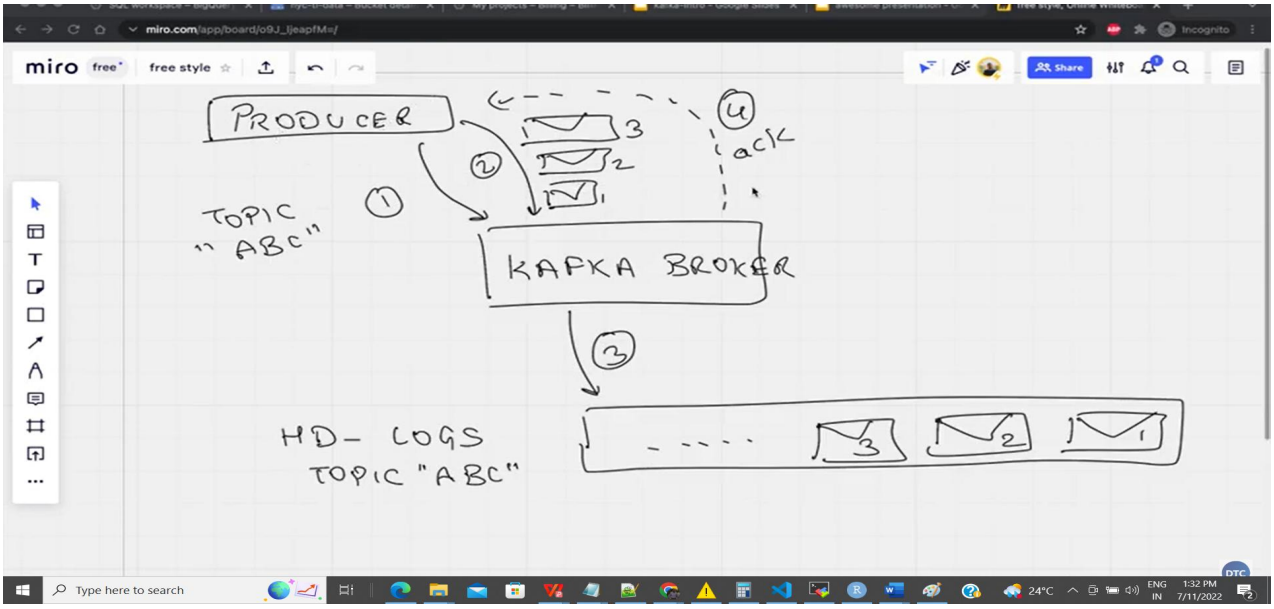
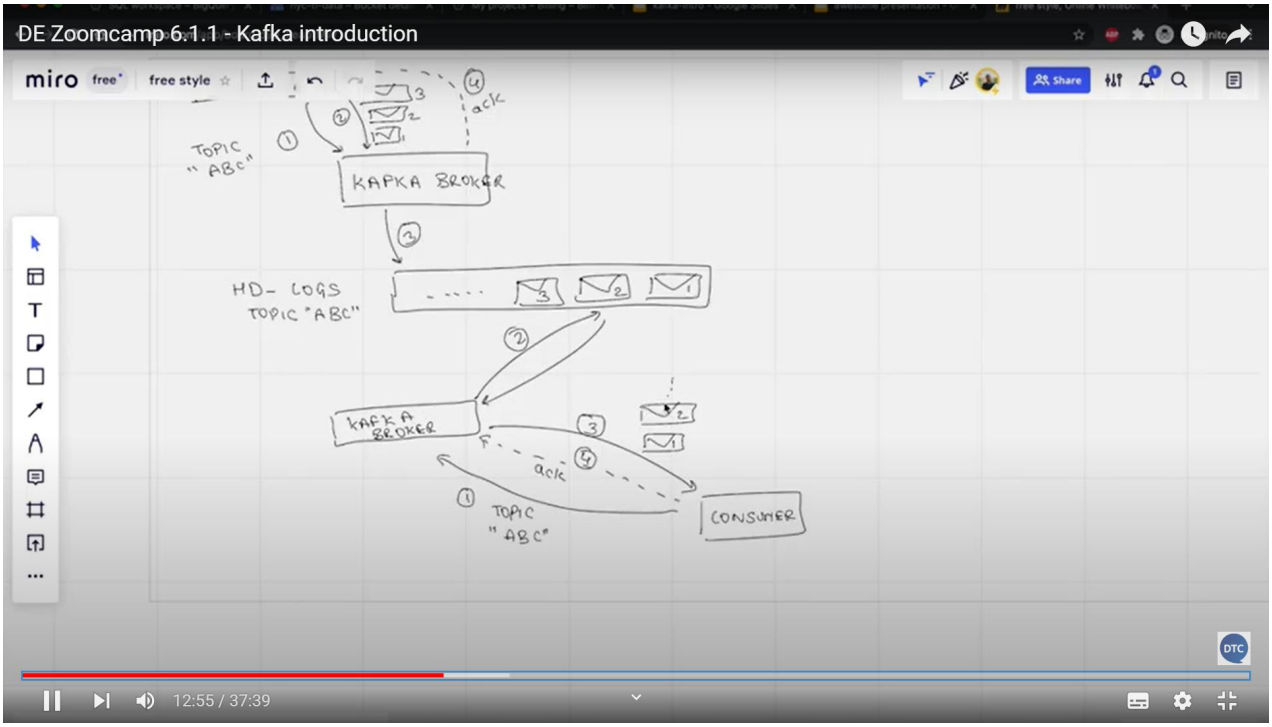
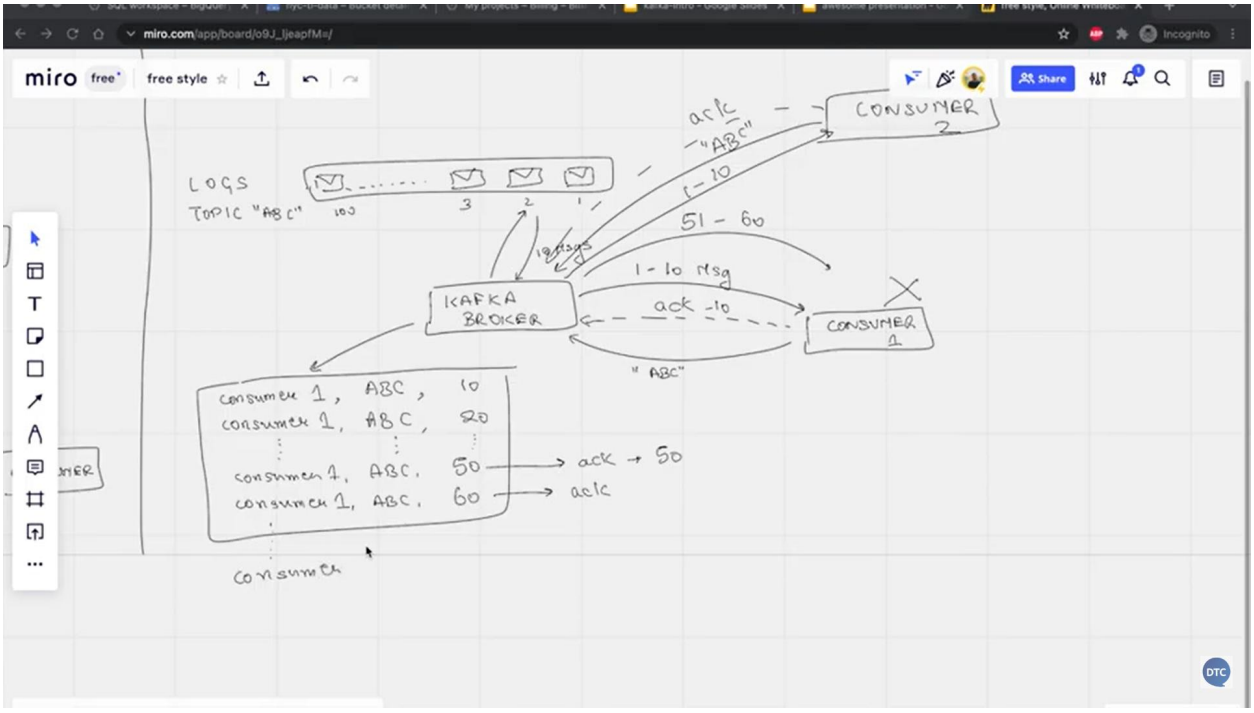
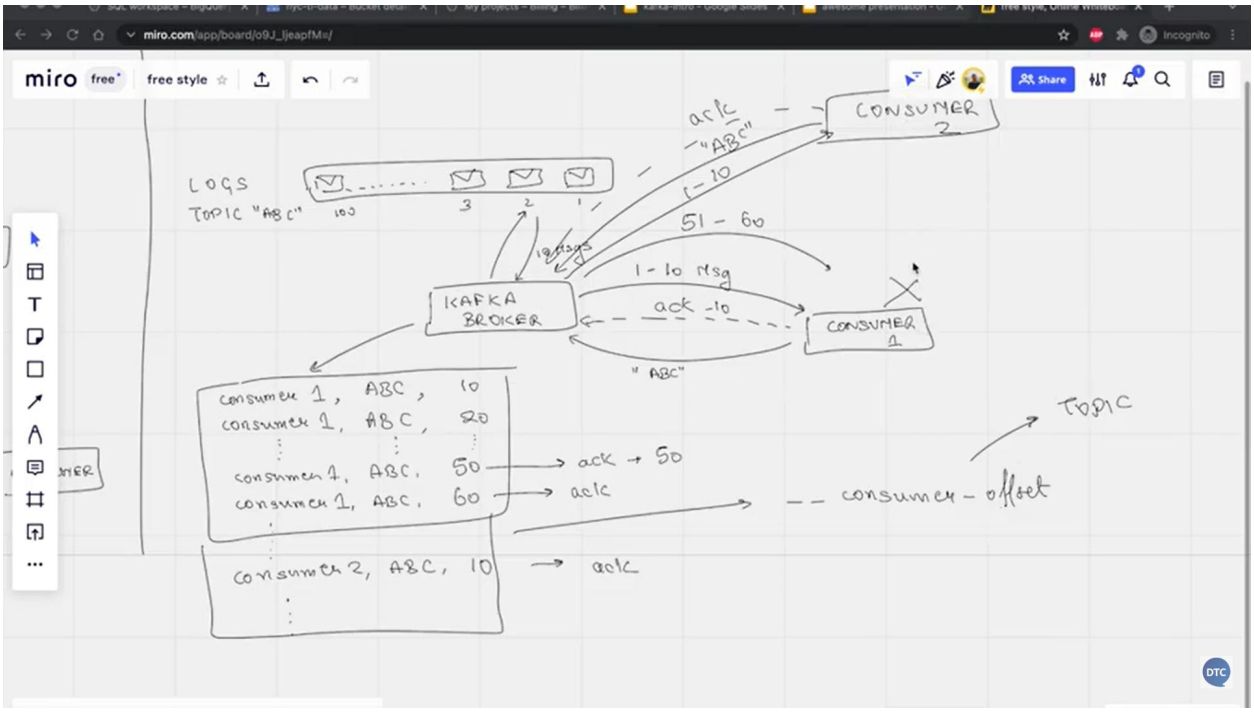
Introduction to Kafka





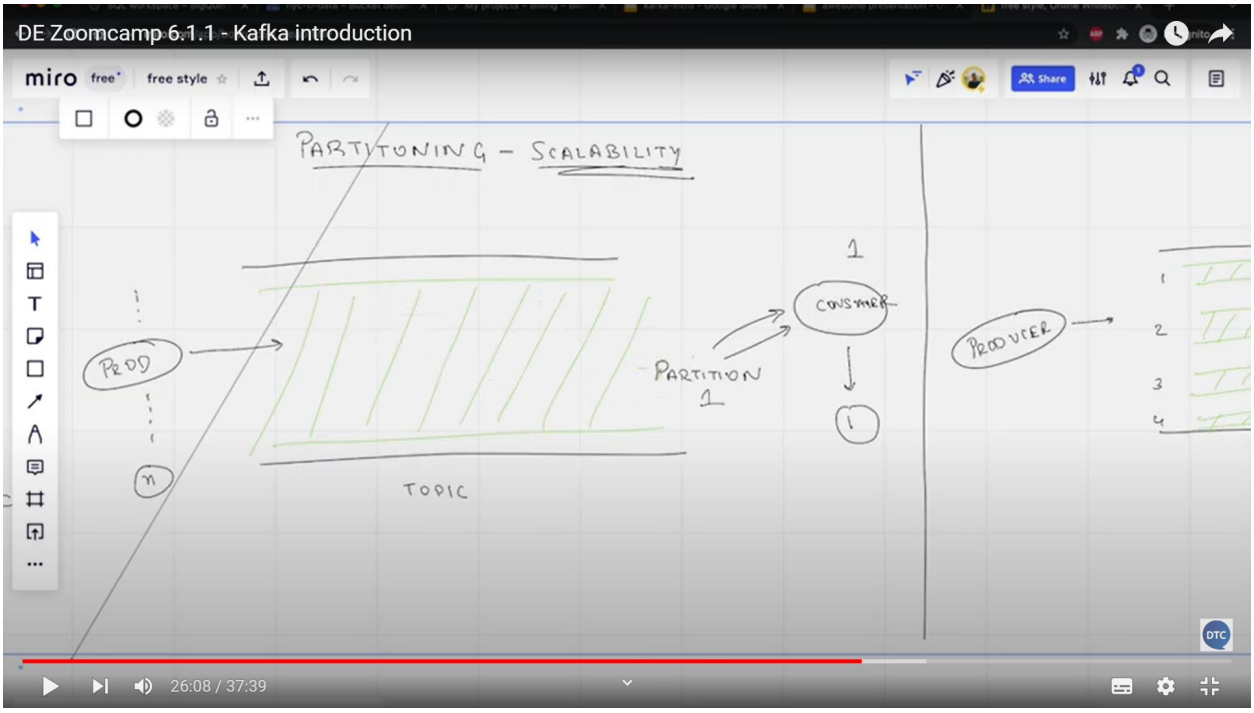


Consumer offset was maintained by kafka to know about num. of msg's read by each consumer. If a consumer dies and comes back, then it starts reading messages from where it left before dies.

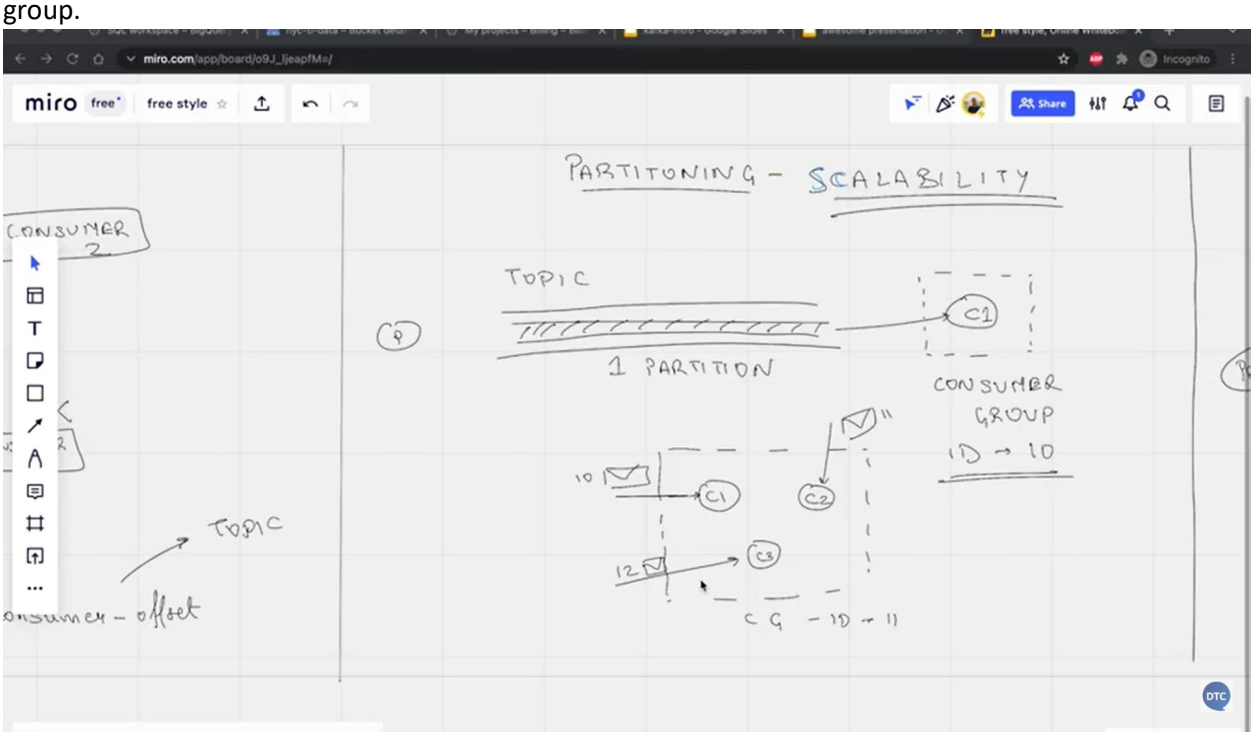


Kafka Partition: kafka partition happens over real time data to accomplish scalability.

1-producer-1partition-1consumer

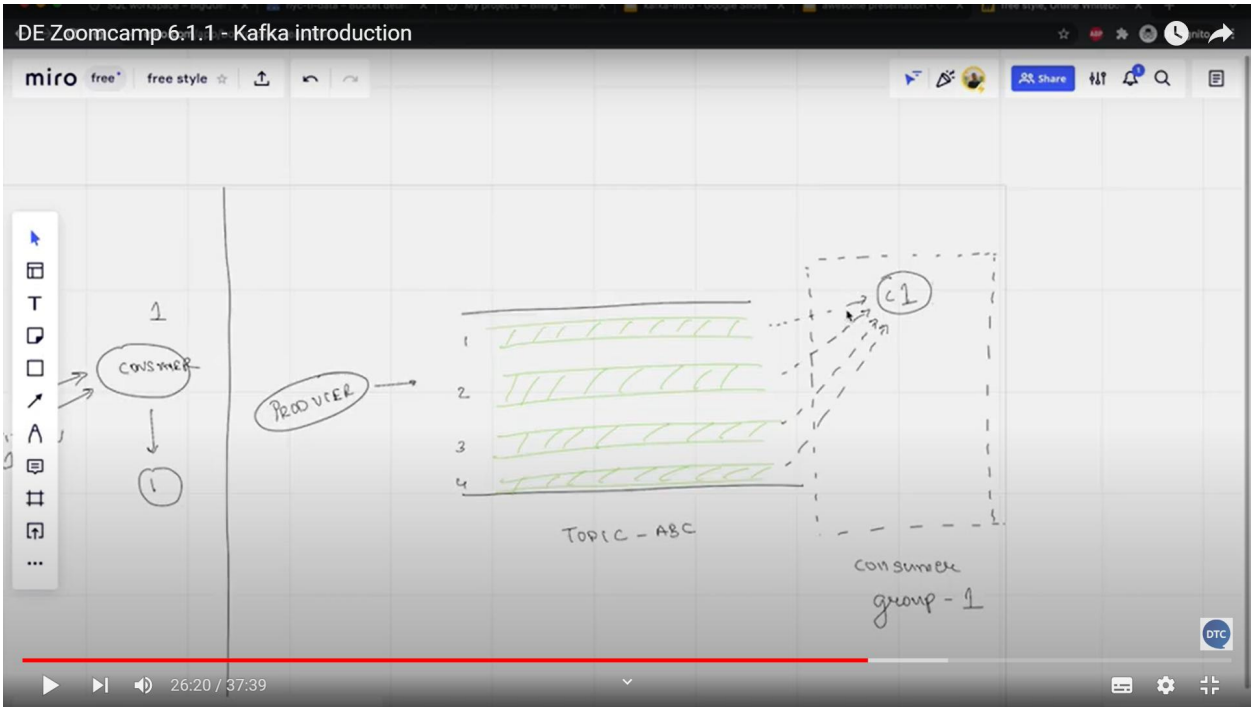


Consumer group:A consumer group can consist of multiple consumers, but only one consumer can receive a msg from kafka-broker for the subscribed topic. Once a consumer from a group received msg from broker the same can't be received by other consumer in the
group.

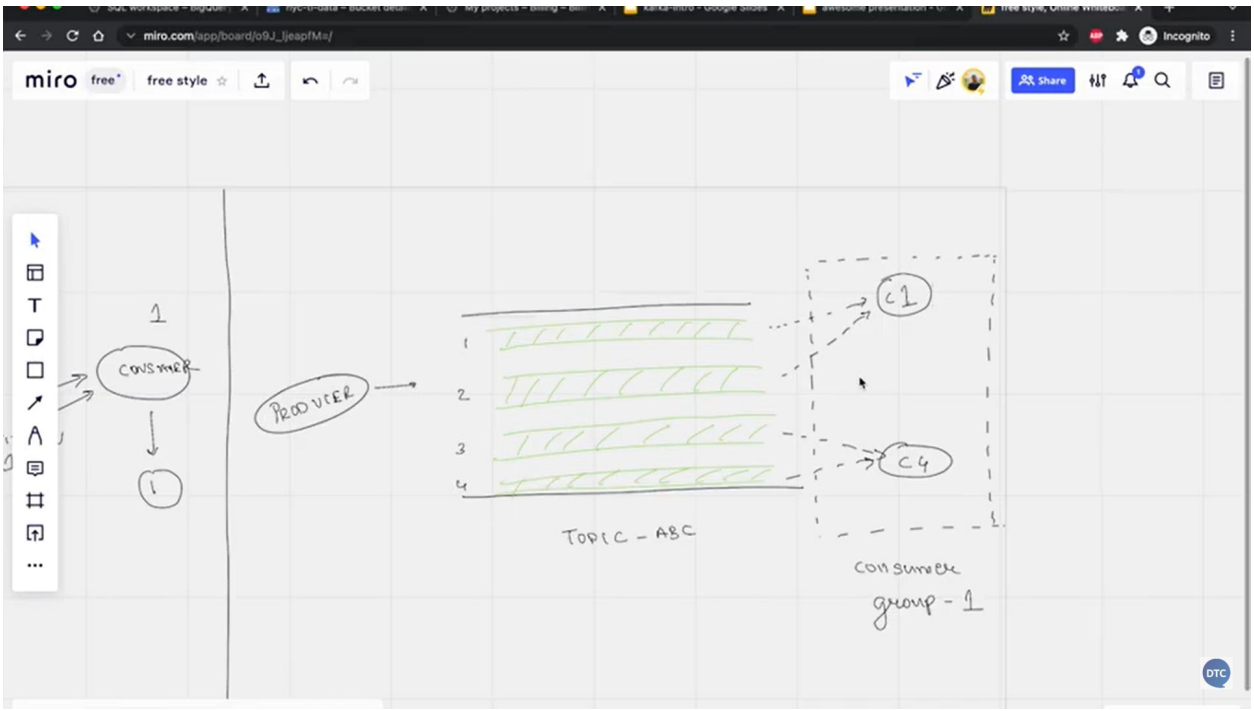


Producer pushing messages to a topic that consist of 4 partitions.

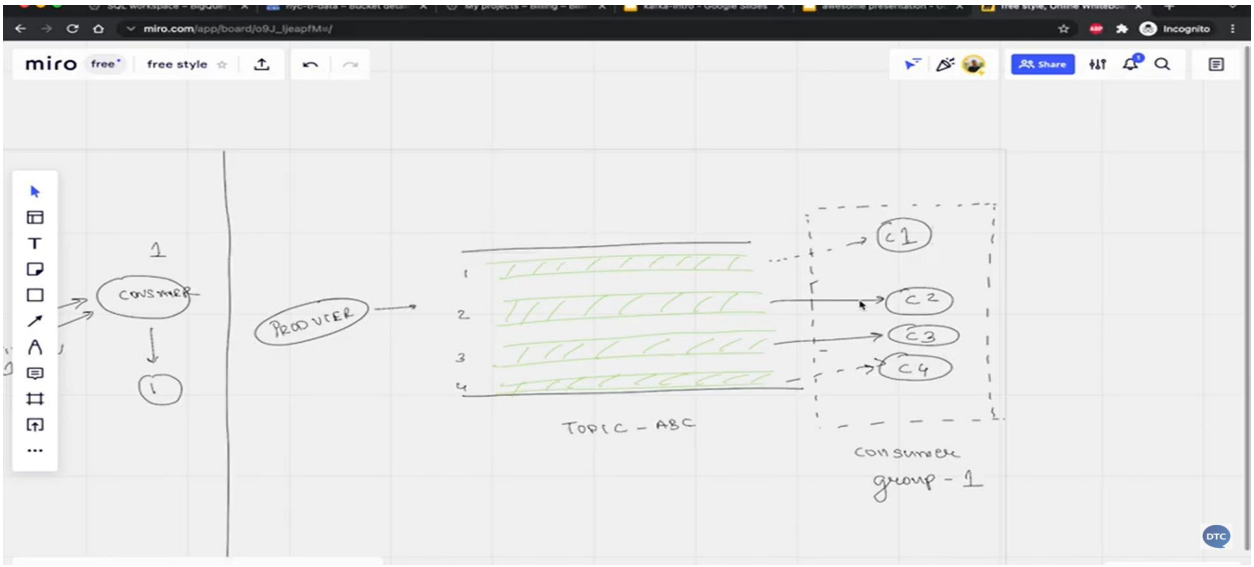
1-producer-4partition-1consumer group(1consumer)



1-producer-4partition-1consumer group(2consumers) 2 partition gets assigned to consumer1 and another 2 partition gets assigned to consumer2. Here reading the msg twice fast as earlier case.

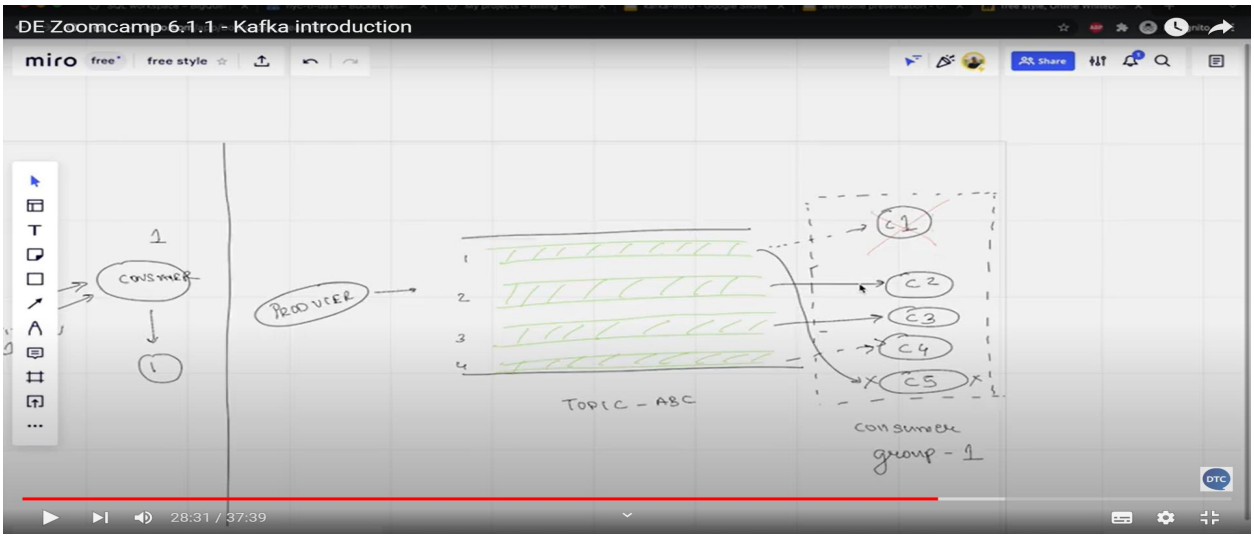


If 4 consumers were introduced in a consumer group, each 1 partition is assigned to each consumer in a group. Here the reading is 4 times speed as compare to 1-consumer case.



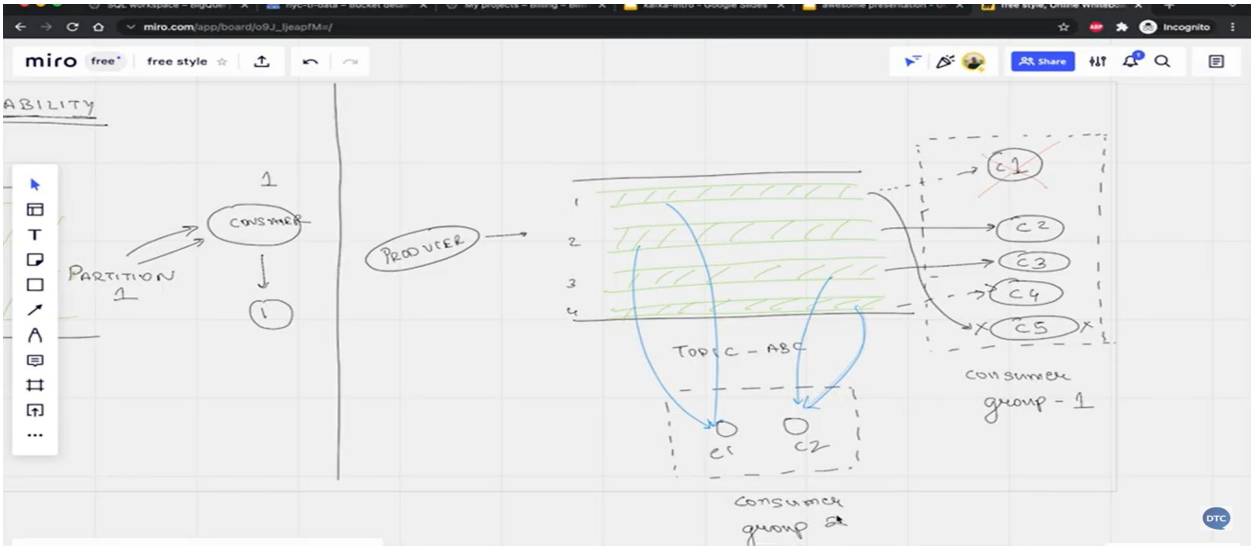
Number of consumers = Partition size+1

good practice to create a consumer greater than one partition size. Additional consumer keeps idle, if any one of a consumer in a group fails this consumer take over that job.



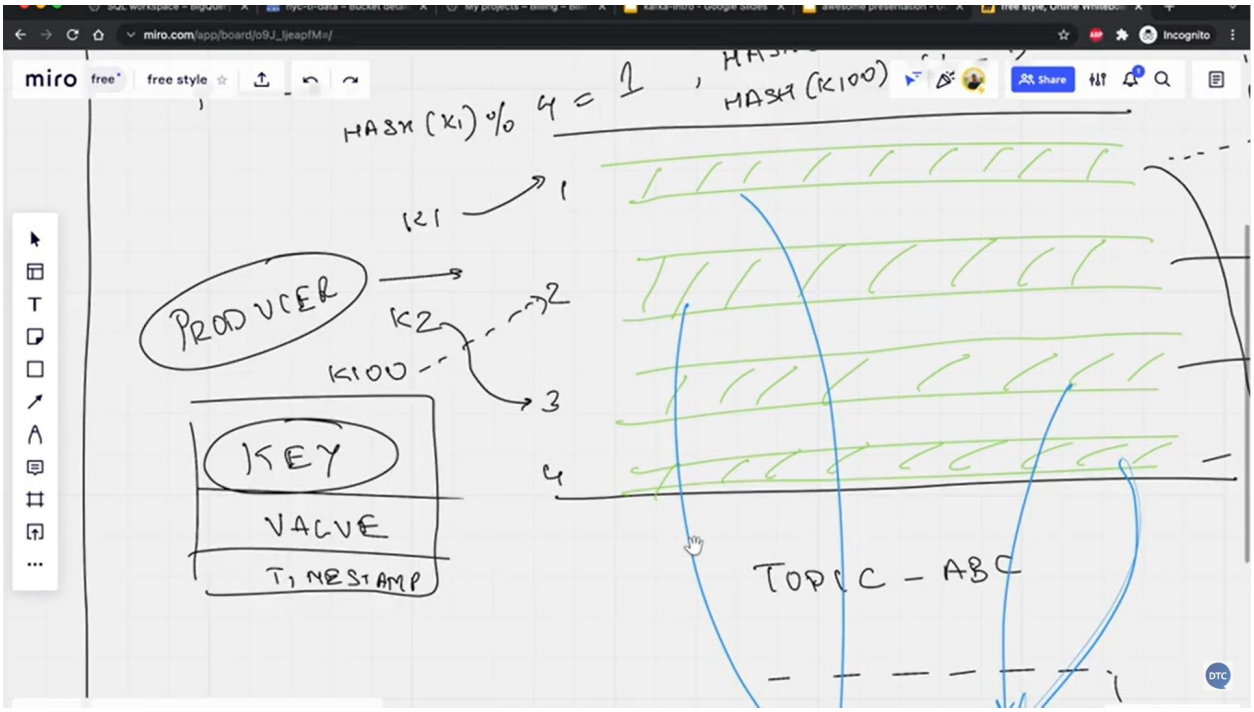
Consumer1- 4partition - 2consumer groups:

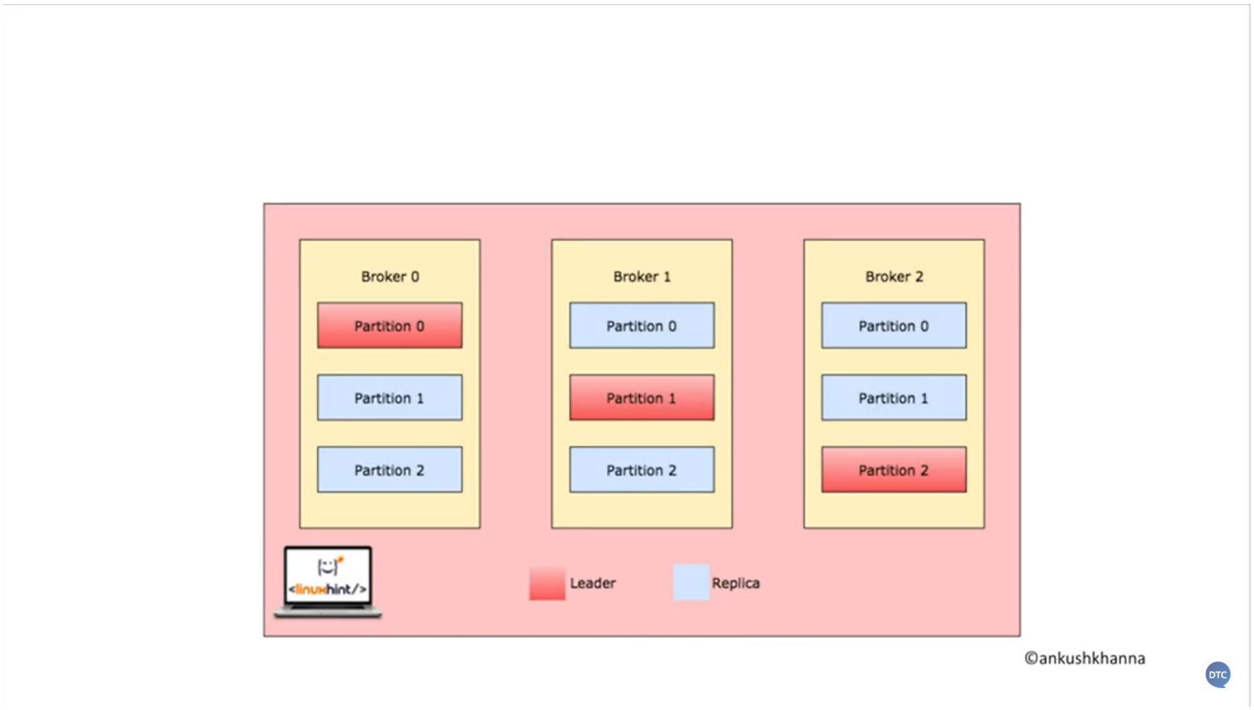
Here partitions were assigned to consumer group1 and consumer group2 separately. Number of messages read by CG1, CG2 can be different and not related to each other.



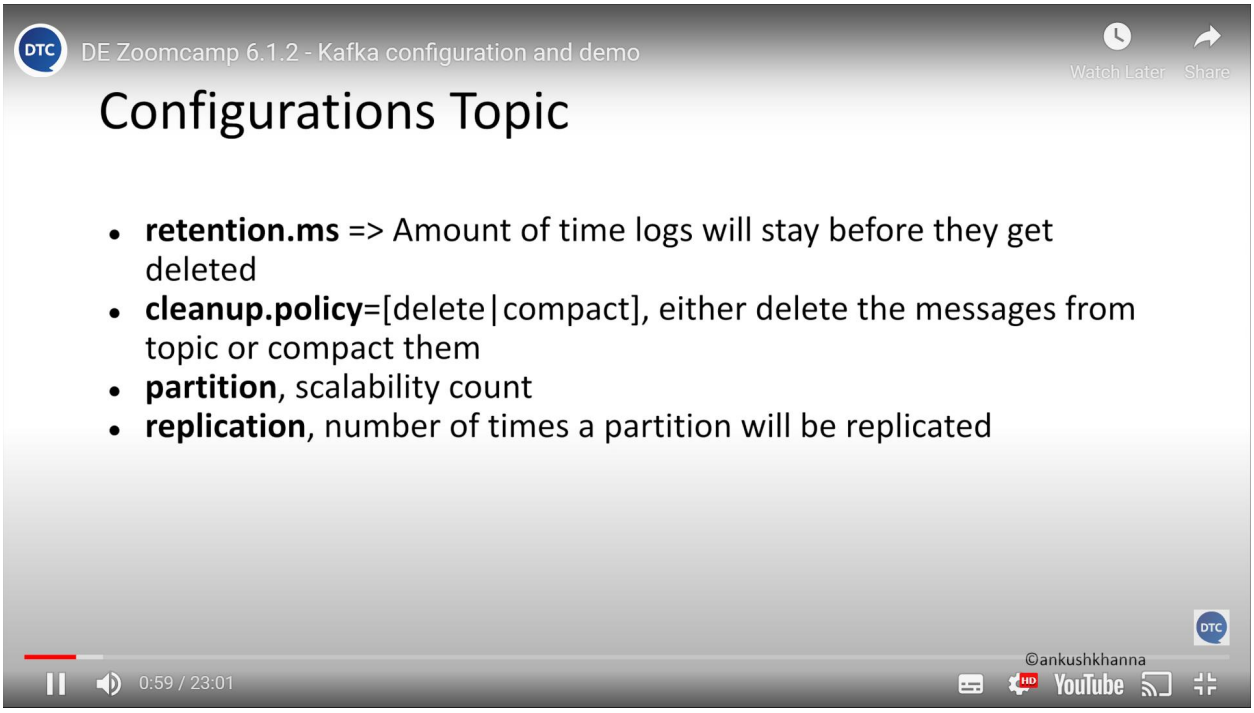
How does kakfa group same key into same partition?

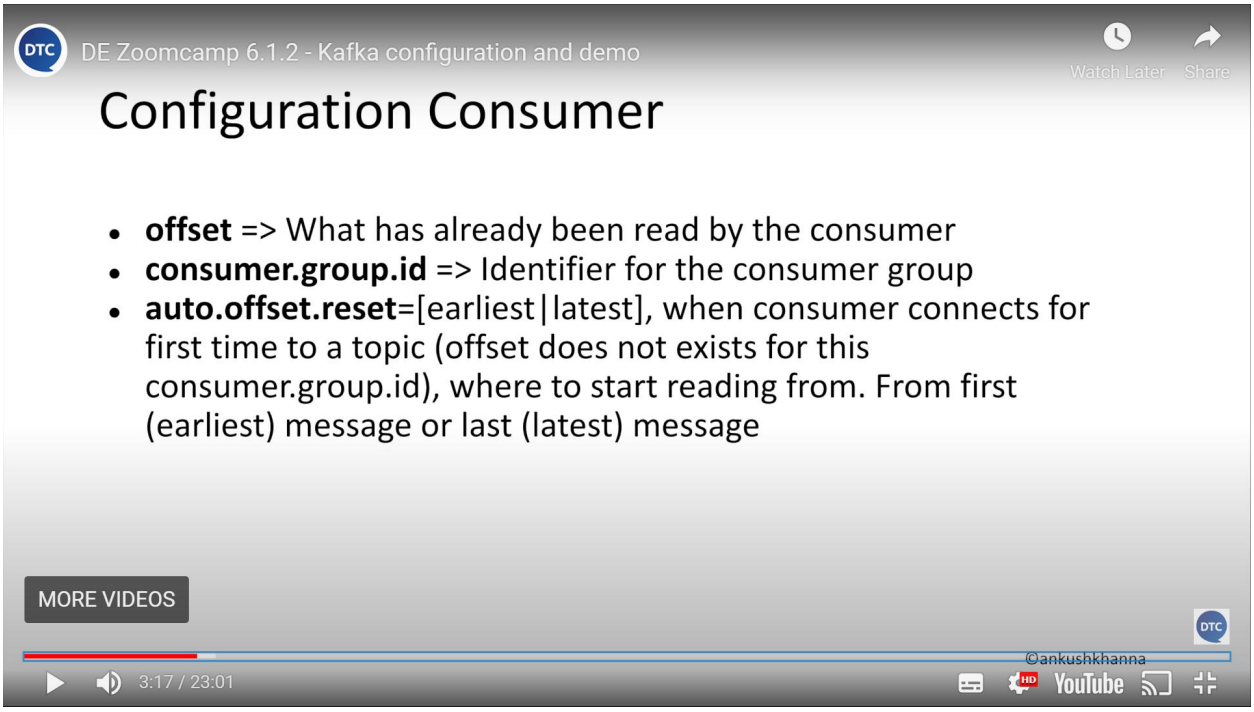
For each key value, hash(key) was computed that decides which partition a key supposed to go with. Here, hash(k1)%4==1, then key goes to partition1.

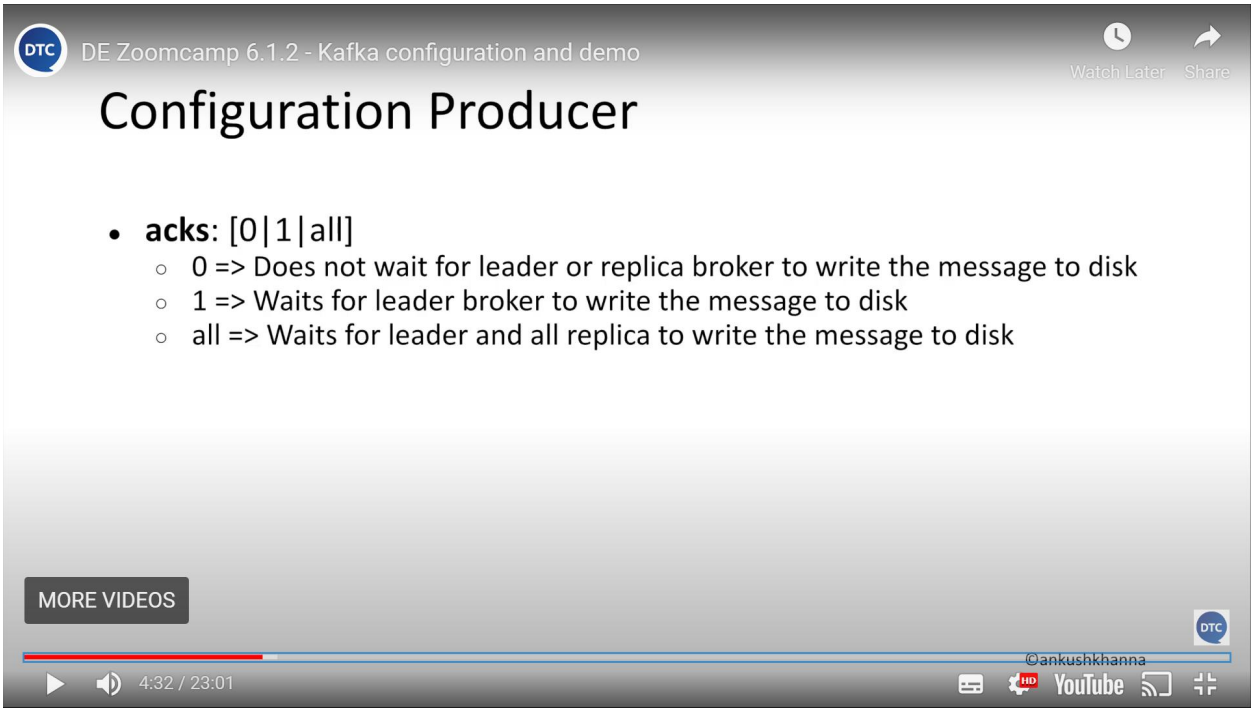


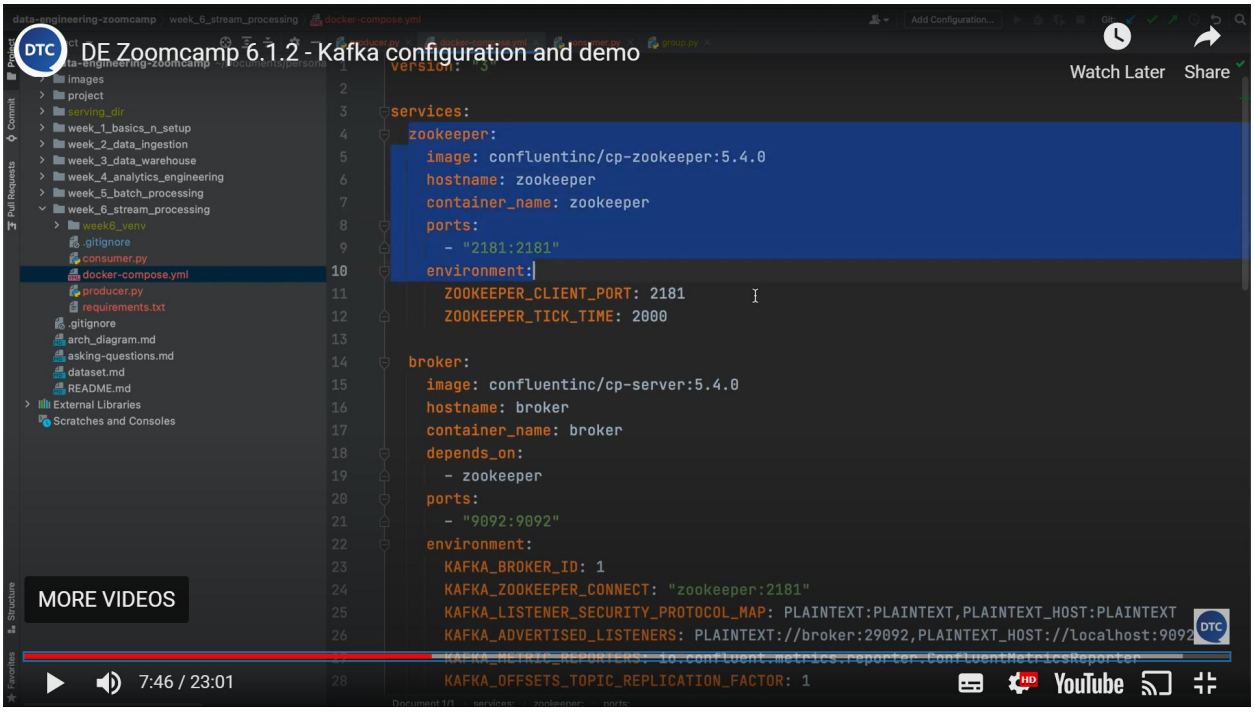


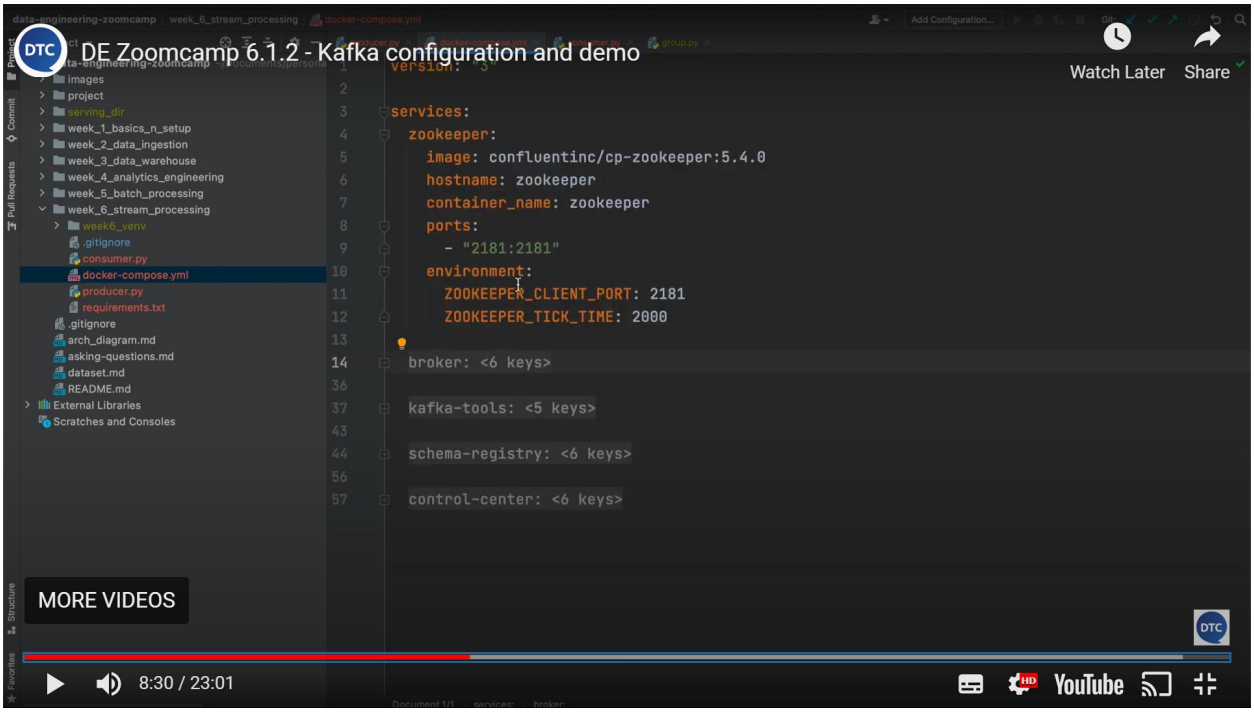
Docker configuration


DE Zoomcamp 6.1.2 - Kafka configuration and demo
Watch Later
Share
Configurations Topic
• retention.ms => Amount of time logs will stay before they get deleted
• cleanup.policy=[delete|compact], either delete the messages from topic or compact them
• partition, scalability count
• replication, number of times a partition will be replicated
©ankushkhanna
0:59 / 23:01
YouTube


DE Zoomcamp 6.1.2 - Kafka configuration and demo
Watch Later
Share
Configuration Consumer
• offset => What has already been read by the consumer
• consumer.group.id => Identifier for the consumer group
• auto.offset.reset=[earliest|latest], when consumer connects for first time to a topic (offset does not exists for this consumer.group.id), where to start reading from. From first (earliest) message or last (latest) message
MORE VIDEOS
©ankushkhanna
3:17 / 23:01
YouTube


DE Zoomcamp 6.1.2 - Kafka configuration and demo
Watch Later
Share
Configuration Producer
acks: [0|1|all]
0 => Does not wait for leader or replica broker to write the message to disk
1 => Waits for leader broker to write the message to disk
all => Waits for leader and all replica to write the message to disk
MORE VIDEOS
©ankushkhanna
4:32 / 23:01
YouTube


DE Zoomcamp 6.1.2 - Kafka configuration and demo
Watch Later
Share
MORE VIDEOS
7:46 / 23:01
YouTube



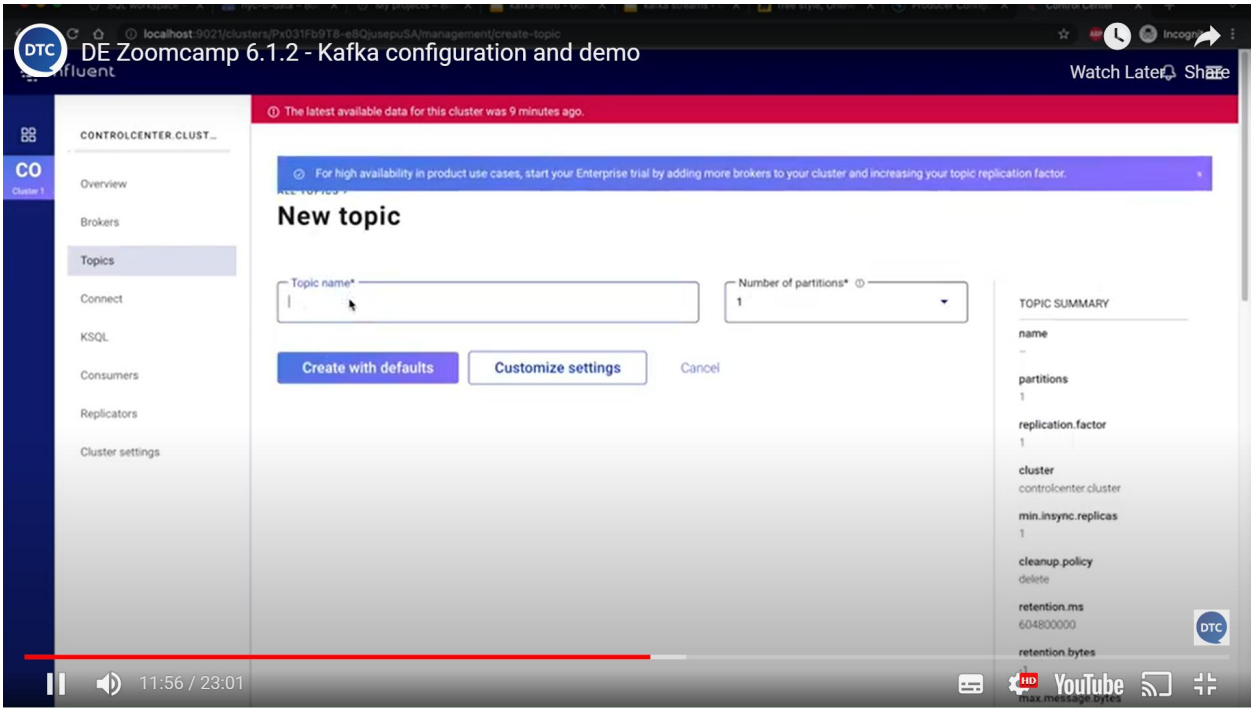
$docker-compose up // start docker container for kafka

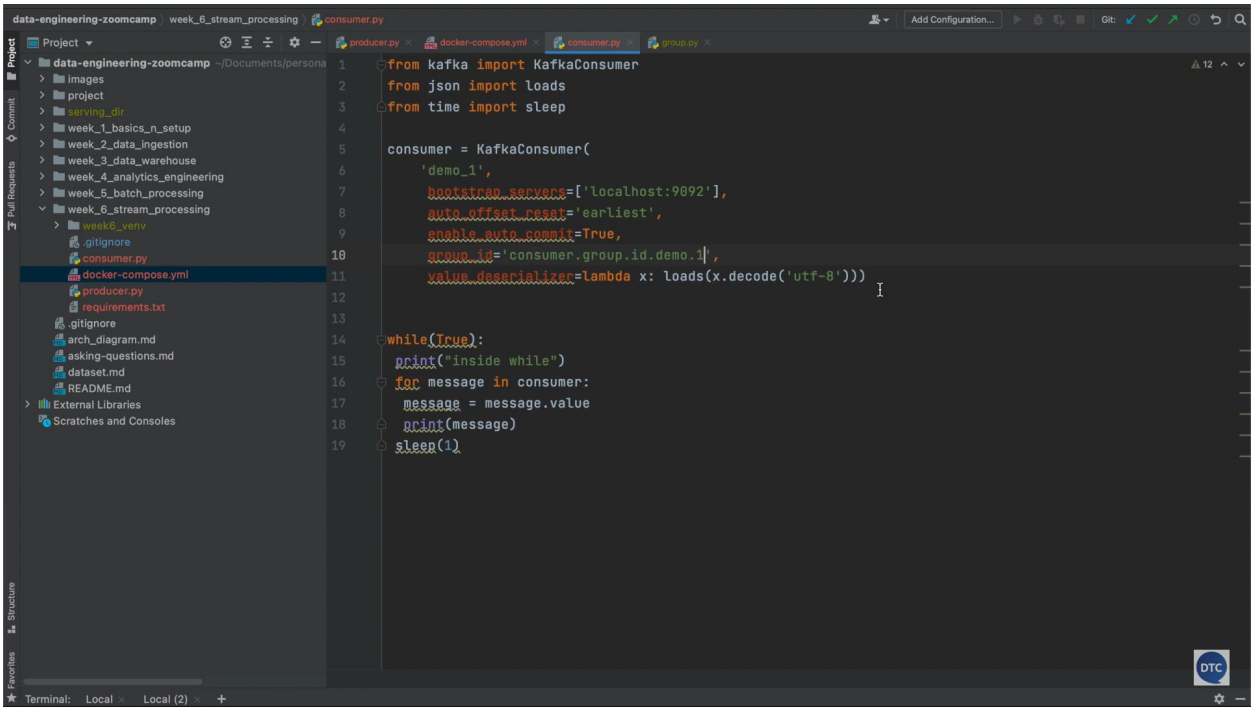
$docker ps // to check docker has started w/t error.

UI for docker configuration and management as part of enterprises edition, this can be done in command line too.

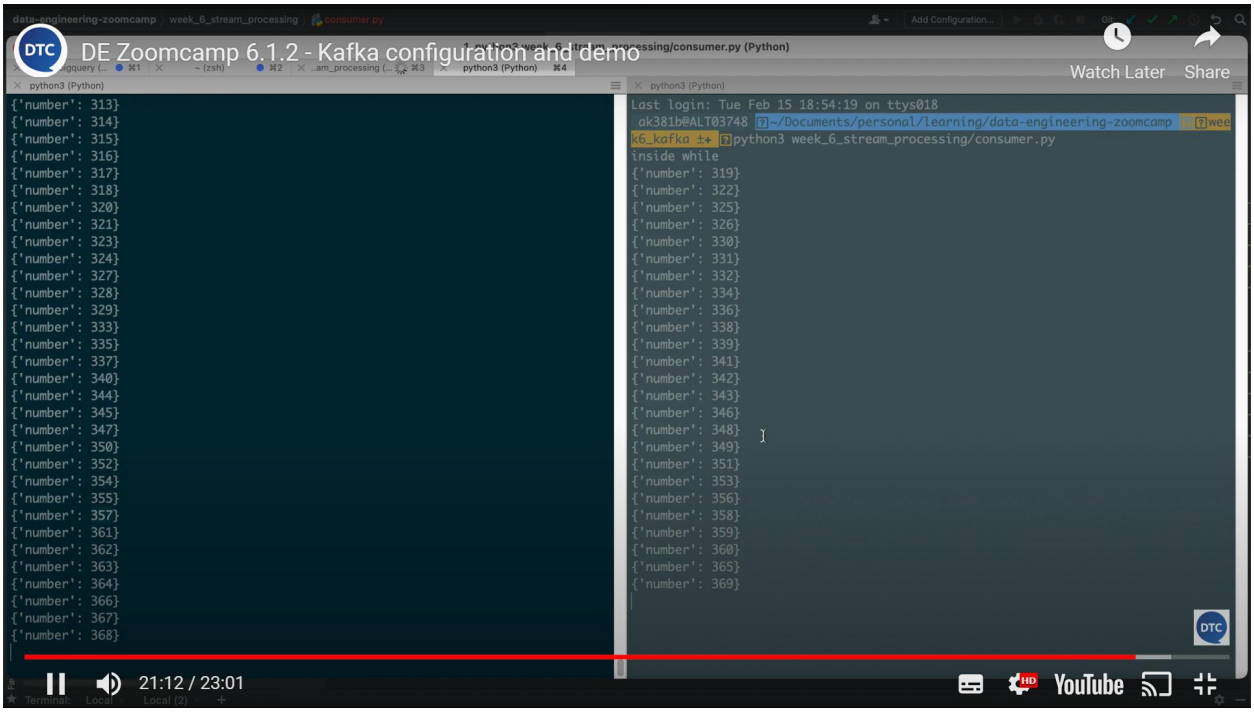
http://localhost:9021/cluster

Create a new topic using “topics” column.



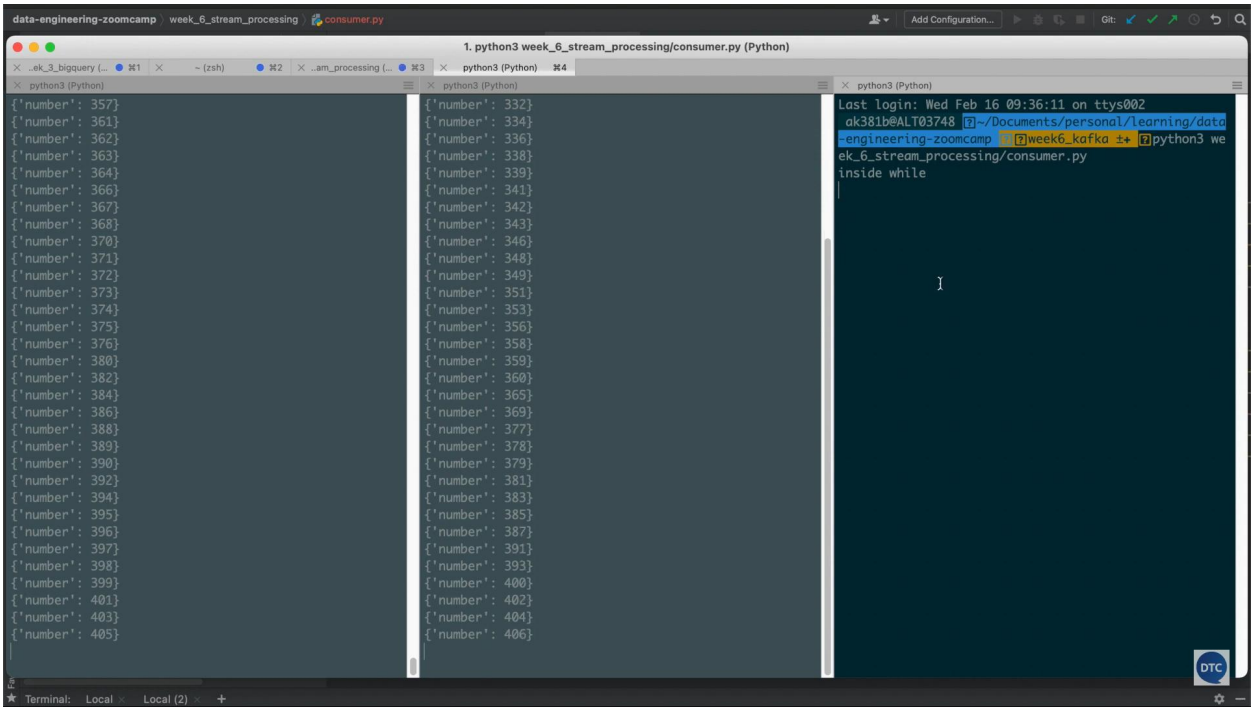


Topic partitioned 2 and 2 consumers running in parallel:

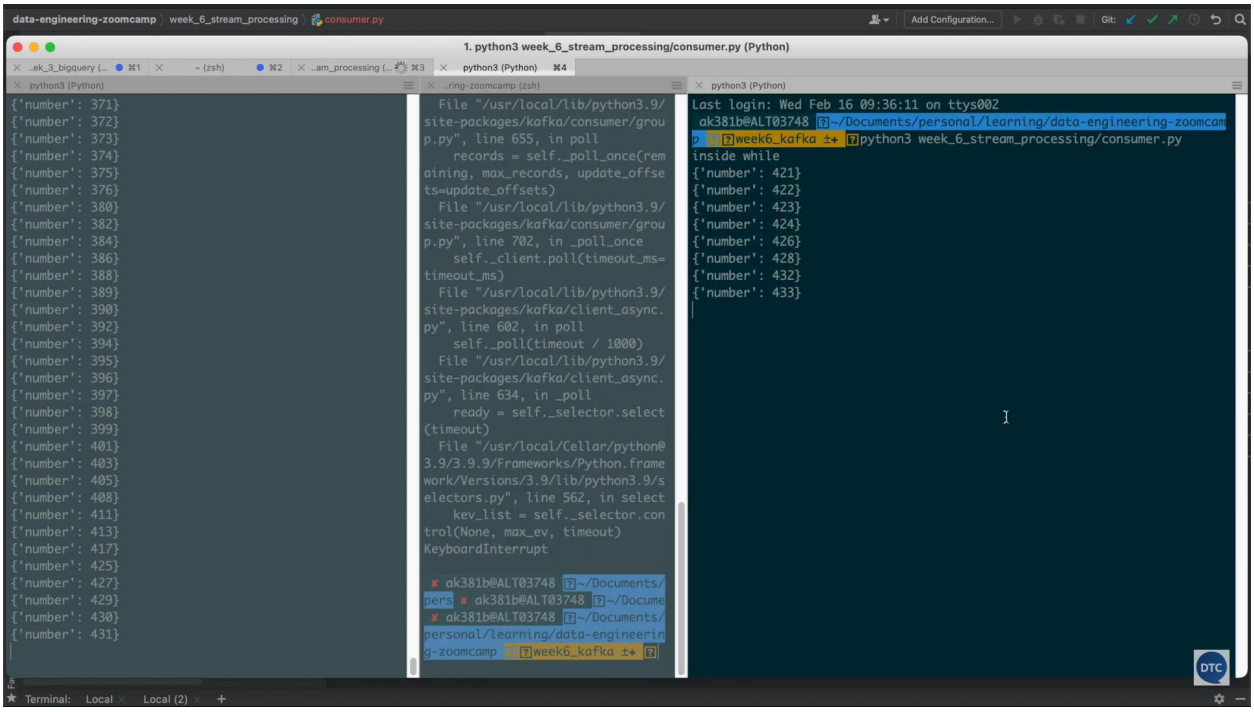


Consumers > number of partition that consumer is idle and not processing any data.

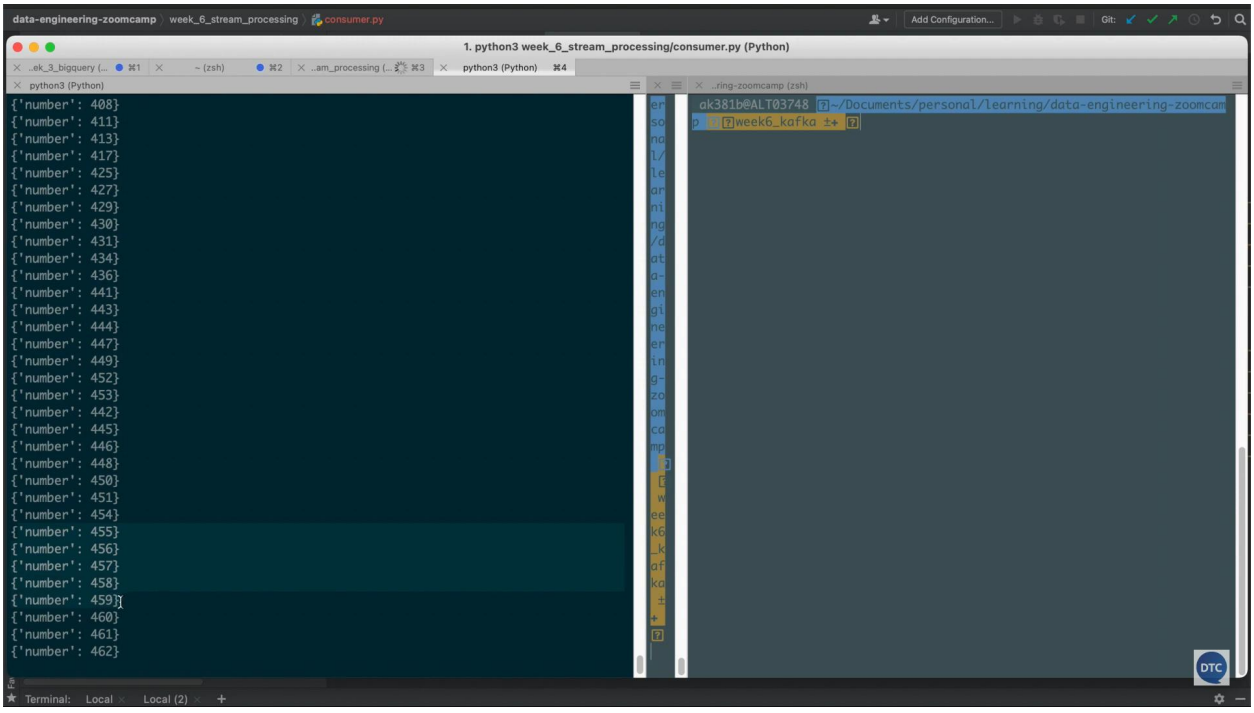
Topic demo_1 with 2 partitions and 3 consumers running in parallel. 2 consumers were consuming data and third consumer is idle.



Killed consumer2 and 3rd consumer got registered into kafka and start receiving data from kafka:

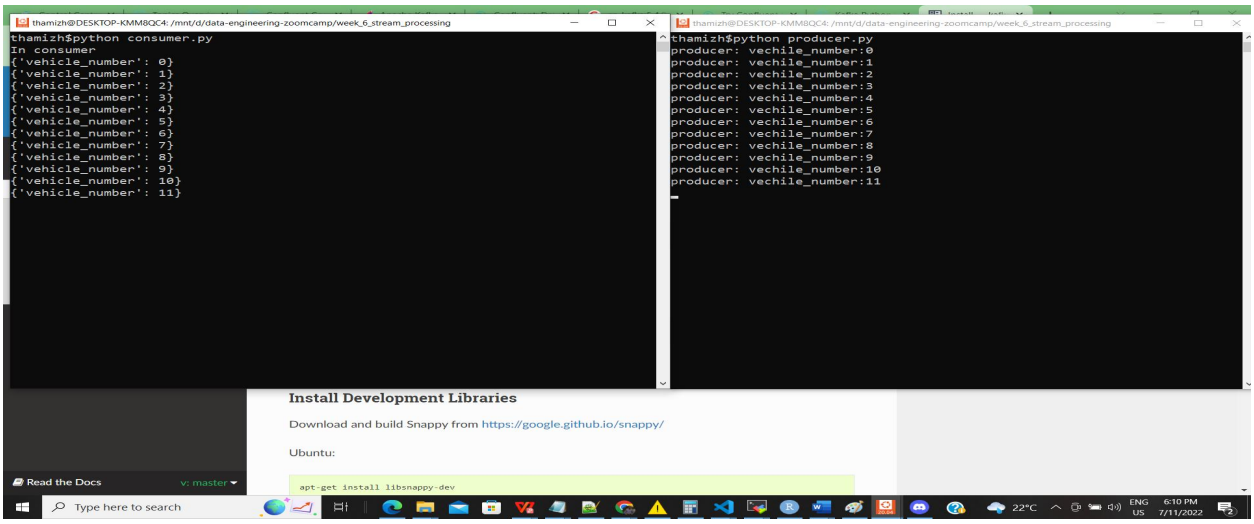


Killed consumer2 & consumer3, now consumer1 start receiving all the messages sequentially:

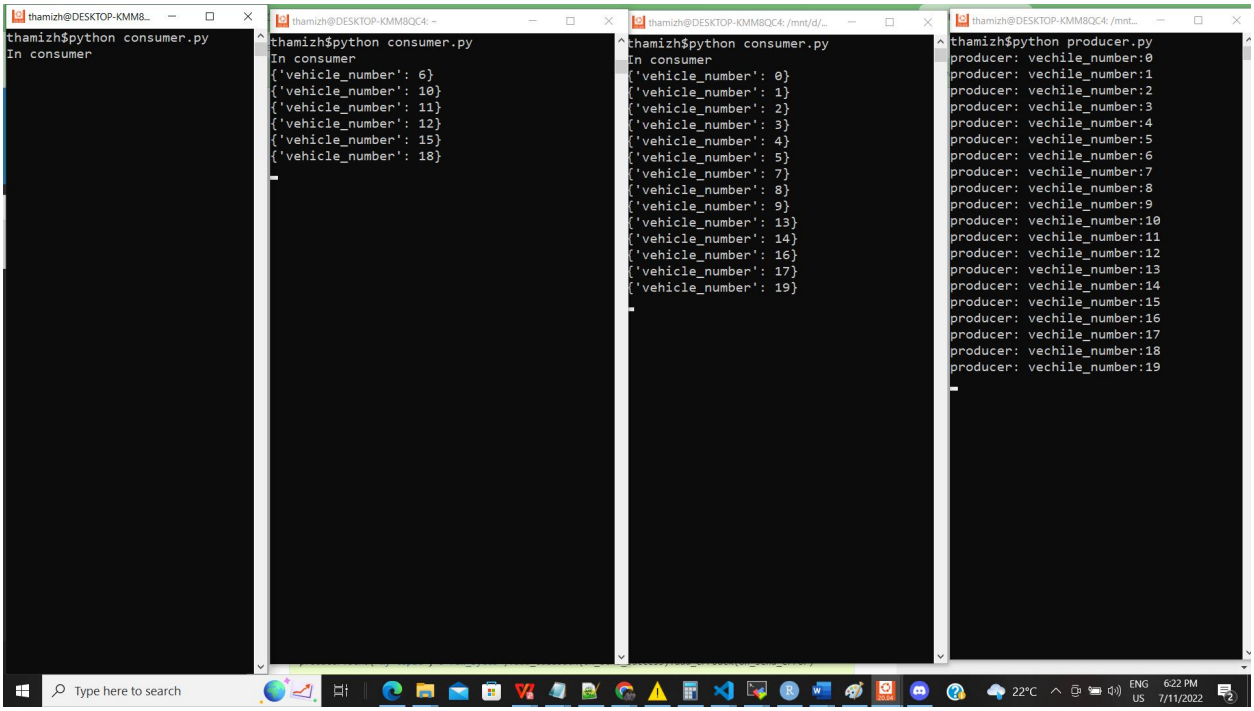


Experiment:

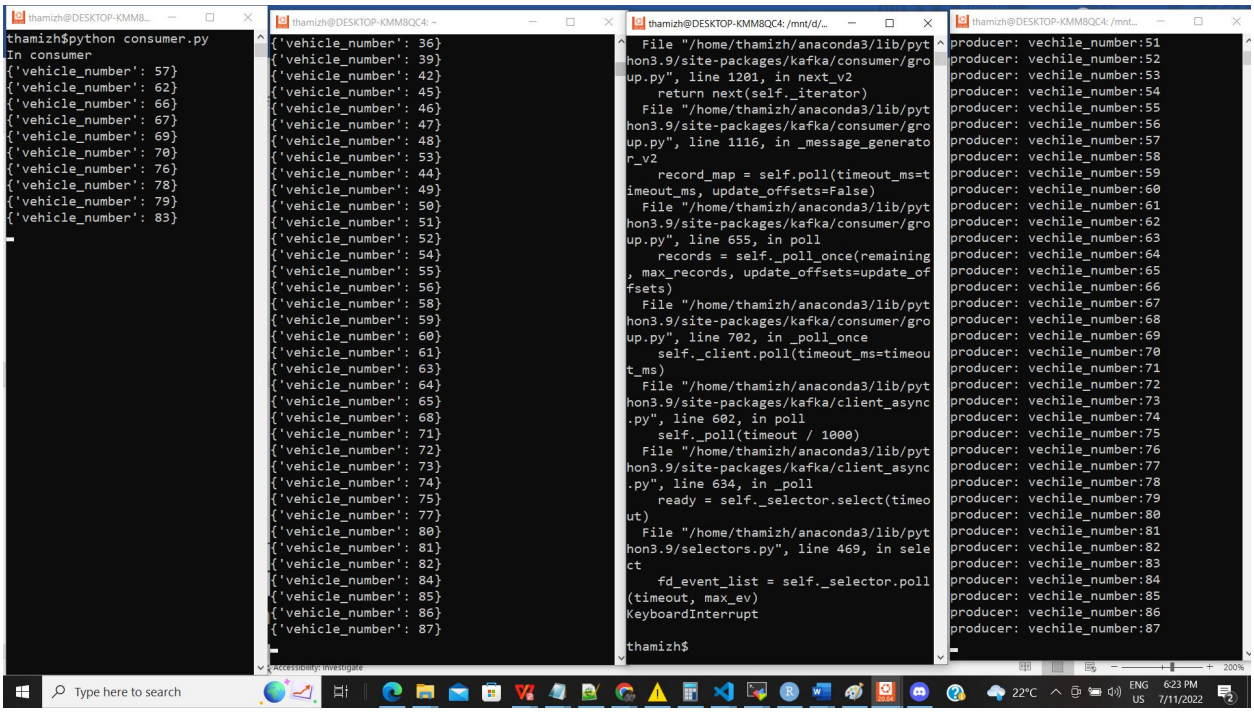
Kafka 1 producer and 1 consumer



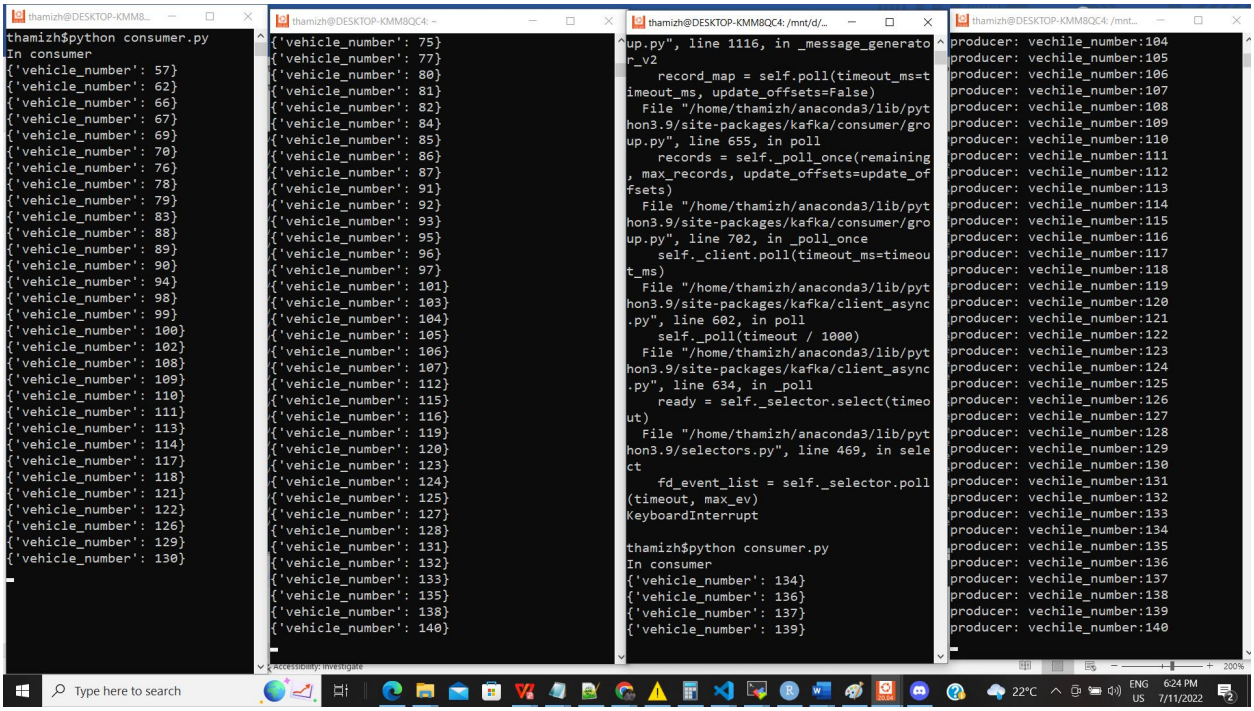
1 producer, 2 partitions, 3 Consumers configured. Consumer2 & 3 started processing kafka topics. Consumer1 is idle.



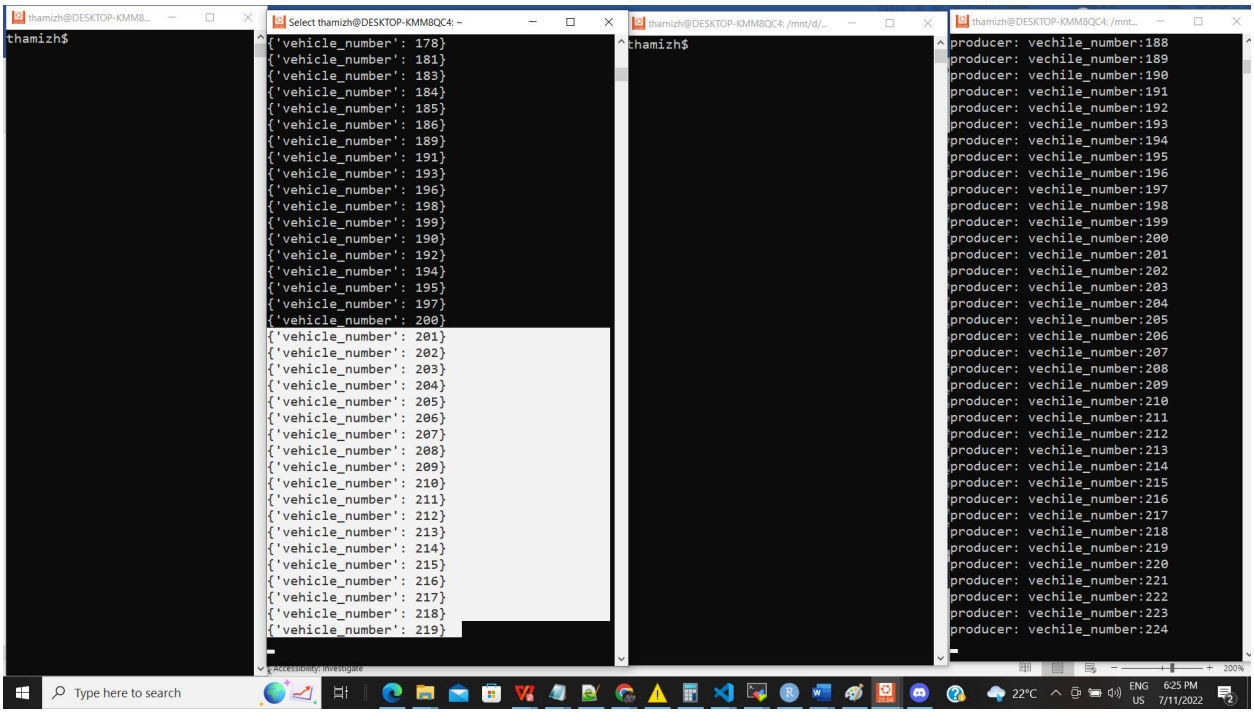
Consumer3 was killed



<u>1 producer, 2 partitions, 3 Consumers configured. Initially consumer 2 & 3 started receiving kafka messages, after consumer 3 got killed, consumer1 automatically started processing kafka topics:</u>

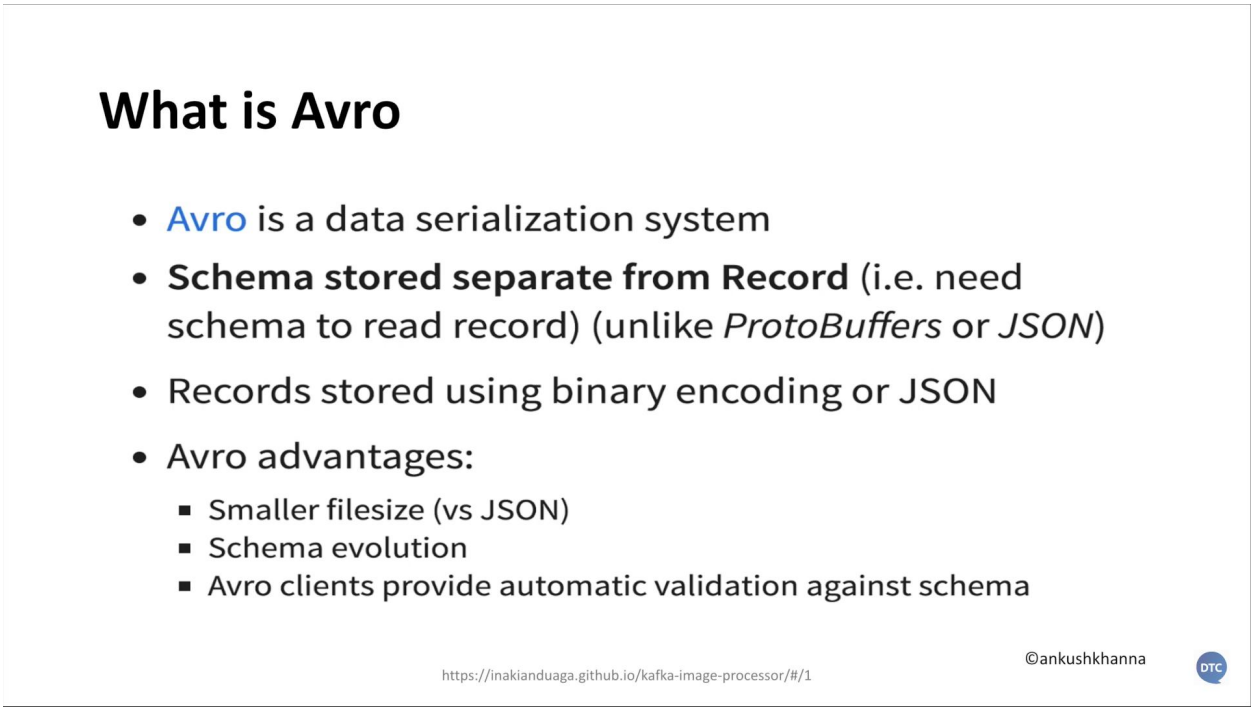


2 kafka borker partition, 3 consumers confugured, only one consumer running and receiving messages

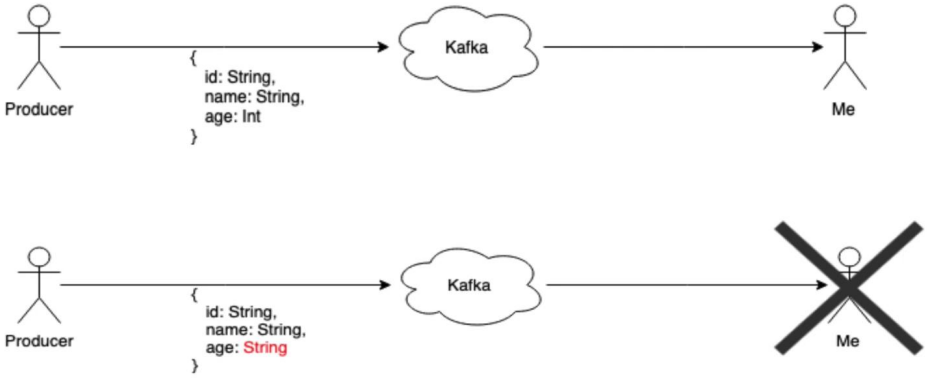


Avro Schema Registry:

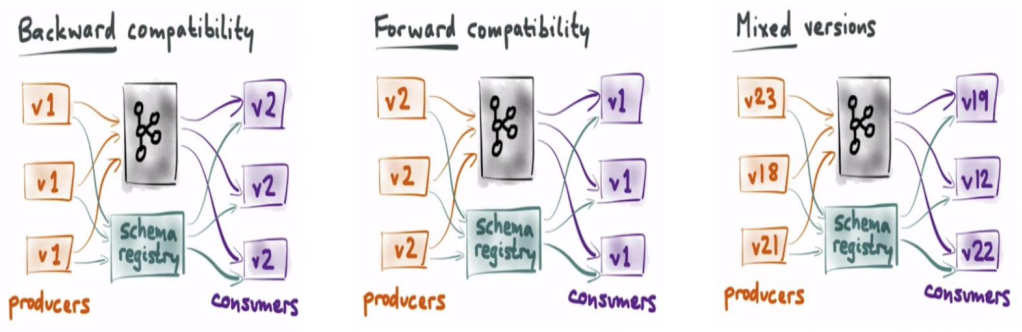
PHPAppln -> Kafka-> Java consumer



Why Schema Compatibility?

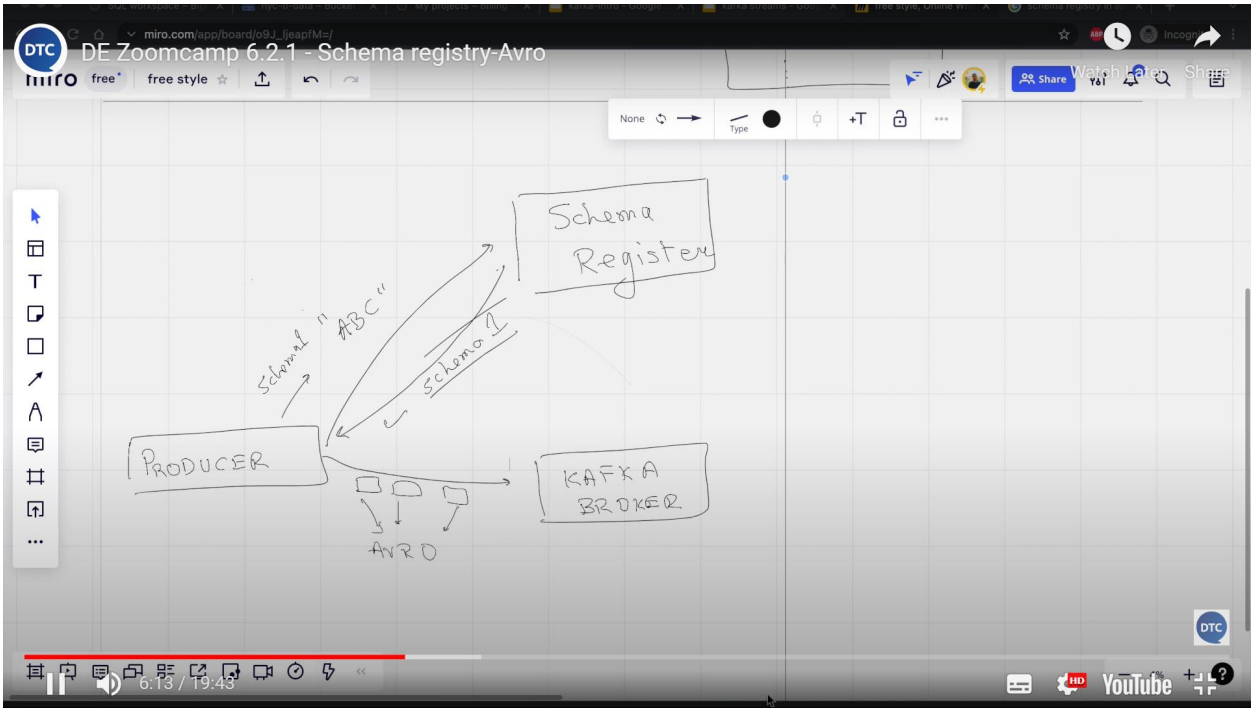


Avro schema evolution

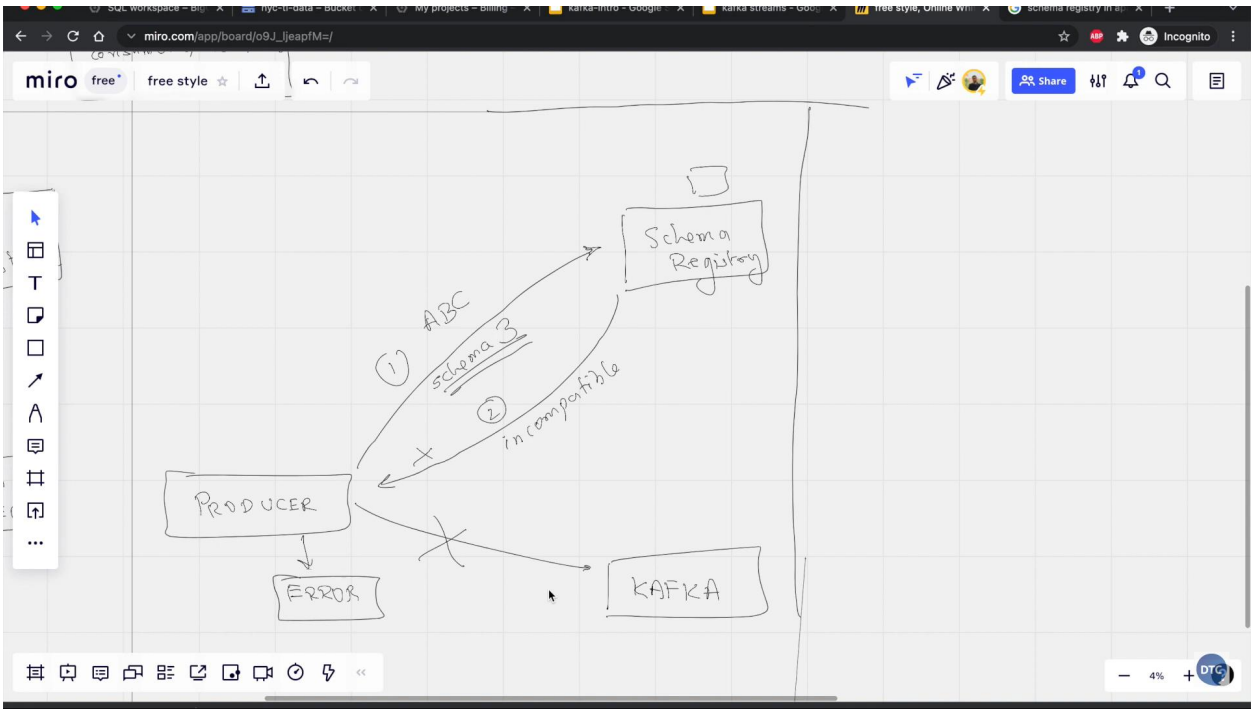


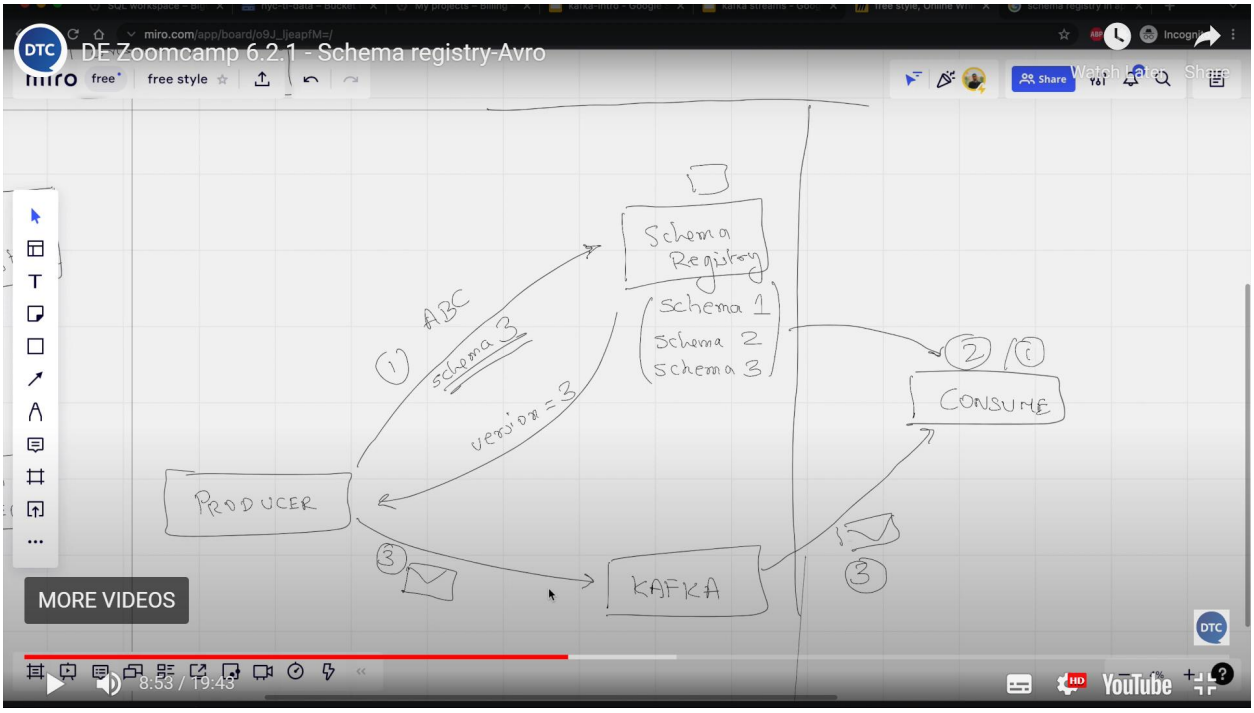
https://inakianduaga.github.io/kafka-image-processor/#/1

Avro serialization: schema is compatible

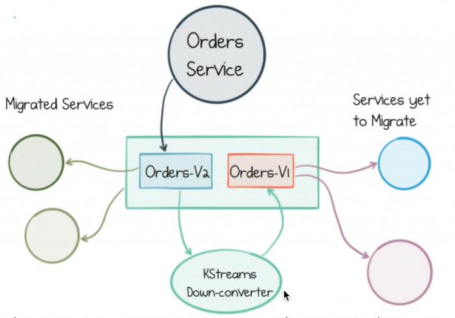


Avro serialization: schema is not compatible





What if Schema cannot be compatible?

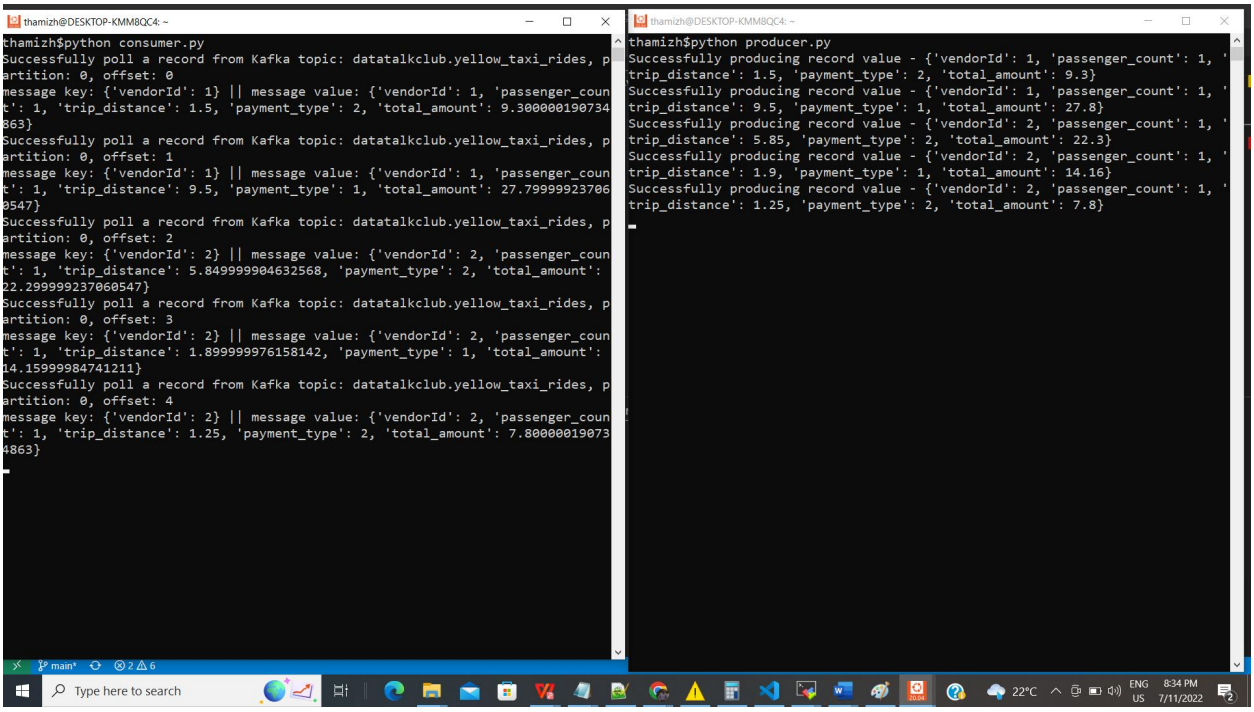


The same data coexists in two topics, with different schemas, so there is a window during which services can upgrade.

https://www.confluent.io/blog/apache-kafka-for-service-architectures/

©ankushkhanna

Kafka Producer consumer with schema validation:



Changed taxi payment as string from int