Fundamentals of Data Engineering

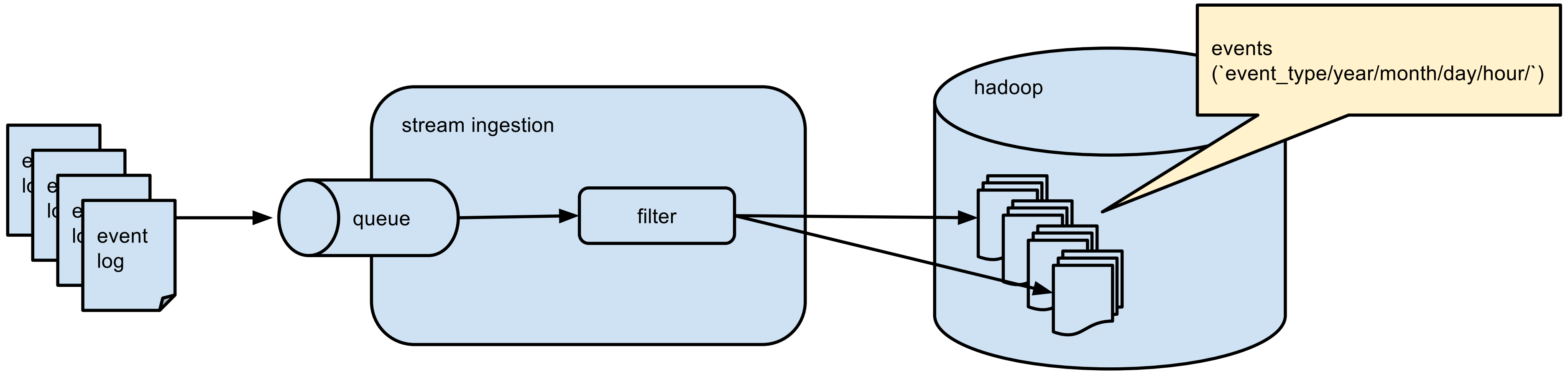
Week 07 - sync session

datascience@berkeley

While we’re getting started

- Review your Assignment 06
- Get ready to share

Due Friday (PR)



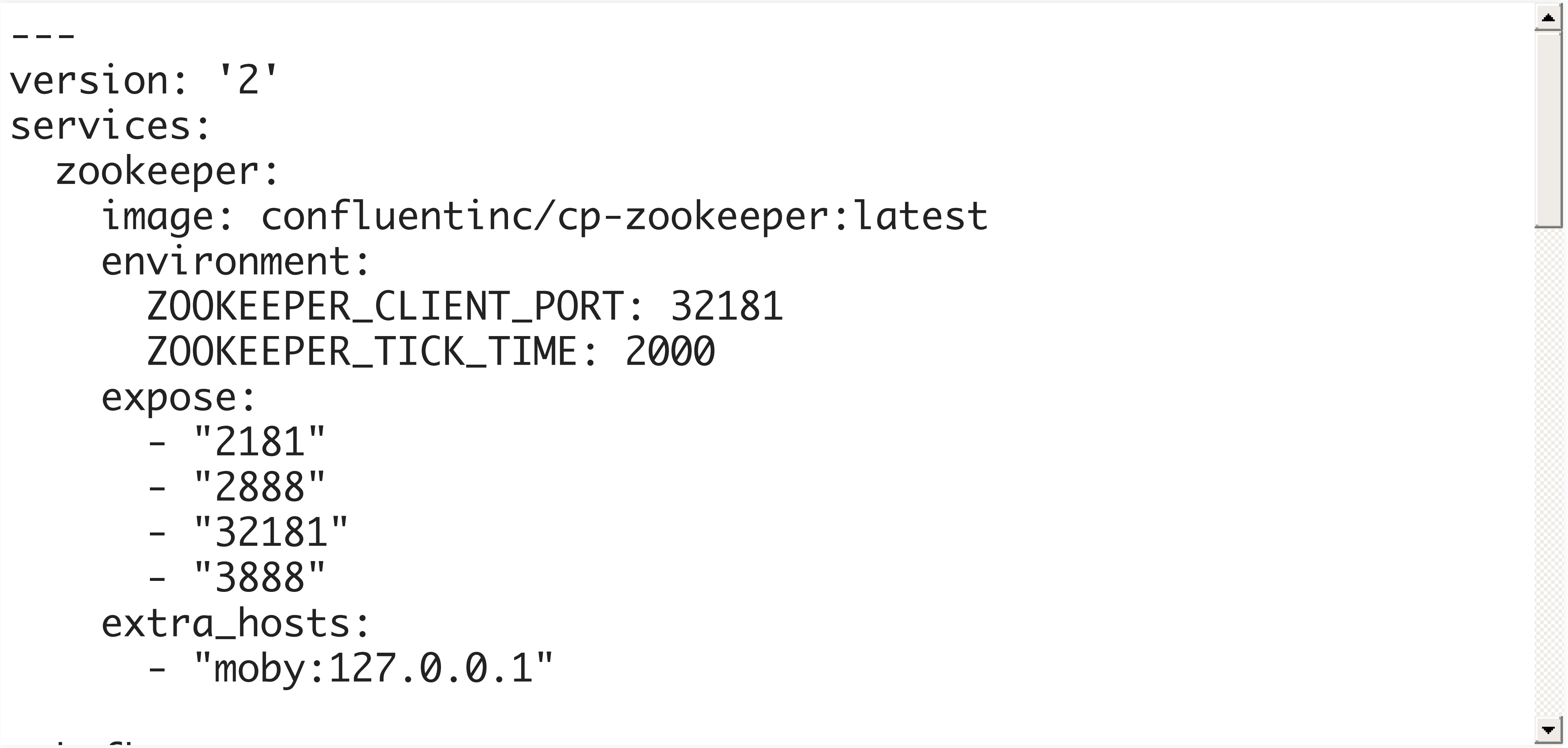
events (`event_type/year/month/day/hour/`)

Spark Stack with Kafka

Setup

```
mkdir ~/w205/spark-with-kafka
cd ~/w205/spark-with-kafka
cp ~/w205/course-content/07-Sourcing-Data/docker-cor
```

docker-compose.yml



```
---
version: '2'
services:
  zookeeper:
    image: confluentinc/cp-zookeeper:latest
    environment:
      ZOOKEEPER_CLIENT_PORT: 32181
      ZOOKEEPER_TICK_TIME: 2000
    expose:
      - "2181"
      - "2888"
      - "32181"
      - "3888"
    extra_hosts:
      - "moby:127.0.0.1"
```

up

```
docker-compose up -d

docker-compose logs -f kafka
```

create a topic

```
docker-compose exec kafka \
  kafka-topics \
    --create \
    --topic foo \
    --partitions 1 \
    --replication-factor 1 \
    --if-not-exists \
    --zookeeper zookeeper:32181
```

Should show

```
Created topic "foo".
```

Check the topic

```
docker-compose exec kafka \
  kafka-topics \
  --describe \
  --topic foo \
  --zookeeper zookeeper:32181
```

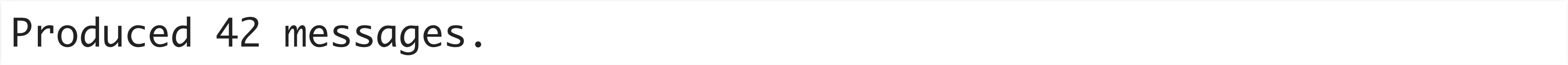
Should show

```
Topic:foo   PartitionCount:1    ReplicationFactor:1 Configs:
Topic: foo  Partition: 0    Leader: 1    Replicas: 1  Isr: 1
```

Publish some stuff to kafka

```
docker-compose exec kafka \
  bash -c "seq 42 | kafka-console-producer \
    --request-required-acks 1 \
    --broker-list kafka:29092 \
    --topic foo && echo 'Produced 42 messages.'"
```

Should show



```
Produced 42 messages.
```

Run spark using the `spark` container

```
docker-compose exec spark pyspark
```

read stuff from kafka

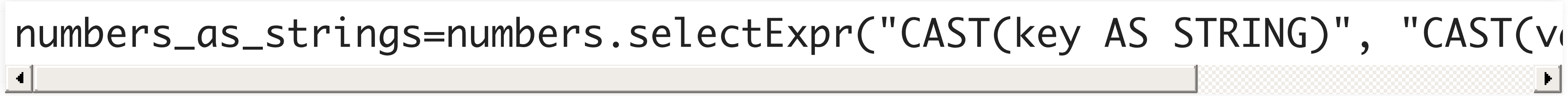
At the pyspark prompt,

```
numbers = spark \
  .read \
  .format("kafka") \
  .option("kafka.bootstrap.servers", "kafka:29092") \
  .option("subscribe","foo") \
  .option("startingOffsets", "earliest") \
  .option("endingOffsets", "latest") \
  .load()
```

See the schema

```
numbers.printSchema()
```

Cast it as strings



```
numbers_as_strings=numbers.selectExpr("CAST(key AS STRING)", "CAST(v
```

Take a look

```
numbers_as_strings.show()
```

```
numbers_as_strings.printSchema()
```

```
numbers_as_strings.count()
```

down

```
docker-compose down
```

Spark stack with Kafka with “real” messages

docker-compose.yml file

- same
- still in your `~/w205/spark-with-kafka`

Pull data

```
curl -L -o github-example-large.json https://goo.gl/Hr6erG
```

Spin up the cluster & check

```
docker-compose up -d
```

```
docker-compose logs -f kafka
```

create a topic

```
docker-compose exec kafka \
  kafka-topics \
    --create \
      --topic foo \
      --partitions 1 \
      --replication-factor 1 \
      --if-not-exists \
      --zookeeper zookeeper:32181
```

Should see something like

```
Created topic "foo".
```

Check the topic

```
docker-compose exec kafka \
  kafka-topics \
    --describe \
    --topic foo \
    --zookeeper zookeeper:32181
```

Should see something like

```
Topic:foo   PartitionCount:1    ReplicationFactor:1 Configs:
Topic: foo  Partition: 0    Leader: 1    Replicas: 1  Isr: 1
```

Publish some stuff to kafka

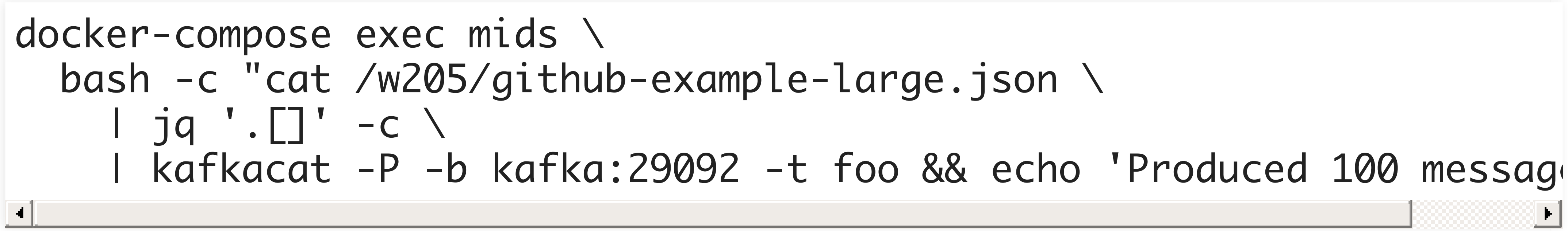
Check out our messages

```
docker-compose exec mids bash -c "cat /w205/github-example-large.jso
docker-compose exec mids bash -c "cat /w205/github-example-large.jso
```

Individual messages

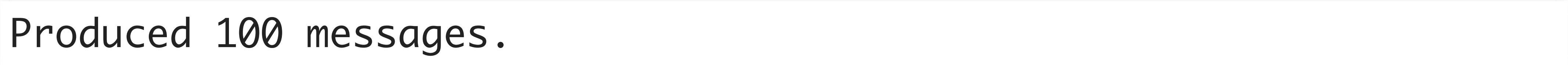
```
docker-compose exec mids bash -c "cat /w205/github-example-large.json
```

Publish some test messages to that topic with the kafka console producer



```
docker-compose exec mids \
  bash -c "cat /w205/github-example-large.json \
    | jq '.[]' -c \
    | kafkacat -P -b kafka:29092 -t foo && echo 'Produced 100 messag
```

Should see something like



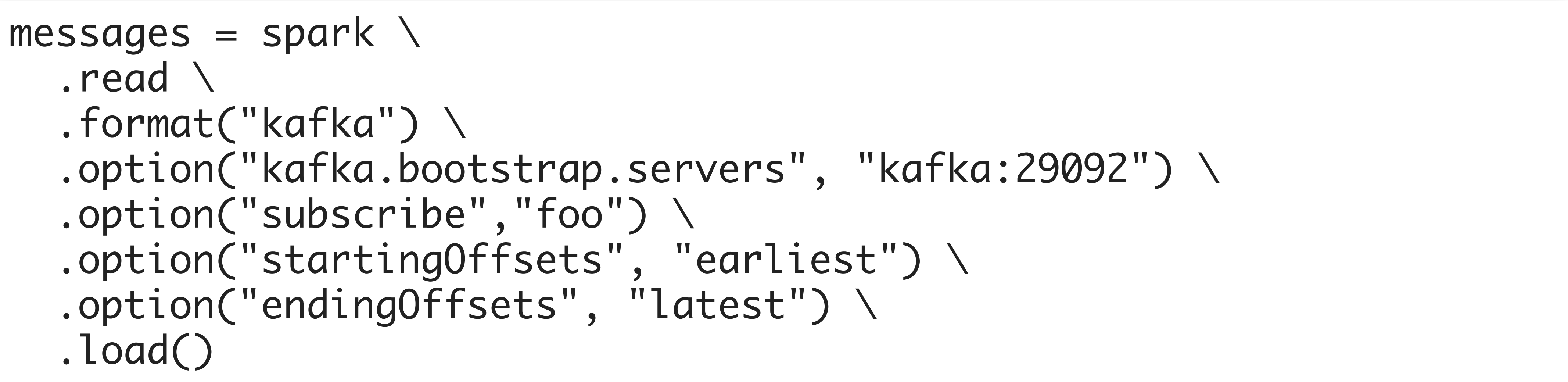
```
Produced 100 messages.
```

Run spark using the `spark` container

```
docker-compose exec spark pyspark
```

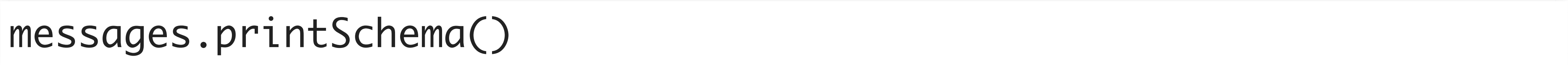
read stuff from kafka

At the pyspark prompt,



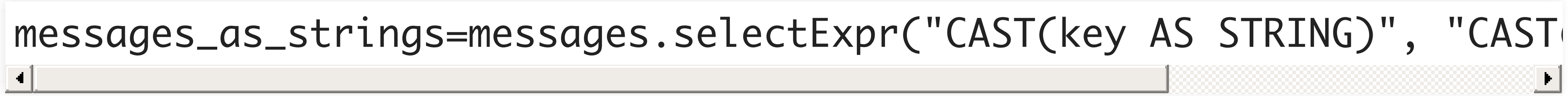
```
messages = spark \
  .read \
  .format("kafka") \
  .option("kafka.bootstrap.servers", "kafka:29092") \
  .option("subscribe","foo") \
  .option("startingOffsets", "earliest") \
  .option("endingOffsets", "latest") \
  .load()
```

See the schema



```
messages.printSchema()
```

Cast as strings



```
messages_as_strings=messages.selectExpr("CAST(key AS STRING)", "CAST(
```

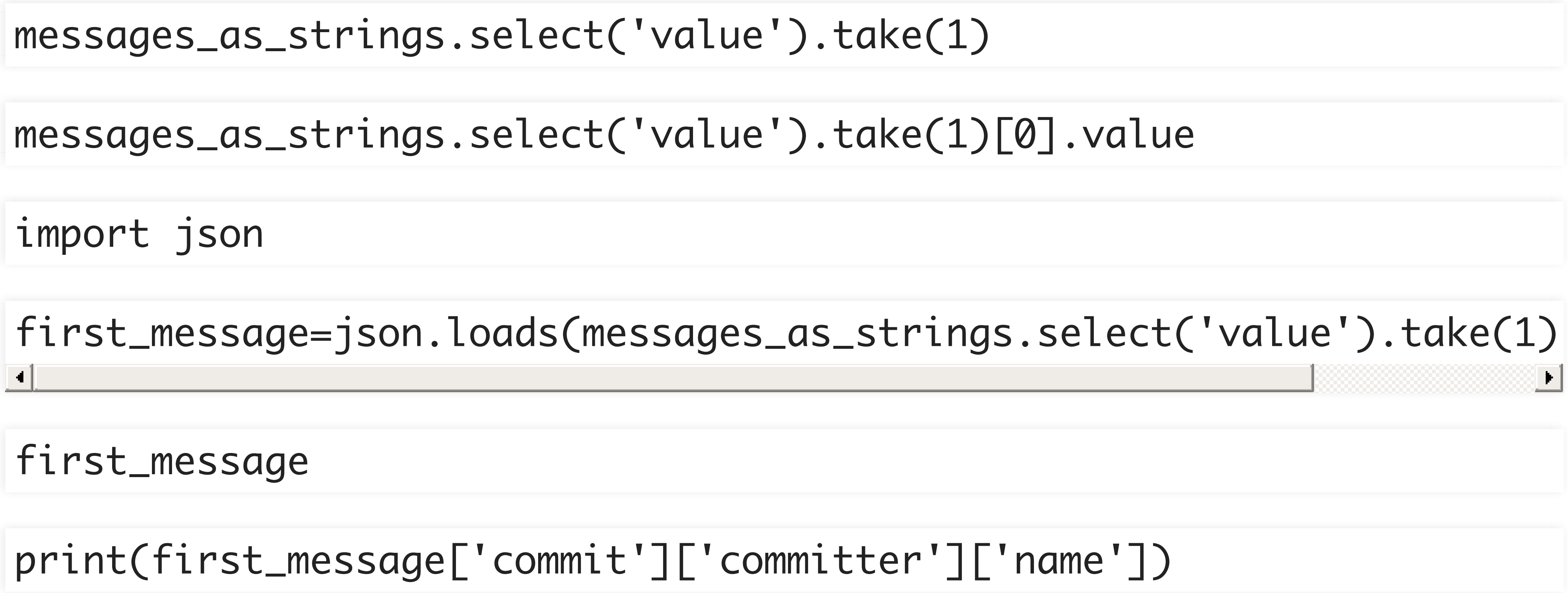
Take a look

```
messages_as_strings.show()
```

```
messages_as_strings.printSchema()
```

```
messages_as_strings.count()
```

Unrolling json



```
messages_as_strings.select('value').take(1)
```

```
messages_as_strings.select('value').take(1)[0].value
```

```
import json
```

```
first_message=json.loads(messages_as_strings.select('value').take(1)
```

```
first_message
```

```
print(first_message['commit']['committer']['name'])
```

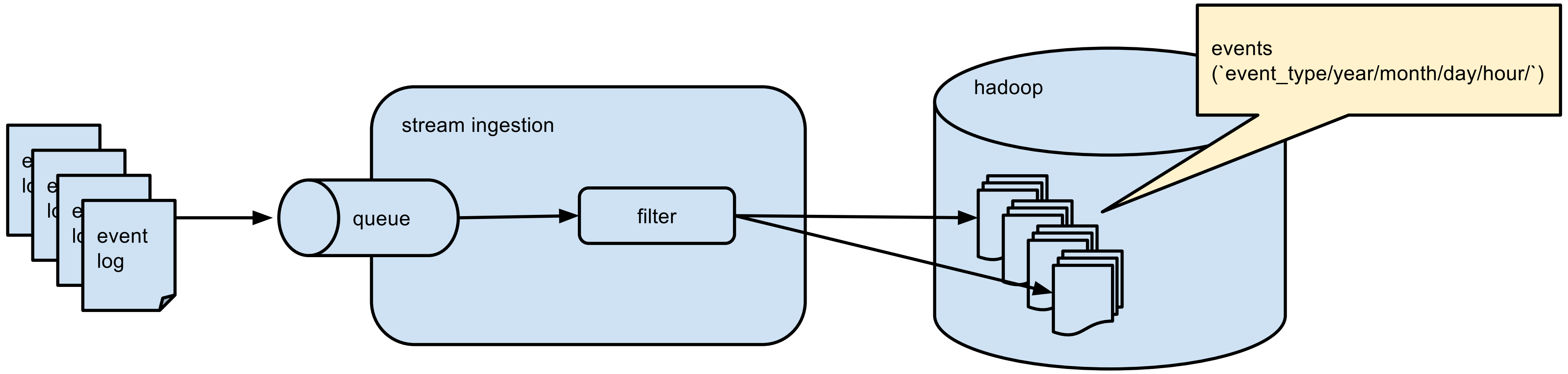
Down

```
docker-compose down
```

Assignment 07

- Step through this process using the Project 2 data
- What you turn in:
- In your `/assignment-07-<user-name>` repo:
    - your `docker-compose.yml`
    - once you’ve run the example on your terminal
        - Run `history > <user-name>-history.txt`
        - Save the relevant portion of your history as `<user-name>-annotations.md`
        - Annotate the file with explanations of what you were doing at each point.

Summary


event log
queue
stream ingestion
filter
hadoop
events
(`event_type/year/month/day/hour/`)

Berkeley SCHOOL OF INFORMATION