**S15 15619 Project Phase 2 Report**

**Performance Data and Configurations**

|  |  |
| --- | --- |
| Best Configuration and | Results from the Live Test |
| Number and type of instances | HBase Live Test:  MySQL Live Test: |
| Cost per hour  (assume on-demand prices) | HBase Live Test:  MySQL Live Test: |
| Queries Per Second (QPS) | INSERT HERE: (Q1,Q2H,Q2M,Q3H,Q3M,Q4H,Q4M)   |  |  |  |  |  |  |  |  | | --- | --- | --- | --- | --- | --- | --- | --- | |  | Q1 | Q2H | Q2M | Q3H | Q3M | Q4H | Q4M | | score |  |  |  |  |  |  |  | | tput |  |  |  |  |  |  |  | | ltcy |  |  |  |  |  |  |  | | corr |  |  |  |  |  |  |  | | err |  |  |  |  |  |  |  | |
| Rank on the scoreboard: | Q1:  Q2H:  Q2M:  Q3H:  Q3M:  Q4H:  Q4M:  HBase Live Test:  MySQL Live Test: |

**Team :**

**Members :**

**Rubric:**

**Each unanswered question = -7%**

**Each unsatisfactory answer = -3%**

**[Please provide an insightful, data-driven, colorful, chart/table-filled, and interesting final report. This is worth a quarter of the grade for Phase 2. Use the report as a record of your progress, and then condense it before sharing it with us. Questions ending with “Why?” need evidence (not just logic)]**

**Task 1: Front end (you may/should copy answers from your earlier report-- each report should form a comprehensive account of your experiences building a cloud system. Please try to add more depth and cite references for your answers from the P1Report)**

**Questions**

1. Which front end framework did you use? Explain why you used this solution. [Provide a small table of special properties that this framework/platform provides].
2. Did you change your framework after Phase 1? Why or why not?
3. Explain your choice of instance type and numbers for your front end system.
4. Explain any special configurations of your front end system.
5. Did you use an ELB for the front-end? Why, or why not? Condense your experience with ELB in the next few sentences. Talk about load-balancing in general and why it matters in the cloud.
6. Did you explore any alternatives to ELB? List a few of these alternatives. What did you finally decide to use? (if possible) Provide some graphs comparing performance between different types of systems.
7. Did you automate your front-end instance? If yes, how? If no, why not?
8. Did you use any form of monitoring on your front-end? Why or why not? If you did, show us a capture of your monitoring during the Live Test. Else, try to provide CloudWatch logs of your Live Test in terms of memory, CPU, disk and network utilization. Demarcate each query clearly in the submitted image capture.
9. What was the cost to develop the front end system?
10. What are the best reference URLs (or books) that you found for your front-end? Provide at least 3.

[Please submit the code for the frontend in your ZIP file]

**Task 2: Back end (database)**

**Questions**

1. Describe your schema for each DB. Explain your schema design decisions. Would your design be different if you were not using this database? How many iterations did your schema design require? Also mention any other design ideas you had, and why you chose this one? Answers backed by evidence (actual test results and bar charts) will be valued highly.
2. What was the most expensive operation / biggest problem with each DB that you had to resolve for each query? Why does this problem exist in this DB? How did you resolve it? Plot a chart showing the improvements with time.
3. Explain (briefly) **the theory** behind (at least) 7 performance optimization techniques for databases. How are each of these implemented in MySQL? How are each of these implemented in HBase? Which optimizations only exist in one type of DB? How can you simulate that optimization in the other (or if you cannot, why not)? Use your own words (paraphrase, this document goes through plagiarism detection software).
4. Plot a graph showing results with/without each individual optimization that you used. Extremely impressive will be a timeline of rps v/s submission id (mentioning which optimization was in use at that time).
5. Would your design work if your web service also implemented insert/update (PUT) requests? Why or why not?
6. Which API/driver did you use to connect to the backend? Why? What were the other alternatives that you tried? What changed from Phase 1?
7. Can you quantify the speed differential (in terms of rps or Mbps) between reading from disk versus reading from memory? Did you attempt to maximize your usage of RAM to store your tables? How much (in % terms) of your memory could you use to respond to queries?
8. Does your usage of HBase maximize the utility of HDFS? How useful is it to have a distributed file system for this type of application/backend? Does it have any significant overhead?
9. How can you reduce the time required to perform scan-reads in MySQL and HBase?
10. Did you use separate tables for Q2-Q4 on HBase and MySQL? How can you consolidate your tables to reduce memory usage?
11. How did you profile the backend? If not, why not? Given a typical request-response for each query (q2-q4) what percentage of the overall latency is due to:
    1. Load Generator to Load Balancer (if any, else merge with b.)
    2. Load Balancer to Web Service
    3. Parsing request
    4. Web Service to DB
    5. At DB (execution)
    6. DB to Web Service
    7. Parsing DB response
    8. Web Service to LB
    9. LB to LG

How did you measure this? A 9x6 (Q2H to Q4M) table is one possible representation.

1. What was the cost to develop your back end system?
2. What were the best resources (online or otherwise) that you found. Answer for HBase, MySQL and any other relevant resources.

[Please submit the code for the backend in your ZIP file]

**Task 3: ETL**

1. For each query, write about:
   1. The programming model used for the ETL job and justification
   2. The number and type of instances used and justification
   3. The spot cost for all instances used
   4. The execution time for the entire ETL process
   5. The overall cost of the ETL process
   6. The number of incomplete ETL runs before your final run
   7. Discuss difficulties encountered
   8. The size of the resulting database and reasoning
   9. The size of the backup
2. What are the most effective ways to speed up ETL? How did you optimize writing to MySQL and HBase? Did you make any changes to your tables after writing the data? How long does each load take?
3. Did you use EMR? Streaming or non-streaming? Which approach would be faster and why? How can you parallelize loading data into MySQL?
4. Did you use an external tool to load the data? Which one? Why? If you were to do Q2 (Phase 1) ETL again, what would you do differently?
5. Which database was easier to load (MySQL or HBase)? Why?

[Please submit the code for the ETL job in your ZIP file]

**General Questions**

1. Would your design work as well if the quantity of data would double? What if it was 10 times larger? Why or why not?
2. Did you attempt to generate load on your own? If yes, how? And why?
3. Describe an alternative design to your system that you wish you had time to try.
4. Which was/were the toughest roadblock(s) faced in Phase 2?
5. Did you do something unique (any cool optimization/trick/hack) that you would like to share with the class?

**[NOTE:**

* **IN YOUR SUBMITTED ZIP FILE, PLEASE DO NOT PUT MORE ZIP OR GZ FILES (NO NESTED ARCHIVES)**
* **USE A SIMPLE FORMAT**
  + **/etl**
  + **/frontend**
  + **/backend**

**]**