Lab4. Kafka Consumer 애플리케이션 개발

1. 새로운 주제(topic) 생성하기
   1)partition 2개의 topic 생성
      -$ ./kafka-topics.sh --create --bootstrap-server {MSK 클러스터 엔드포인트1},{MSK 클러스터 엔드포인트2} --replication-factor {복제본 갯수(브로커 갯수)} --partitions {토픽당 파티션 갯수} --topic {topic 이름}
      -ex)./kafka-topics.sh --create --bootstrap-server b-2.henrymskcluster.jm4797.c4.kafka.ap-northeast-2.amazonaws.com:9092,b-1.henrymskcluster.jm4797.c4.kafka.ap-northeast-2.amazonaws.com:9092 --replication-factor 2 --partitions 2 --topic henry-topic2
      Created topic henry-topic2.

2. Kafka Producer Application 작성하기
   1)Visual Studio Code의 [Explorer]에서 [kafka-client]에 [producer] 디렉토리 생성
   2)생성한 [producer] 디렉토리에 Dockerfile과 producer.py 옮기기
   3)[kafka-client]에 [consumer] 디렉토리 생성
   4)[consumer] 디렉토리에 consumer.py 파일 작성

3. 방금 생성한 Consumer 애플리케이션 테스트를 위해 Producer 애플리케이션 실행
   1)producer.py 코드 일부 수정
      -topic의 값 변경
      -ex)topic = 'henry-topic' --> topic = 'henry-topic2'

   2)먼저 Consumer 애플리케이션 실행
      $ python3 consumer.py
      -이렇게 하면 producer.py가 실행된 것이 아니기 때문에 10초가 대기 후 종료할 것임.

   3)Tabby에서 producer.py 실행
      $ pwd
      /home/ec2-user/kafka-client/producer

      $ python3 producer.py
      sended data : {'num': '0'}
      sended data : {'num': '1'}
      sended data : {'num': '2'}
      sended data : {'num': '3'}
      sended data : {'num': '4'}
      sended data : {'num': '5'}
      sended data : {'num': '6'}
      sended data : {'num': '7'}
      sended data : {'num': '8'}
      sended data : {'num': '9'}
      sended data : {'num': '10'}
      sended data : {'num': '11'}
      sended data : {'num': '12'}
      sended data : {'num': '13'}
      sended data : {'num': '14'}
      sended data : {'num': '15'}
      sended data : {'num': '16'}
      sended data : {'num': '17'}
      sended data : {'num': '18'}
      sended data : {'num': '19'}
      elpased : 60.088377714157104

   4)Visual Studio Code에서 consumer.py 실행
      topic=henry-topic2, partition=1, offset=0, key=None, value={'num': '0'}
      topic=henry-topic2, partition=1, offset=1, key=None, value={'num': '2'}
      topic=henry-topic2, partition=0, offset=0, key=None, value={'num': '1'}
      topic=henry-topic2, partition=1, offset=2, key=None, value={'num': '3'}
      topic=henry-topic2, partition=0, offset=1, key=None, value={'num': '4'}
      topic=henry-topic2, partition=0, offset=2, key=None, value={'num': '5'}
      topic=henry-topic2, partition=1, offset=3, key=None, value={'num': '6'}
      topic=henry-topic2, partition=0, offset=3, key=None, value={'num': '7'}
      topic=henry-topic2, partition=0, offset=4, key=None, value={'num': '8'}
      topic=henry-topic2, partition=1, offset=4, key=None, value={'num': '9'}
      topic=henry-topic2, partition=1, offset=5, key=None, value={'num': '10'}
      topic=henry-topic2, partition=1, offset=6, key=None, value={'num': '11'}
      topic=henry-topic2, partition=1, offset=7, key=None, value={'num': '12'}
      topic=henry-topic2, partition=1, offset=8, key=None, value={'num': '13'}
      topic=henry-topic2, partition=0, offset=5, key=None, value={'num': '14'}
      topic=henry-topic2, partition=1, offset=9, key=None, value={'num': '15'}
      topic=henry-topic2, partition=1, offset=10, key=None, value={'num': '16'}
      topic=henry-topic2, partition=0, offset=6, key=None, value={'num': '17'}
      topic=henry-topic2, partition=1, offset=11, key=None, value={'num': '18'}
      topic=henry-topic2, partition=1, offset=12, key=None, value={'num': '19'}

   5)발견할 수 있는 정보
      -partition 을 번갈아서 메시지 처리함.
      -같은 partition 에서는 offset이 순차적이지만, partition 끼리를 순차적이지 않다.
      -다시 코드를 둘 다 실행하면 각 partition의 마지막 offset부터 다시 읽어온다.
      -group_id를 변경하면 처음부터 다시 시작한다.
      -consumer.py에서 auto_offset_reset의 값을 earliest로 변경해서 실행해 볼 것
      -group_id를 새로운 그룹이름으로 실행해 볼 것.