Fundamentals of Data Engineering

Week 07 - sync session

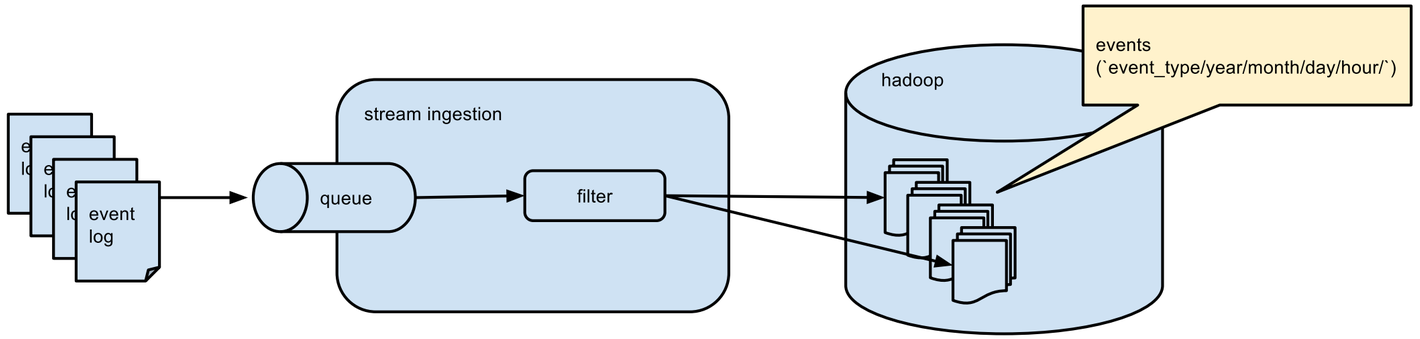
datascience@berkeley

While we're getting started

- Mid-Course Survey

Please get course-eval links from slack

Project Review


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

Spark Stack with Kafka

Setup

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

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"

  kafka:
    image: confluentinc/cp-kafka:latest
    depends_on:
```

Spin up the cluster

```
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(va
```

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/Y4MD58
```

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 real data to kafka

Check out our messages

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

Individual messages

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

Publish some test messages to that topic with kafkacat

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

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()
```

See the messages

```
messages.show()
```

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'])
```

Breakout

- Change around some of the fields to print different aspects of the commit

Down

```
docker-compose down
```

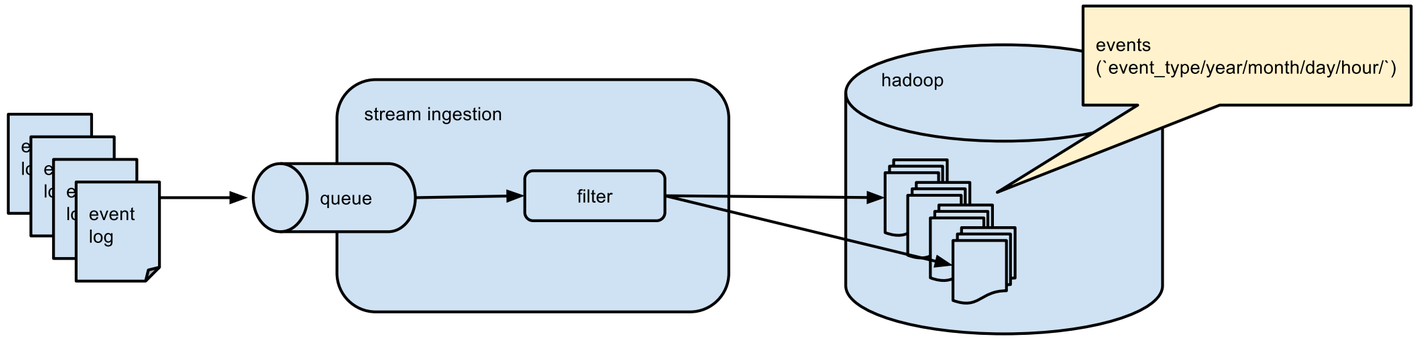
Project 2

- Use Project 2 data
- Step through this process (use spark to read from kafka)
- What you turn in:
- In your `/project-2-<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.

Week 7 Videos:

- Sourcing Data
- Context for Project 2 : The idea is where do you get the data that flows into our pipeline? You wouldn’t usually get it from a file.
- Big Question: What do I need to do to data coming in to get it into Kafka? change the API client code? change app server code?

Summary


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

Berkeley SCHOOL OF INFORMATION