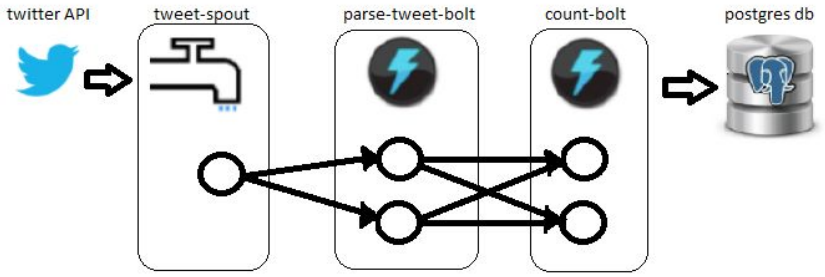
Boris Kletser

Exercise 2: Twitter Application: **Architecture**

Overview

This application takes in tweets and tracks the number of times each word appears.
It runs on streamparse and uses a postgres database.

Figure 1: Application Topology



Tweet-spout takes in the data, parse-tweet-bolt breaks up each tweet into words which get passed onto count-bolt which writes them in a postgres database.

Postgres DB	**Table**	**Fields**
tcount	Tweetwordcount	word - *TEXT PRIMARY KEY NOT NULL* count - *INT NOT NULL*

Setup:

You need to use the UCB MIDS W205 EX2-FULL (ami-d4dd4ec3) AMI, setup postgres, and install psycopg2

AMI: UCB MIDS W205 EX2-FULL (ami-d4dd4ec3) AMI.
Additional Programs: postgres via.setup_ucb_complete_plus_postgres.sh)
https://s3.amazonaws.com/ucbdatasciencew205/setup_ucb_complete_plus_postgres.sh

Database Setup

1. log into user postgress to get access to make the tables
2. create database Tcount
3. connect to tcount (it saves names as lower case)
4. create table Tweetwordcount
5. check that the table was properly created
6. log out of psql and user postgres

```
[root@... ~]# su -w postgres
-bash-4.1$ psql
postgres=# CREATE DATABASE Tcount;
postgres=# \connect tcount;
tcount=# CREATE TABLE Tweetwordcount(word TEXT PRIMARY KEY NOT
NULL, count INT NOT NULL);
tcount=# \dt
tcount=# \q
```

Getting the files and setting up (continued)

7. Install psycopg
8. Clone the git
9. **Update credentials at the top of tweets.py using your favorite program**
    a. Press i to edit, press esc :wq when done to save and exit
    b. Note, if you submit them to github in a public repository, others may be able to use them.

```
[root@... ~]# pip install psycopg2

[root@... ~]# su - w205
[w205@... ]$ mkdir ex2
[w205@... ]$ cd ex2
[w205@... ]$ git clone
https://github.com/superbb/MIDS-W205_A5-EXERCISE2.git
[w205@... ]$ cd ex2/Tweetwordcount/

[w205@... ]$ vim src/spouts/tweets.py
```

Running Queries

1. Go to the Tweetwordcount directory if you're not already there
2. Start the coounter
3. Click **Ctr-C** when you're done.

```
[w205@... ]$ cd ex2/Tweetwordcount/
[w205@... ]$ sparse run
```

Examining The Data

1. finalreturns.py returns an alphabetized list of words with their counts in the stream
2. finalresults.py word retuns the count of appearances by that word

3. histogram.py k1,k2 returns a histogram of words with counts between k1 and k2

```
[w205@... ]$ cd ex2/Tweetwordcount/
[w205@... ]$ python finalresults.py
[w205@... ]$ python finalresults.py boris
[w205@... ]$ python histogram.py 5,14
```

File Structure:

	Ex2
Overview information	| README.md
This doc	| Architecture.pdf
plot-quick.png is based on a short run	| plot-quick.png
plot.png is based on longer run	| plot.png
	|
-Screenshots of the program running	|-screenshots
	| See below for descriptions
	|
Directory originally set up by sparse (you need to run programs from it	|-Tweetwordcount
Final wordcounts	| README.md
Histogram between ranges	| finalresults.py
The files in the project	| histogram.py
	| project.clj
Folder containing the spouts and bolts	|
	|---src
	|-----bolts
Parses the tweets into words	| parse.py
Counts the words	| wordcount.py
	|-----spouts
Takes in tweets coming from twitter API (Make sure it has your credentials!)	| tweets.py
	|---topologies
The topology that makes the streamparse run	| Tweetwordcount.clj

Screenshots (titles describe the stage)

screenshots/screenshot-start.png
screenshots/screenshot-running.png
screenshots/screenshot-break.png
screenshots/screenshot-finalresults-word.png
screenshots/screenshot-finalresults.png
screenshots/screenshot-finalresults-and-histogram.png

screenshots/screenshot-break-after-longer-for-plot.png