Phase 2

Writeup (/writeup/5/3/) | Submit ▾ | Submissions (/submissions/5/3/) | ScoreBoard (/scoreboard/5/3/)

Phase	Open	Deadline
Phase 2	Mar. 19, 2015 00:01	Apr. 01, 2015 16:59

Twitter Analytics on the Cloud

Phase 2

Resource Tagging And Budgets

Tag all of your instances with `Key: 15619project` and `Value: phase2` for all resources

In **addition** to the tag above, all instances in your HBase cluster should be tagged with `Key: 15619backend` and `Value: hbase`, and all instances with MySQL installed should be tagged with `Key: 15619backend` and `Value: mysql`.

You can use any instances for ETL and debugging.

You can use only **m1.large** or cheaper instances (based on on-demand pricing) for your web service.

You can use any free AMI as your base. You should be building your own AMIs for this project.

> Each run (test submitted to the website) must have a maximum budget of **$1.75/hour** (include on-demand EC2, storage and ELB costs. Ignore EMR and Network, Disk I/O costs). Even if you use spot pricing, the constraints that apply pertain to on-demand pricing.
>
> You will have a budget **$60/team** for all the tasks in this phase, **including** the cost for Live Test

Introduction

In the second phase of the 619 project, you will extend your two web services to support more complex queries. Each web service has to respond to four types of queries, with data fetched from your backend storages, which you must design and control.

A report must be written for this phase that conforms to a template (http://goo.gl/FXQVEb).

Query 1(Heartbeat and Authentication)

Query 1 for Phase 2 is the same as Phase 1 (https://theproject.zone/writeup/5/1#Q1).

Query 2(Text Cleaning and Analysis)

Query 2 for Phase 2 is the same as Phase 1 (https://theproject.zone/writeup/5/1#Q2).

Query 3(Retweet Buddies)

This query asks for the retweet relationship for a user. The retweets can be read from 'retweeted_status' field. We require you to process all tweets in Query 3, irrespective of its timestamp. The query asks for a userid (say X), your web service should respond with a list of users (say Y) who have the retweet interaction with user X. There are three kind of relationships:

- Plus relationship(+): if user Y has retweeted any of user X's tweets, but the inverse is not true. You should also count the number of Y's tweets that retweeted X's tweets.
- Minus relationship(-): if user X has retweeted any of user Y's tweets, but the inverse is not true. You should also count the number of X's tweets that retweeted Y's tweets.
- Asterisk relationship(*): if both users have mutually retweeted each others tweets. The count should be the

sum of counts of both relationship.

You should group the users by their retweet relationship with user X, list the relationship count and their userid. The list should be sorted:

1. first, in the order of relationship type(*,+,-)
2. then, in the order of **descending** count **numerically**
3. at last, in the order of **ascending** userid **numerically**.

Notice that the count should be calculated by each **unique** tweetid(**id** field, not **retweeted_statue.id** field). And when querying user X, user X should **not** be included in the list (no self-retweets).

Request Format

```
GET /q3?userid=<uid>
```

Response Format

```
TEAMID,AWS_ACCOUNT_ID1,AWS_ACCOUNT_ID2, AWS_ACCOUNT_ID3\n
*,count1,userid1\n
*,count2,userid2\n
+,count3,userid3\n
+,count4,userid4\n
-,count5,userid5\n
-,count6,userid6\n
```

Sample Request

```
GET /q3?userid=1000000477
```

Sample Response

```
Team,1234-5678-1234,1234-5678-1234,1234-5678-1234
+,1,319616954
+,1,955176360
+,1,2210772459
-,1,719221276
-,1,2380428746
```

Query 4(Trending Hashtags)

This query asks for the popularity of a hashtag during a period. The hashtags can be read from the 'entities' object (not from the text). Unlike Q2, we require you to process all tweets, meaning that the Time Filtering does not apply here in Q4. The request contains the hashtag, the start date and the end date of the period. You should return a tuple of tweetid, userid and tweettime for each tweet containing the hashtag and created during the period.

You should consider **English case-sensitive(0-9,A-Z,a-z)** hashtags only. The range between the start date and end date is **inclusive**. Each tweet in the response should be **unique** and in order of **ascending** tweetid **numerically**.

Request Format

```
GET /q4?hashtag=<string>&start=<yyyy-mm-dd>&end=<yyyy-mm-dd>
```

Response Format

```
TEAMID,AWS_ACCOUNT_ID1,AWS_ACCOUNT_ID2, AWS_ACCOUNT_ID3\n
tweetid1,userid1,tweettime1\n
tweetid2,userid2,tweettime2\n
```

Sample Request

```
GET /q4?hashtag=SamSmith&start=2014-03-26&end=2014-03-30
```

Sample Response

```
Team,1234-5678-1234,1234-5678-1234,1234-5678-1234
448743782499229696,1252920085,2014-03-26+08:50:29
449516318752919552,33208389,2014-03-28+12:00:16
449535969092702208,2206887799,2014-03-28+13:18:21
450136920572440576,162279034,2014-03-30+05:06:19
450326784148381696,293643088,2014-03-30+17:40:46
```

Live Test

This phase will be evaluated using a Live Test. During the Live Test, all web services of all teams will be simultaneously tested. There will be a MySQL Live Test and a HBase Live Test. You should build your web services and submit the DNS for both- the web service with the MySQL backend and the web service with the HBase backend. Each Phase 2 Live Test will last for about 3 hours (per backend type).

Your grade will be calculated as the average of both Live Tests as well as the report. If you focus on doing only one database well (the other one scores below 50/100), your score may be capped.

Grading

Phase 2 accounts for 30% of the total grade for this 15619Project. The grade for Phase 2 will be calculated on the result of both Live Tests and the Report.

Value	Target	Weight	Live Test Duration
Warmup	-	0%	30 min
Q1	15000	5% x2	30 min
Q2	6000	10% x2	30 min
Q3	10000	10% x2	30 min
Q4	6000	10% x2	30 min
Mix-Q1Q2Q3Q4	3000/1500/2500/1500	5% x2	30 min

Report	Excellence	20%	-

Grading Penalties

The following table outlines the violations of the project rules and their corresponding grade penalties for Phase 1

Violation	**Penalty of the project grade**
Using more than $60 to complete this phase(including Live Test)	-10%
Using more than $90 to complete this phase(including Live Test)	-100%
Publishing your code publicly (e.g. Public Repository on Github)	-100%
Copying any code from the Internet, other teams, solutions from previous semesters, or anywhere	-200% at least
Any kind of collaboration across teams	-200% at least

Query Reference Server

To help your team through the 15619Project design, implementation and system test, we provide a query reference server (http://ec2-54-85-149-134.compute-1.amazonaws.com)

Using the query reference server, you can submit queries Q1 - Q4 to study the expected results. Access to this server is restricted to teams who demonstrate progress to the course staff in a weekly meeting with an assigned TA who will be your team's mentor.

> Your team should meet the assigned TA mentor every week to talk about your progress and challenges, as well as the contribution of each member of the team. After the weekly meeting with the TA mentor, your team will be handed an authentication token with a limited time validity, which you can use to submit requests against the query reference server.