**Architecture for Exercise 2 Application**

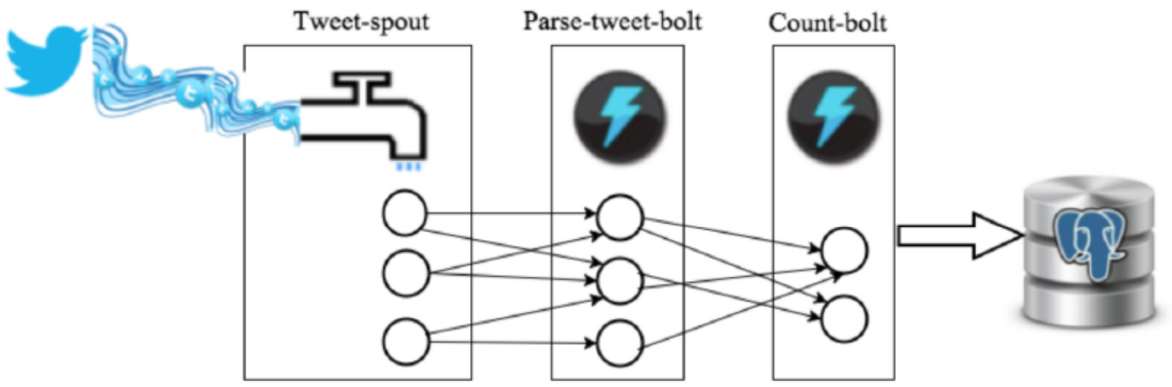


Figure 1: Application Topology

The above figure shows the application topology for the twitter application. The application contains:

1. A spout connected to the Twitter streaming API (3 instances in parallel), that pulls tweets and emits them to the parse bolt.
2. A parse-tweet-bolt (3 instances in parallel), that parses the tweets emitted by the spout, and extracts individual words out of the received tweet text.
3. A count-bolt (2 instances in parallel), that counts the number of words emitted by the tweet-parse bolt, and updates the total counts for each word in the Postgres table. Modify the code in wordcount.py so that it updates the table. You can find sample code on how to use the psycopg library to interact with Postgres in psycopg-sample.py.file.
4. A Postgres database called tcount, with a table called tweetwordcount.

**Serving Scripts:**

The is a python script called **'finalresults.py'**, which when passed a single word as an argument, returns the total number of word occurrences in the stream. For example:

$ python finalresults.py hello
Total number of occurrences of of "hello": 10

Running finalresults.py without an argument returns all the words in the stream, and their total count of occurrences, worted alphabetically, one word per line. For example:

$ python finalresults.py
$ (<word1>, 2), (<word2>, 8), (<word3>, 6), (<word4>, 1), ...

Another python script called '**histogram.py'** gets two integers k1,k2 and returns all the words with a total number of occurrences greater than or equal to k1, and less than or equal to k2. For example:

$ python histogram.py 3,8
<word2>: 8
<word3>: 6
<word1>: 3

**Directory Structure:**

The exercise_2 folder is the main folder for the project on ec2 instance as well as the github repo. It contains the following sub-folders:

1. extweetwordcount: This folder contains the streamparse application files as well as sub folders for:
   a. topologies – this has the tweetwordcount clojure file which ties all bolts and spouts together.
   b. src - this has source code for bolts and spouts.
   c. Other necessary folders like virtualenvs and config.json, project.clj needed for the stream parse project.
2. Screenshots: this folder contains a directory called screenshots/ that has at various screenshots demonstrating an end-to-end execution of the application.
3. Other files: besides the above subfolders, the exercise_2 folder contains other files such as
   a. `Readme.txt`: a file that shows the step-by-step instructions on how to run the application.
   b. `Plot.png`: a bar chart, generated however you prefer, that shows the top 20 words in your Twitter stream.
   c. Serving scripts histogram.py (used to generate plot.png), finalresults.py and other python files needed for the application (Twittercredentials.py, hello-stream-twitter etc.)

**Instructions for running the application:**

1. Connect to the EC2 instance and mount the external hard drive using this command:

mount -t ext4 /dev/xvdf /data

2. Start the postgres database engine by typing this command:

/data/start_postgres.sh

3. Switch to w205 user (optional step):

su - w205

4. Navigate to the directory home/w205/W205/exercise_2/extweetwordcount/
This directory has the twitter application files.

5. Run the application using the command below. This will run the Stream Parse topology end to end. It pulls Tweets, parses, and word counts them, looking at the count bolt for changes and updates Postgres:

sparse run

6. Navigate to the exercise_2 folder which is one level above the current folder.

7. In this step, we execute simple scripts that query the database, and return specific results as follows:

a. python finalresults.py

When passed a single word as an argument, finalresults.py returns the total number of word occurrences in the stream.

b. python histogram.py k1 k2

The script gets two integers k1,k2 and returns all the words with a total number of occurrences greater than or equal to k1, and less than or equal to k2.