**S17 15619 Team Project Phase 1 Report**

**Team Name:**

**Members (First Name, Last Name, Andrew ID):**

**Performance Data and Configurations**

|  |  |
| --- | --- |
| Best Configuration | Results |
| Number and type of instances | Q1:  Q2H:  Q2M:  Q3H:  Q3M: |
| Cost per hour of entire system |  |
| Queries Per Second (QPS) of your best submission | |  |  |  |  |  |  | | --- | --- | --- | --- | --- | --- | |  | Q1 | Q2H | Q2M | Q3H | Q3M | | score |  |  |  |  |  | | submission id (for the above score) |  |  |  |  |  | | throughput |  |  |  |  |  | | latency |  |  |  |  |  | | correctness |  |  |  |  |  | | error |  |  |  |  |  | |

**Rubric:**

**Each unanswered question = -5%**

**Each unsatisfactory answer = -2%**

**Use the report as a record of your progress, and then condense it before sharing with us.**

* **Questions ending with “Why?” need evidence (not just logic)**
* **Always use your own words (paraphrase); we will check for plagiarism**

**Task 1: Web tier**

**Questions**

1. Compare the functionality, configuration, deployment and performance of the two web-tier frameworks that you compared in your evaluation. Describe your team’s configuration of both frameworks that enabled you to achieve an RPS of 24K.
2. Describe the design process and architecture of your selected web tier. Draw the architecture and provide experimental evidence as well as arguments to support your design decisions.
   1. For the selected web tier framework, explain why you selected this solution. [Provide a small table of special properties that this framework/platform provides]
   2. Explain your choice of instance type and numbers for your web tier.
3. Explain any optimizations or special configurations used in your web tier. [provide experimental results or charts to support your decision as to which optimization provided improvements in performance]
4. Did you use an ELB for the front-end? Why, or why not? Describe your experience with ELB in a few sentences. Discuss load-balancing in general, why it matters in the cloud and how it’s applicable to your web tier.
5. 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.
6. How did you package and automate the deployment of your system? What steps are taken every time you deploy your system.
7. Did you use any form of monitoring of your front-end? Why or why not? If you did, show us what you monitored and the results.
8. What was the cost to develop the web tier? You can use tagging to figure out the cost to develop and test your web tier.

1. What are the best reference URLs (or books) that you found for your front-end? Provide at least 3.

1. Please provide a comparison between at least two web frameworks. What were the most important differentiating factors?

**Task 2: Database (Q2 MySQL, Q2 HBase, Q3 MySQL, Q3 HBase)**

**Answer the following questions for Q2 and Q3**

1. Describe your schema.
   1. Explain your schema design decisions.
   2. Would your design be different if you were not using this database?
   3. How many iterations did your schema design require?
   4. Also mention any other design ideas you had, and why you chose this one?
   5. Your answers should be backed by evidence (actual test results and bar charts).
2. What was the most expensive operation / biggest problem with your DB that you had to resolve for this query (Q2 and Q3)? 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) 3 performance optimization techniques for databases in general. How are each of these optimizations implemented in MySQL? How are each of these optimizations implemented in HBase? Which optimizations only exist in one type of DB and why? How can you simulate that optimization in the other (or if you cannot, why not)? .
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 database? Why? What were the other alternatives that you tried?
7. You should profile your database to understand the source of a bottleneck. Describe how you profiled your database databases?

We expect you to know the end-to-end latency of your system. Given a typical request-response for Q3 query, describe the latency of each of these components:

* 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

For each step, think about how you could measure it, and write down the tools / techniques that you used.

**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?
3. Did you use EMR? Streaming or non-streaming? Which approach would be faster and why?
4. Did you use an external tool to load the data? Which one? Why?
5. Which database was easier to load (MySQL or HBase)? Why?

[Please submit the code for the ETL job in your code 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 roadblocks faced in Phase 1?
5. Did you do something unique (any cool optimization/trick/hack) that you would like to share?