auto_offset_earliest

Simon Bäurle, Nicolas Neudeck

Tech-Stack

kafka

KSQL

Task 1

- org.apache.kafka.clients.producer.KafkaProducer
- kotlinx.serialization.json
- Create topic with Conduktor (UI) or broker bash (“events”)

- Read JSON file line by line:
    - Parse Line and extract ID field from event (use as Event ID)
    - Convert Line String to ByteArray and send event to Kafka
    - producer.serializer = ByteArraySerializer

```kotlin
fun produce(topicName:String, value: String){
        val key = json.parse(Data.serializer(), value).id
        val producerRecord = ProducerRecord(topicName, key, value.toByteArray())
        kafkaProducer.send(producerRecord)
}
```

Task 2

- Added ksqlDB and ksqlDB-CLI to setup
- Streams
  - meetup_events_stream
  - meetup_events_stream_de
  - meetup_events_stream_de_munich

```
CREATE STREAM meetup_events_stream_de AS SELECT name,"GROUP"->city
AS city , "GROUP"->country AS country FROM meetup_events_stream
WHERE "GROUP"->country = 'de';
```

```
CREATE STREAM meetup_events_stream_de_munich AS SELECT name, city, country
FROM meetup_events_stream_de WHERE city = 'Munich' OR city = 'München';
```

Latency

- org.apache.kafka.clients.producer.KafkaConsumer
- Kafka automatically adds creation timestamp on event (per topic)
- Two concurrent consumers (using Coroutines)
  - Consumer 1 (“events”)
  - Consumer 2 (“meetup_events_stream_de_munich”)
  - Consumers store event_id and event_timestamp in map
- Compare timestamps for same event_id to get processing time of kafka/ksqlDB

- Result: Processing time of Kafka is very low (<1 ms)
- Potential error: Copied creation timestamp between different queues

Throughput & Kafka Metrics

- Initial Idea: Collect built-in metrics from Kafka Producer/Consumer
  - Multiple, inconsistent results per metric (no reliable data source!)
- Own Measurement in code
  - Measure execution time and event file size
- Built-in metrics from Conduktor
  - Interesting Outcome: Docker on Mac has substantially higher networking overload (way slower than on Linux)

- Results: produced full events.json (~350MB) in ~18 seconds to Kafka (~19,4 MB/s)
- Cf. Kafka Performance Evaluation (better hardware): ~78MB/s

Task 3 - Hands On