Kafka Bootcamp

2017, December, 14 9:04 AM

Message Keys are what determines order

- Not required, producer can provide a NULL key which round robin the messages to partitions

Gwen Shapira

- ETL Lens
  - Kafka is an integration platform not a point to point process
- Scalability
- Publisher/subscriber architecture, storage, processing
- Future?
  - Additional languages (mainly java now)
  - More security (PII etc)
  - Cloud
- How do you know how many topics/partitions
  - Think of topics as tables in a database
    - More flexible
      - Can change schema etc…
  - # of partitions is chosen on scale
- Data should be stored in a raw format, kafka has an immutable datastore
  - History has to be repeatable
  - Audit purposes
- Data has to be encrypted by consumer

Chris Matta

- Why Streaming?
  - All data is even streaming and has been for some time
  - Databases are materialzationms of events, Change Data Capture
  - Capture allows these CDC logs to be centralized
- Apache Kafka
  - Built to handle data flowing into Hadoop
    - Needed a system to act as a shock absorber for the volume of data into HDFS
    - "Immutable append only distributed commit log"
      - Distributed by design
        - Replication
        - Fault Tolerance
        - Partitioning
        - Elastic Scaling
      - Cannot edit messages in queue and cannot delete an individual message (except for compact queue??)
  - Runs on commodity linux host
  - Strict ordering and guarenteed write
    - Persistence
      - Consumers job to acknowledge it read the message then the partition deletes it
  - When you write data into Kafka you write it into memory of the broker
    - Data must be serialized as bytes so you need to know the data when you deserialize it
      - Kafka says this topic offset points to this memory offset in unix
        - Java writes to unix memory log
        - Linux page cache is very important
          - Linux page cache is held in memory for a certain amount of time then flushed to disk
          - When data is read it will need to be promoted to cache as well
            - Can cause instability when consumers are reading slowly
  - While you are writing data consumers can be reading from any point in the log
  - If you provide no key then round robin is used to store values to topics

- If you do provide a key then the key is hashed to determine which partition it is written too
- When you write to Kafka you are writing to the leader then it is mirrored to the other boxes
- In the event of a failure a new leader is chosen from the topics that are in sync
- Kafka will "self heal" and wil ltry to move leaders so that one broker is not a leader for more then one topic
- Order is guarenteed per partition but NOT along partitions
- Processing of data and writing/reading of data is decoupled
- Bootstrap URL is used to introduce the produce to each of the brokers provided in the bootstrap
- "Topics are metadata that wrap partitions"
- Want to plan the # of partitions that you can scale into so that you can horizontally scale partitions
  - If you have 5 servers and 4 partitions it is hard to setup a 5th partitions
  - But if you have 8 partitions and 4 servers it is relatively easy to scale up brokers
- Producers can set a config to when the brokers send a notification that messages are ready
  - Ack 0
    - No ackknowledgement
    - Fastest way to right to kafka
  - Ack 1
    - Acknowledge that the broker wrote the message to itself
  - Ack all (-1)
    - Min.insync.replica=2
- Every 3 minutes consumer will send a message to consumer offset topic saying where its at in the queue
  - This topic is typically replicated across the cluster
- Consumers
  - Consumers give themselves a name (consumer groups)
  - Multiple consumers can be in the consumer group, and can be given their own partition to read messages
  - Heartbeat for health of each consumer
  - This also gives us the ability to rebalance if a consumer dies
  - If you have more consumers then partitions then a consumer will sit idle
  - Can be written to consume from 1 partition
- Kafka Connect
  - Piece of the Kafka toolbox
  - Noticed lots of boiler print kagka producers and consumers
  - Instead you can download a Kafka connectors and configure it
  - Classified as sources or Sinks
- Kafka Streams
  - 3 paradigms of programming
    - Request processing
    - Batch processing
    - Stream processing
  - Event driven micro service
  - Simple java library
    - Look into creating a simple kafka stream app
  - ** No micro batch, true event processing **
  - Null keys are not supported in Kafka streams
  - Compacted topic will retire records where we have a new record for that key
  - Can read and write from topics
  - Ktable is a compacted topic in Kafka

***** # of partitions is the amount of paralelism****
"One is none and 2 is one"
***** Keys will Determine Partition *****
***** avro is a serialization schema ****

Things to go over:
- Load balancing
- Change Data capture

- Z os linux OS
- Throughput of data
- Serializing data
- Linux page cache and linux heap and cache miss/hit
- Zookeeper
- https://www.confluent.io/resources
- Eclipse neon
- https://github.com/cjmatta?tab=repositories
- STAR schema