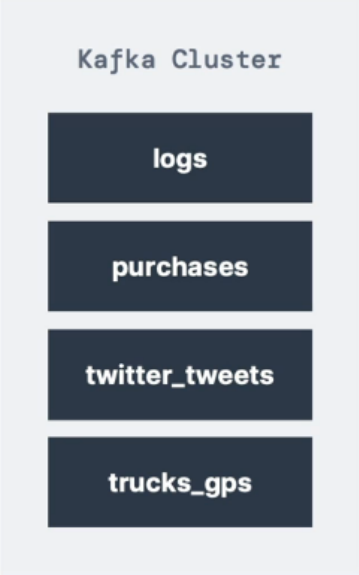
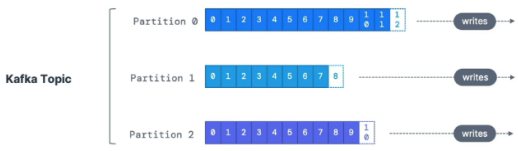
Kafka Theory Part 1

1 Topics, Partitions, Offsets

- Kafka Topic: a particular stream of data
  - Lika a table in DB without constraints



  - Identified by name
  - Any kind of message format
  - sequence of messages is called data stream
  - Cannot query Topic
  - Immutable
- Partitions and Offsets
  - Topics are split into partitions
  - Messages in partitions are ordered



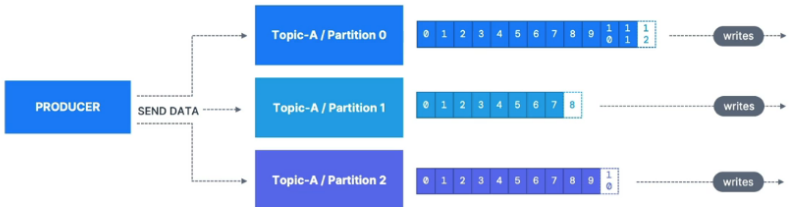
  - Each message within a partition gets an inc id called offset
  - Offset only have a meaning for a specific partition
  - Example: truck_gps - GPS pos
    - 10 partitions

- Truck send msg every 20 sec
- Location service & notification service consumer

2 Producers and Message Keys

Producer

- Producers write data to topics



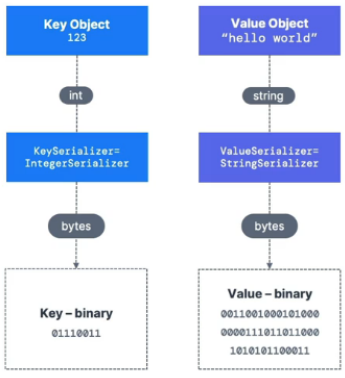
- Producer know to which partition to write to
- Producer auto recover if broker failures

Message keys

- Producers can choose to send a key with msg
- if key=null, round robin
- if key !=null all message will go to same partition

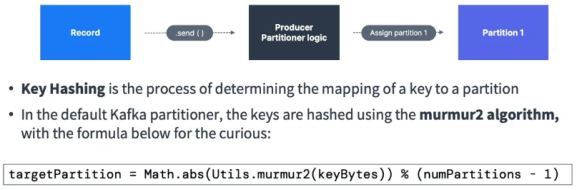
Message

- Message Anatomy: <Key-binary, Value-binary>, CompressionType, Headers, Partition+offset, TimeStamp
- Kafka Message Serializer: kafka accepts only bytes as inputs and sends bytes. Message Seriazlization means transforming obj to bytes



- Common serializer: string (json), int, float, avro, protobuf

- Kafka Key hashing

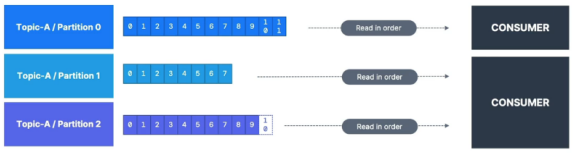


4 Consumer Groups and Consumer Offsets

- All the consumers in app read data as consumer groups
- if consumers > partition, some consumers will be inactive
- Multiple consumer group per Topic is accepted
- Kafka stores the offsets at which a convumer group has been reading
- Offserts committed are in Kafka topic named __consumer_offsets
- When a consumer consumers it should periodically commit the offsets
  - If the consumer dies, it can pick up from where it left
- Delivery semantics for consumers
  - Default: auto commit offsets
  - Manual Commit
    - Atleast once: offsets commit after processing. if goes wrong again read. So need idempotent(processing again wont impact)
    - At most once: offsets are commited as soon as messages are received. if processing goes wrong some msg will be lost
    - Exactly Once: kafka => kafka workflows: use Transactional API

3 Consumers and Deserialization

- Consumers read data from a topic - pull model



- Consumer know which partition to read from
- In case of broker failure consumer knows to recover
- Data is read in order within each partition

- Consumer Deserializer - deserialize from binary to object