15619 Project Phase 2 Report

Performance Data and Configurations

	Front end	Web service with HBase	Web service with MySQL
Query	q1	q2/q3/q4/mixed	q2/q3/q4/mixed
Scoreboard request ID	11694	27260/27327/27393/27605/27606/27607/27609	26679/26746/26838/27117/27118/27119/27110
Submission timestamp	2014-10-29 23:07:27 (-0700)	2014-11-06 23:37:15 (-0800)	
Instance type	m3.large	front end: t2.small t2.micro backend: two clusters with 1master + 2 core respectively, both m1.large	m1.large
Number of instances	1	2 small, 10 micro	2 large
Queries Per Second (QPS)	14663.4	Q2 379 Q3 1604 Q4 1773	Q2 2.6 Q3 4174.5 Q4 6477.3
Error rate	0.00	0	92
Correctness	100.00	100	8
Cost per hour	$0.140	$1.25	$0.35

Do you want to be graded on MySQL or HBase? _Hbase_

Task 1: Front end

Questions
1. Which front end system solution did you use? Explain why did you decide to use

this solution.
Tomcat. We also tried Play Frameworks. But we couldn’t solve some problems we met when connecting the front end to the Hbase. So the Play Framework is only used in part of the MySQL front end currently (some of). Since tomcat has a more stable performance than other front ends do.

2. Explain your choice of instance type and numbers for your front end system.
Instance type: We used 2 t2.small, and 10 t2.micro.
We noticed that the throughput between a small instance and a large instance is very tiny for Q3 and Q4. Also we found the CPU utilization of the front end is very low. So we move to smaller instances. However, when we used multiple small instances with ELB, we observed a much greater improvement in thoughput. So we use multiple tiny instances

3. Did you do any special configurations on your front end system? Explain your design decisions and provide details here.
We tried to modify the number of threads used in the front end system. However, there’s no significant effect. And we tried to use Htable pool, which was not helping a lot either.
We don’t think our front end is efficient enough. But we did try many ways to improve it. But the result was always disappointing.

4. What is the cost to develop the front end system?
We usually used micro instance to develope the front end. The price ranges between $0.013 per instance hour, the total cost is ~ $0.052

5. What did you change from Phase 1 and why?
We tried the play framework but failed. We used an array of HTable class in front end to add the performance.

Task 2: Back end (database)

Questions
1. Describe your table design for both HBase and MySQL. Explain your design decisions.
Q2:

Hbase	
Key	create_at+userid
Column Family	sorted tweetid:score:text
MySQL	

Primary Key	create_at+userid, VARCHAR(83)
TweetidScoreText	sorted tweetid:score:text, TEXT

Q3:

Hbase	
Key	userid
Column Family	sorted retweet userids
MySQL	
Primary Key	userid, VARCHAR(64)
Retweetids	sorted retweet userids, TEXT

Q4:

Hbase	
Key	date+location
Column Family	value: hashtag:tweetids, qualifier: rank
MySQL	
Key	date+location, VARCHAR(50)
Rank	rank, INT
HashtagTweetids	hashtag:tweetids, TEXT

2. What is the cost to develop your back end system?
We used 19 m1.large instances to load the data into the Hbase and load 1 m1.large to load data for MySQL. In both cases, we used spot instance. We put 0.017 or 0.018 for most of the time.
Q2 takes around 2 hours, Q3 takes 3 hours and 43 mins, Q4 take 10 mins.
Q2 takes around 8 hours, Q3 takes 7 hours, Q4 takes 40 mins.
The total cost is: $2.92

Task 3: ETL
Since ETL was performed for both HBase and MySQL, you will be required to submit information for each type of database.

MySQL:

1. **The code for the ETL job**
   Please refer to the attached README file to find code for ETL jobs.

2. **The programming model used for the ETL job and justification**
   We used java mapreduce programs to do the ETL job. The output csv file only contains the information we needed for this query. This job dramatically reduced the amount of data to be stored in the database. (from ~1TB to ~ 1.7G(Q3) and ~1.2G(Q4)). We used EMR since the data set is too big to run on our local machine.

3. **The type of instances used and justification**
   We primarily used (spotted) m1.large to do the ETL job. The spot price is very reasonable (~$0.017/hr) and the performance is good.

4. **The number of instances used and justification**
   Varies from 3~19 instances.

5. **The spot cost for all instances used**
   Q2 ETL takes 2 hours, Q3 ETL takes about 3 hours*2, Q4 ETL takes about 44 mims*3. The spot instance price is 0.017~0.018
   The total instance cost is $4.14

6. **The execution time for the entire ETL process**
   about 3 hours for Q2, about 3 hour for Q3, and 44 min for Q4

7. **The overall cost of the ETL process**
   About $10 for all the ETL jobs (both MySQL and HBase)

8. **The number of incomplete ETL runs before your final run**
   Usually 3. But we have about 3 complete runs with 19 instances generating incorrect result.

9. **Discuss difficulties encountered**
   During the live test, there may be something wrong with the ELB so most of the MySQL tests didn't get through in the beginning, and most of them failed after we fixed the ELB connection problem.

10. **The size of the resulting database and reasoning**
    The size of the Q2 and Q3 is pretty similar to the size of the ETL output. While we build index for Q4 on rank. So the size should be a little bigger thant the ETL output size.
    Q2: 32G
    Q3: 2G

Q4: 3.5G

11. **The time required to backup the database on S3**
    about 20min

12. **The size of S3 backup**
    Q2: 32G, Q3: 2.7G Q4:4G

<u>HBase</u>:

13. **The code for the ETL job**
    Please refer to the attached Hitman java file.

14. **The programming model used for the ETL job and justification**
    The .csv files are the same with those in MySQL ETL
    Use Map Reduce to transfer the csv file to the HFiles.
    Then configure the environment and load the data to Hbase.

15. **The type of instances used and justification**
    We primarily used (spotted) m1.large to do the ETL job. The spot price is very reasonable (~$0.017/hr) and the performance is good.

16. **The number of instances used and justification**
    Varies from 3~19 instances.

17. **The spot cost for all instances used**
    Q2 ETL takes 2 hours, Q3 ETL takes about 3 hours*2, Q4 ETL takes about 44 mims*3. The spot instance price is 0.017~0.018
    The total instance hours are 4.14

18. **The execution time for the entire ETL process**
    About 3 hours for Q2, about 3 hour for Q3, and 44 min for Q4

19. **The overall cost of the ETL process**
    About $10 for all the ETL jobs (both MySQL and HBase)

20. **The number of incomplete ETL runs before your final run**
    Usually 3. But we have about 3 complete runs with 19 instances generating incorrect result.

21. **Discuss difficulties encountered**
    We tried many ways to improve the performance of our front end. We spent more time in designing the ETL job so that we can use the data without any modification right after we

get the data out from the database. But we still didn't get a decent performance. We also tried to set different front end environment, like increasing thread count and using Htable pool. But our performance was still disappointing

The ETL part is pretty much the same as MySQL. Apart from that, when setting up the live test. We found out that our Q2 and Q3 Hbase table names were the same. But it was too late for us to modify the table. So we ended up with having two hbase backend. One with table for Q2, Q4 and another with table for Q3 and Q4. With two hbase backend, we don't have enough money to get larger instances for front end. This may undermine our performance seriously during the test.

22. **The size of the resulting database and reasoning**

Since we store the hfiles for Q3 and Q4, we load the hfiles into hbase later. So we think hfiles size is very close to the database size. For Q2, we don't have the hfiles. But since the design for Q2 is pretty similar to Q3, we think the size is pretty similar to the size of its backup

Q2: 33G
Q3: 3.0G
Q4: 1.5G

23. **The time required to backup the database on S3**

    It usually takes seconds to backup, but for Q2 it takes about 15min

24. **The size of S3 backup**

    Q2: 33G, Q3: 3.0G Q4:4.4G

Questions

1. What are the advantages and disadvantages of MySQL for each of the queries you've encountered so far? Which queries are better suited for MySQL (not HBase)?

If the key itself can't determine the value; we also need qualifier to get the result, then we think MySQL is more suitable. For example, Q4 in our case.

2. What are the advantages and disadvantages of HBase for each of the queries you've encountered so far? Which queries are better suited for MySQL (not HBase)?

If the value for a key is unique, we think Hbase is more suitable for this kind of query. For example, Q2 and Q3 in our example.

3. For your backend design, what did you change from Phase 1 and why?

   We used multiple backends rather than one single backend. During our test, it turned out to be good. But from the live test, the result is not thriving.