Fundamentals of Data Engineering

Week 05 - sync session

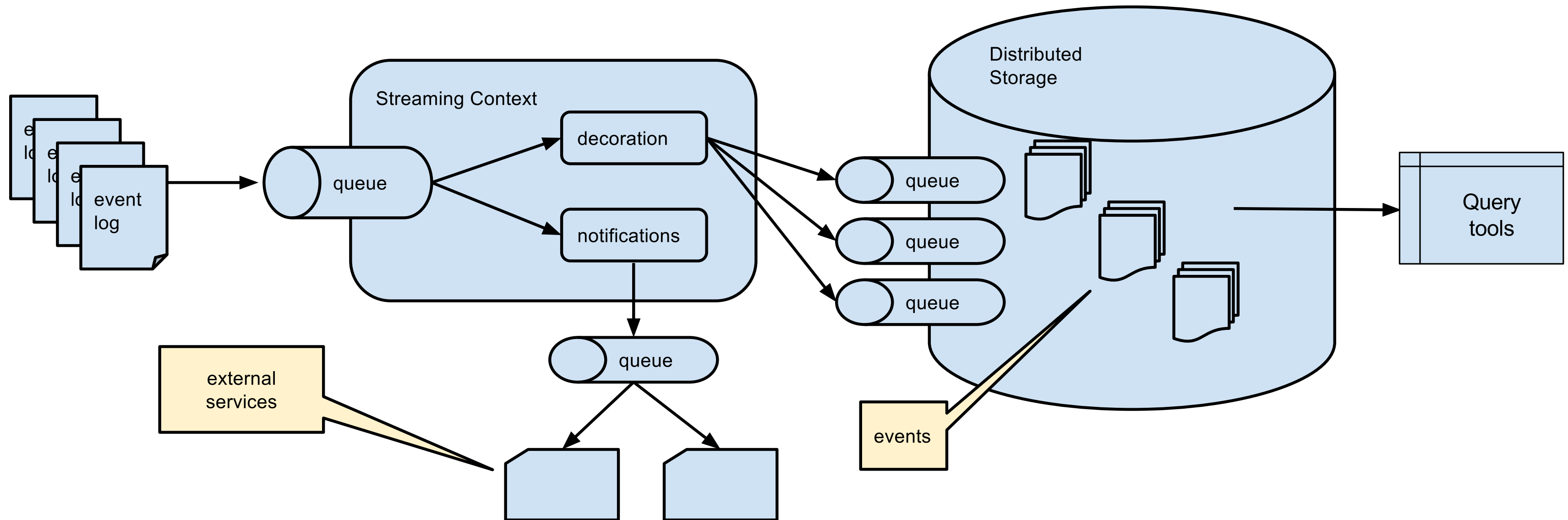# Assignment 04 Breakout

Where are we?

Today

- Assignment 04 wraps up the Query Project
- Creating basic `docker-compose` clusters
- NoSQL store (Redis) with `docker-compose`
- Redis to track state

Class 6

- Start "Tracking User Activity" Project (spans Assignments 6-8)

Where are we in the pipeline?

Docker

Docker

Let's play with Redis

spin it up

```
docker run redis
```

```
docker run -d redis
```

```
docker run -d --name redis redis
```

```
docker run -d --name redis -p 6379:6379 redis
```

Docker compose

- helps manage options easily
- manage multiple containers at once

Update your course content repo in w205

```
cd ~/w205/course-content
git pull --all
```

Docker compose .yml file

```
mkdir ~/w205/redis-standalone
cd ~/w205/redis-standalone
cp ../course-content/05-Storing-Data-II/example-0-docker-compose.yml
```

i.e.,

```
---
version: '2'
services:
  redis:
    image: redis
    expose:
      - "6379"
    ports:
      - "6379:6379"
```

Spinup

Start up the cluster

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

Check stuff

```
docker-compose ps
```

Peek at the logs

```
docker-compose logs redis
```

datascience@berkeley

Should see log output ending in

```
Ready to accept connections
```

Run stuff

```
ipython
```

Try out redis

```
import redis
r = redis.Redis(host='localhost', port='6379')
r.keys()
exit
```

Tear down your stack

```
docker-compose down
```

Verify

```
docker-compose ps
```

Clusters with docker-compose

Setup

Create a workspace for this other example

```
mkdir ~/w205/redis-cluster
cd ~/w205/redis-cluster
cp ../course-content/05-Storing-Data-II/example-1-docker-compose.yml
```

Save the following to
`docker-compose.yml` in that directory

```
---
version: '2'
services:
  redis:
    image: redis:latest
    expose:
      - "6379"

  mids:
    image: midsw205/base:latest
    stdin_open: true
    tty: true
```

Spinup

Start up the cluster

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

Check stuff

```
docker-compose ps
```

Should see

```
          Name                    Command               State      P
----------------------------------------------------------------------
redisexample_midsbase_1    /bin/bash                     Up        88
redisexample_redis_1       docker-entrypoint.sh redis ...   Up        63
```

Peek at the logs

```
docker-compose logs redis
```

Should see log output ending in

```
Ready to accept connections
```

Run stuff

Connect to the mids container

```
docker-compose exec mids bash
```

At the prompt, run

```
ipython
```

Try out redis

```
import redis
r = redis.Redis(host='redis', port='6379')
r.keys()
exit
```

Exit that container

```
exit
```

Tear down your stack

```
docker-compose down
```

Verify

```
docker-compose ps
```

Jupyter Notebooks

Change the `docker-compose.yml` file

```
---
version: '2'
services:
  redis:
    image: redis:latest
    expose:
      - "6379"

  mids:
    image: midsw205/base:latest
    stdin_open: true
    tty: true
    expose:
      - "8888"
    ports:
      - "8888:8888"
```

Save that and bring it up

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

Start up a notebook

```
docker-compose exec mids jupyter notebook --no-browser --port 8888 --
```

Copy token... should look something like

```
open http://0.0.0.0:8888/?token=<your token>
```

Open a browser

```
http://0.0.0.0:8888
```

Paste token

datascience@berkeley

Drop the cluster when you're done

```
docker-compose down
```

Wrapping Up

Assignment 5

Assignment 5: Bonus& More bonus

- extra challenge
- not about extra credit

Week 5 Videos

- Hadoop
- Some basics. We won't use Hadoop until later in the project
- Virtualization
- The notion of using docker-compose to bring up clusters of services. This is the core of Project 2.

Extras

Automate notebook startup

Just for fun,

```
---
version: '2'
services:
  redis:
    image: redis:latest
    expose:
      - "6379"

  mids:
    image: midsw205/base:latest
    stdin_open: true
    tty: true
    expose:
      - "8888"
    ports:
      - "8888:8888"
    command: jupyter notebook --no-browser --port 8888 --ip 0.0.0.0
```

Test it out

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

Run to get the token

```
docker-compose logs mids
```

Open a browser

```
open http://0.0.0.0:8888/?token=<your token>
```

Open New Python3 Notebook

datascience@berkeley

Try redis

```
import redis
r = redis.Redis(host='redis', port='6379')
r.keys()
```

Add some values

```
r.set('foo', 'bar')
value = r.get('foo')
print(value)
```

Drop cluster

```
docker-compose down
```

Redis to track state

Setup

Download data:

```
cd ~/w205/
curl -L -o trips.csv https://goo.gl/QvHLKe
```

Setup

Add volumes to your `docker-compose.yml`:

```
volumes:
  - ~/w205:/w205
```

```
---
version: '2'
services:
  redis:
    image: redis:latest
    expose:
      - "6379"

  mids:
    image: midsw205/base:latest
    stdin_open: true
    tty: true
    volumes:
      - ~/w205:/w205
    expose:
      - "8888"
    ports:
```

Spin up cluster

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

Run to get the token

```
docker-compose logs mids
```

Open a browser

```
open http://0.0.0.0:8888/?token=<your token>
```

Open New Python3 Notebook

```
import redis
import pandas as pd
```

```
trips=pd.read_csv('trips.csv')

date_sorted_trips = trips.sort_values(by='end_date')

date_sorted_trips.head()
```

```
for trip in date_sorted_trips.itertuples():
  print(trip.end_date, '', trip.bike_number, '', trip.end_station_nam
```

```
current_bike_locations = redis.Redis(host='redis', port='6379')
current_bike_locations.keys()
```

Add values

```
for trip in date_sorted_trips.itertuples():
  current_bike_locations.set(trip.bike_number, trip.end_station_name)
```

```
current_bike_locations.keys()
```

Where is bike 92?

```
current_bike_locations.get('92')
```

Drop cluster

```
docker-compose down
```