15-319 / 15-619 Cloud Computing

Recitation 10

Oct 31th 2017

Overview

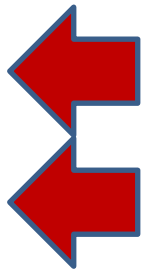
- **Last week's reflection**
  - Team Project, Phase 1, Queries 1, 2
  - Unit 4 - Modules 15
  - Quiz 8
- **This week's schedule**
  - Team Project, Phase 1, report
  - Team Project, Phase 2, Queries, 1, 2, 3
  - Project 4.1, Batch Processing with MapReduce
  - Project 2.3. Serverless Functions (Make-up - Optional)
  - Unit 4 - Modules 16 and 17
  - Quiz 9
- **Twitter Analytics: The Team Project**

Reminders

- Monitor AWS expenses regularly and tag all resources
  - Check your bill both on AWS and TPZ
- Piazza Guidelines
  - Give your submission ID
- Provide clean, modular and well documented code
  - Large penalties for not doing so.
  - **Double check** that your code is submitted!!
- Utilize Office Hours
  - No one comes to mine office hour

Modules to Read

- UNIT 4: Cloud Storage
  - Module 14: Cloud Storage
  - Module 15: Case Studies: Distributed File System
    - HDFS
    - Ceph
  - Module 16: Case Studies: NoSQL Databases
  - Module 17: Case Studies: Cloud Object Storage

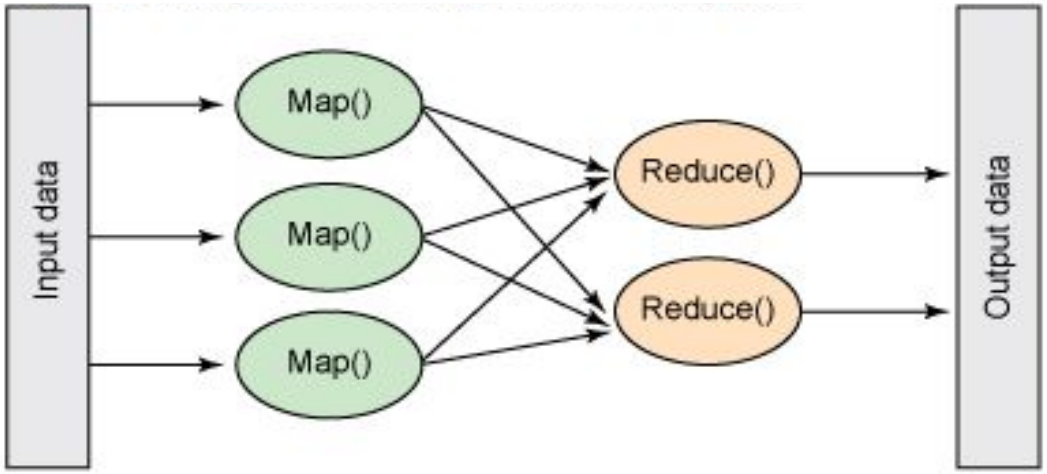


Project 4

- Project 4.1, Batch Processing with MapReduce
  - MapReduce Programming
- Project 4.2
  - Iterative Programming Using Apache Spark
- Project 4.3
  - Stream Processing using Kafka/Samza

Introduction to MapReduce

- **Definition**: Programming model for processing large datasets with a parallel, distributed algorithm on a cluster
- **Phases of MapReduce:**
  - **Map**
  - **Shuffle**
  - **Reduce**

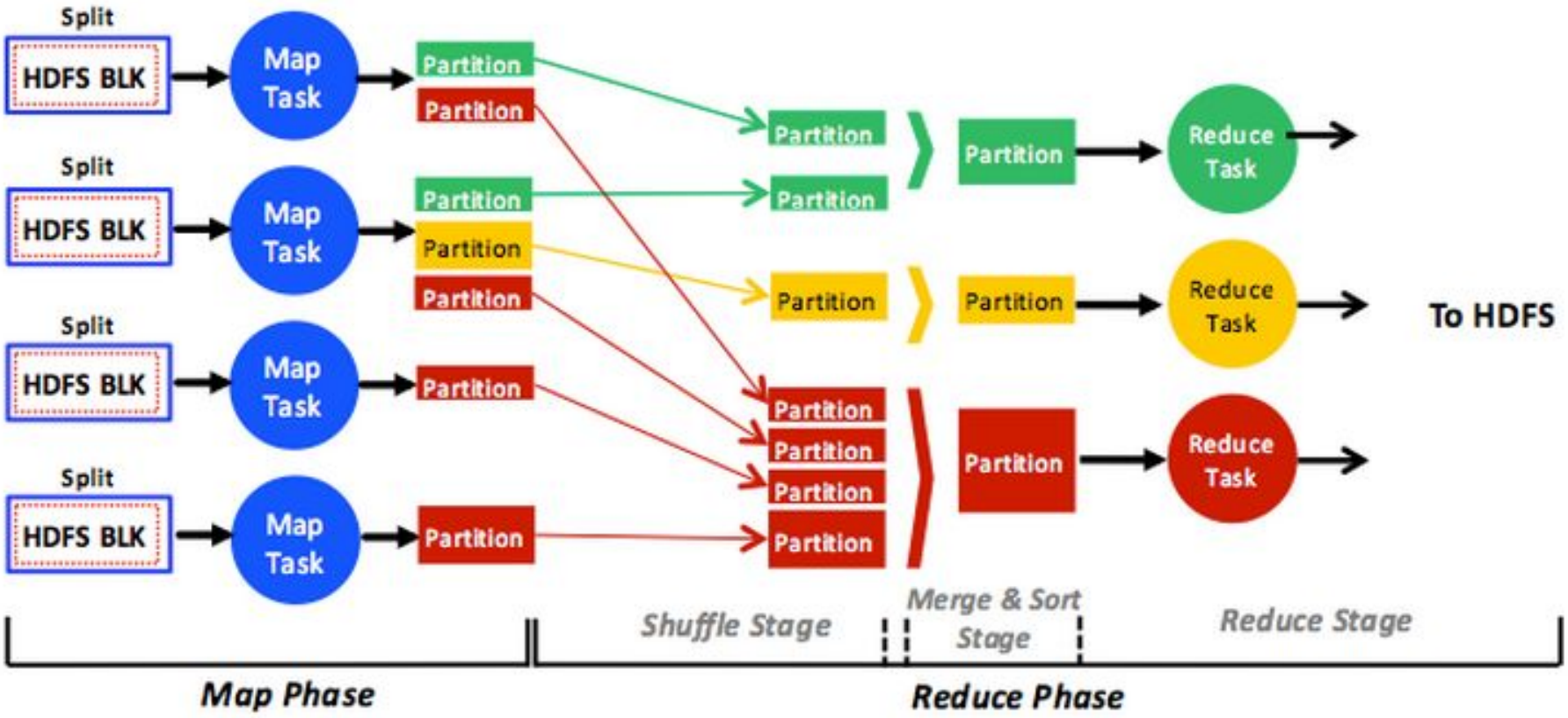


MapReduce: Framework

- The MapReduce framework:
  - Partitions the input data
  - Schedules the program's execution across a set of machines
  - Performs the group by key (sort & shuffle) step
  - Handles machine failures
  - Manages required inter-machine communication

MapReduce and HDFS

- Detailed workflow



MapReduce Data Types - 1

- Mapper (default)
  - Input: **key-value pairs**
    - **Key:** byte offset of the line
    - **Value:** the text content of the line
  - Output: **key-value pairs**
    - **Key:** specified by your program
    - **Value:** specified by your program based on what content you expect the reducer to receive as a list

(k1,v1) -> Mapper -> (k2,v2)

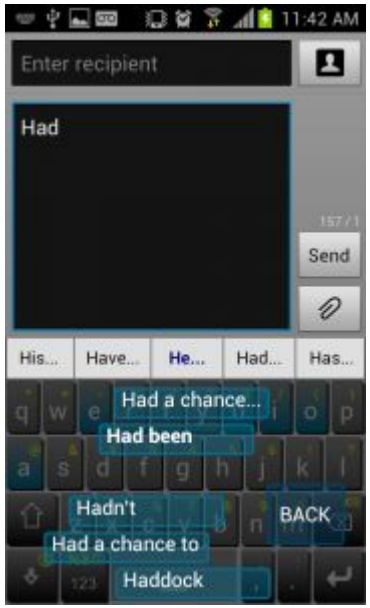
MapReduce Data Types - 2

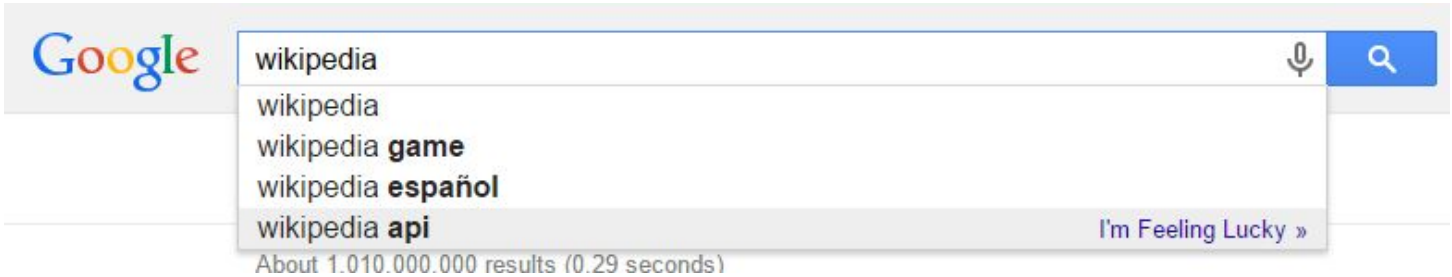
- Reducer
  - Input: **key-value pairs**
    - A list of values for each key output from the mapper
  - Output: **key-value pairs**
    - The desired result from your aggregation

(k2,list(v2)) -> Reducer -> (k3,v3)

Project 4.1 - Input Text Predictor

- Suggest words based on phrases already typed
  - Use an English Corpus to build a language model





Project 4.1 - Input Text Predictor

- Steps to build an Input Text Predictor
  - Implement and optimize a WordCount job
  - Utilize a raw XML dataset of English text
  - Extract XML content and clean the input data
  - Perform the N-Gram count
  - Build the Statistical Language Model
  - Predict the next word given a phrase
- You have to use a Custom JAR in EMR
  - **CANNOT use EMR Streaming as in P1.2**

MapReduce Tutorial Task

Learn the sources of potential bottlenecks or overhead and how to overcome them.

1. Given a wordcount job code
   - Pom file
   - The implementation
2. Correct and optimize the code to meet the requirements
   - Correct the **code** to get correct outputs
   - Profile your job by checking the runtime of mappers and reducers on the web console
   - Use proper variable types to save time/space

Task 1 - Count phrases (n-grams)

1. Given a language corpus
   - Wikipedia dataset (~7.8 GB)
   - Clean the dataset
2. Construct an n-gram model of the corpus
   - An n-gram is a phrase with n contiguous words
   - For example a set of 1,2,3,4,5-grams with counts:
     - this 1000
     - this is 500
     - this is a 125
     - this is a cloud 60
     - this is a cloud computing 20

Task 2 - Statistical Model

3. Build a Statistical Language Model to calculate the probability of a word appearing after a phrase

$$\Pr(\text{word} \mid \text{phrase}) = \frac{\text{Count}(\text{phrase} + \text{word})}{\text{Count}(\text{phrase})}$$

$$\Pr(\text{is} \mid \text{this}) = \frac{\text{Count}(\text{this is})}{\text{Count}(\text{this})} = \frac{500}{1000} = 0.5$$

$$\Pr(a \mid \text{this is}) = \frac{\text{Count}(\text{this is } a)}{\text{Count}(\text{this is})} = \frac{125}{500} = 0.25$$

4. Load the probability data to HBase and predict the next word based on the probabilities

Task 2 - Phrase auto-completion

- MapReduce for phrase auto-completion
  - Given a prefix, suggest the most possible phrases
  - Example: given “carnegie”,
    - Possible phrases are: carnegie mellon, carnegie library, carnegie mellon university faculty….
    - Suggest at most **8 phrases** with the highest probability, and make suggestions on all possible lengths using the n-gram data
    - Three 1-gram, two 2-grams, two 3-grams and one 4-gram
  - Store the probability data to HBase and connect our front-end in order to submit

Hints/Recommendations

- Test for correctness with a small dataset first
- Task 1 submission could take a long time, plan ahead and know what you are supposed to do before submission
- **Don't** start a new cluster for every job
  - EMR will charge you one hour of usage for instances even though your EMR job failed to start
  - The result of task 1 is useful for task 2 and bonus!
- Version of Hadoop
  - It should match the version shown in the EMR AMI
- Start early and try the bonus

Project 4.1 FAQ

- MapReduce job failures or errors
  - Always test locally first
  - On step failure, continue or terminate?
  - Hadoop UI
    - Find job logs
    - Locate the source of the problem
    - Solve the problem
  - For memory problems:
    - Restrict the number of mappers and reducers
    - Increase the Java program memory

Project 4.1 FAQ

- Could not get a full score on the tasks
  - Test your regex…for some corner cases!
  - Incorrect use of combiner
    - Filtered words in Combiner and Reducer

Upcoming Deadlines

- Quiz 9: Unit 4 - Modules 16 & 17

  Due: Friday, Nov 3 2017 11:59PM Pittsburgh

- Project 2.3: Serverless Functions

  Due: Thursday, Nov 5 2017 11:59PM Pittsburgh

- Project 4.1: Batch Processing with MapReduce

  Due: Sunday, Nov 5 2017 11:59PM Pittsburgh

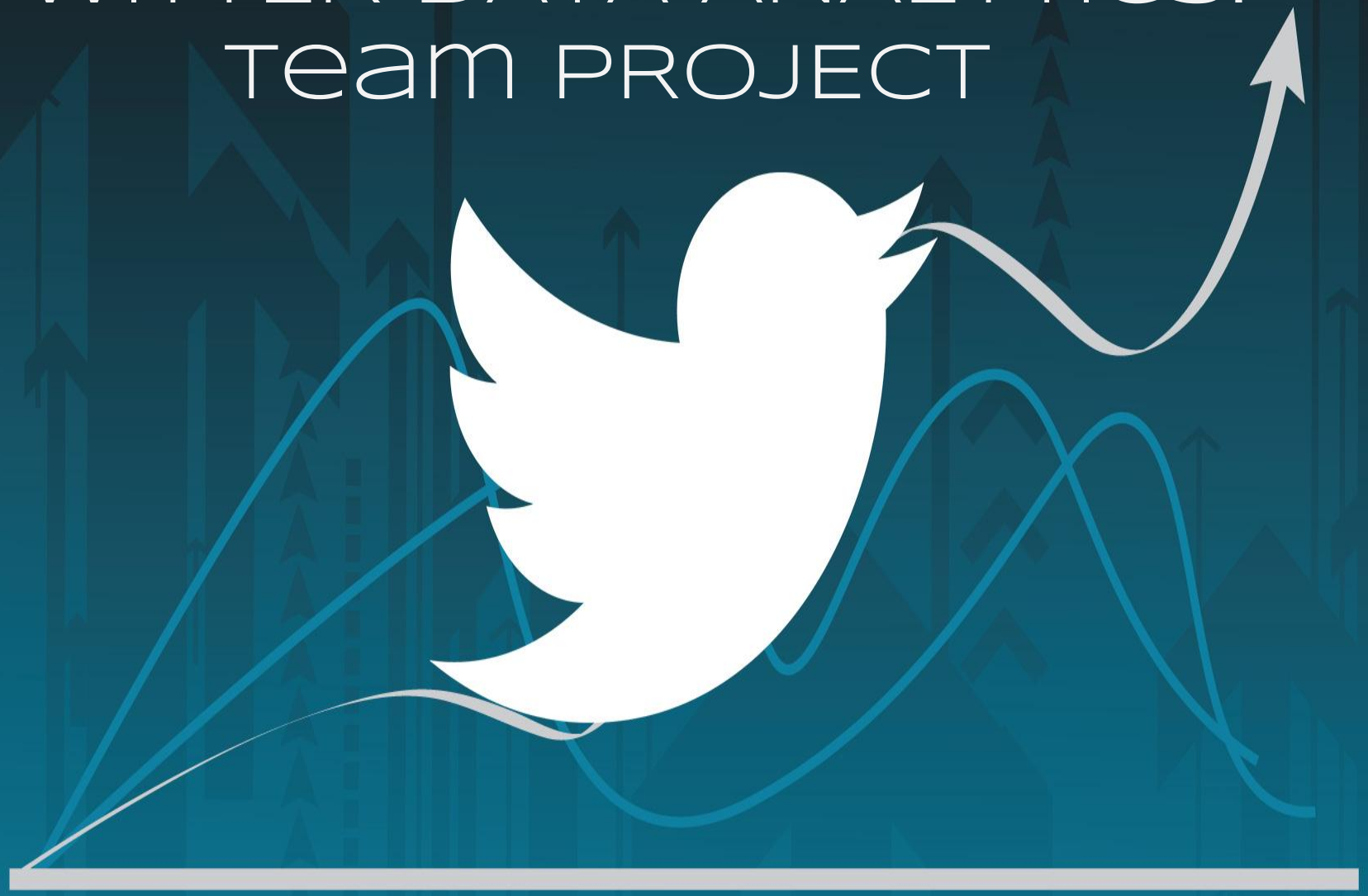
- Team Project - Phase 1

  Due: Tuesday, Oct31 2017 11:59PM Pittsburgh

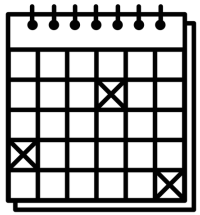
- Team Project - Phase 2 - ongoing

Questions?

TWITTER DATA ANALYTICS: Team PROJECT



Team Project Time Table



Phase	Query	Start	Deadline	Code and Report Due
Phase 1	Q1	Monday 10/9/2017 00:00:01 EST	Sunday 10/22/2017 23:59:59 EST	-
	Q2	Monday 10/9/2017 00:00:01 EST	Sunday 10/29/2017 23:59:59 EST	Tuesday 10/31/2017 23:59:59 EST
Phase 2	Q1, Q2, Q3	Monday 10/30/2017 00:00:01 EST	Sunday 11/12/2017 15:59:59 EST	-
	Live Test Q1, Q2, Q3	Sunday 11/12/2017 18:00:01 ET	Sunday 11/12/2017 23:59:59 EST	Tuesday 11/14/2017 23:59:59 EST
Phase 3	Q1, Q2, Q3, Q4	Monday 11/13/2017 00:00:01 ET	Sunday 11/26/2017 15:59:59 EST	-
	Live Test Q1, Q2, Q3, Q4	Sunday 11/26/2017 18:00:01 ET	Sunday 11/26/2017 23:59:59 EST	Tuesday 11/28/2017 23:59:59 EST

Note:

- There will be a report due at the end of each phase, where you are expected to discuss optimizations
- **WARNING: Check your AWS instance limits on the new account (should be > 10 instances)**

Team Project Phase 1 Review

- Q1
  - Building a heartbeat and authentication web service.

- Q2
  - Handling complex read-only queries.
  - Doing ETLs, building, configuring and optimizing Web Tier and Database Tier.
  - Test MySQL and HBase respectively.

Phase 1, Query 1 Reflection

- Most teams did an excellent job!
  - Compared and tested different web frameworks
  - Configured a fast frontend
- Teams that exceeded our expectations
  - Tourists (dongxiuj, shuny, shilund)
  - Carnegie in NASA (wennad, dayuj)
- Remember to write your auto-deployment (orchestration) script for Query 1!

Phase 1, Query 2 Tips

- Use regex “\p{L}+” to match words in Java
  - So that we only match the unicode letters.
- Treat all contents in the `text` field the same, including hashtag. Meaning if the text contains hashtags, these hashtags can be considered as a word and need to be included as a keyword.
- Keywords and hashtags are case insensitive.
- Teams that exceeded our expectations
  - Carnegie in NASA (wennad, dayuj)
  - BitsPlease (subhadeb, skandhp, byadav)
  - Tricorn (pengchoz, jiakaiz1, yuano)

Query 3

- Problem Statement
  - In a time range and a user id range, which tweets have the most impact and what are the topic words?
- Impact score and topic words (write up for details)
  - Impact of tweets: Which tweet is “important”? Calculate using the effective word count and statistics like retweet count and follower count.
  - Topic words: In this given range, what words could be view as a “topic”? Done using TF-IDF
- Request/Response Format
  - Request: Time range, uid range, #words, #tweets
  - Response: List of topic words with their topic score, as well as a list of censored tweets

Phase 2, Query 3 FAQs

Question 1: How to calculate the topic score?

- Calculate the IDF score idf(w) of word w in the given range of tweets.
- Calculate the term frequency of word w in i-th tweet Ti.
- The impact score of i-th tweet Ti is impact(i).
- Topic score for word w is
  - SUM(Ti*idf(w)*ln(impact(i)+1)) (For tweets in given range)

Phase 2, Query 3 FAQs

Question 2: When to censor? When to exclude stop words?

- Censor in the Web Tier or during ETL. It is your own choice.
  - If you censor in ETL, consider the problem it brings to calculating the topic word scores (two different words might look the same after censoring).

- You should count stop words when counting the total words for each tweet in order to calculate the topic score. Exclude stop words when calculating the impact score and selecting topic words.

Performance Tuning Tips

- To do performance tuning, you first need to identify which part of your system is the bottleneck.
  - Do profiling and monitoring on your system
    - Write a LG yourself to test your system performance
    - Use CloudWatch for resource utilization such as CPU, Network, Disk, etc.

Performance Tuning Tips

- Think about the architecture of your system and what advantages and disadvantages it has compared to other settings
  - Sharding vs Replication
  - ELB vs Customized LB
  - Doing the calculation in Web Tier vs in ETL, etc.

Performance Tuning Tips

- Web Tier
  - Did you put too much computation at the Web Tier?
  - If you have multiple Web Tier servers, is the workload distributed evenly?
  - Have you optimized your algorithm?

Performance Tuning Tips

- Database Tier
  - Try to reduce the number of rows / the size of data retrieved in each request.
  - Remember that Q2 & Q3 are read-only.
    - You can choose schemas that are specifically optimized for Q2 & Q3.

Performance Tuning Tips

- Database Tier - MySQL
  - Tune the parameters
    - Check the official documentation
    - Search Google for MySQL performance tuning

- Database Tier - HBase
  - Tune the parameters
  - Be aware of data distribution and try to identify hotspots
  - Scan can be really slow, try to avoid it when possible
    - If not, try to scan as few rows as possible

Performance Tuning Tips

- Review what we have learned in previous project modules
  - Scaling out
  - Load balancing
  - Replication and Sharding
- Ask on Piazza or go to office hours if you are stuck for too long!

Reminder

- Your team has a total AWS budget of **$50** for Phase 1
- Your web service should cost ≤ **$0.83/hour**, including:
  - EC2
    - We evaluate your cost using the On-Demand Pricing towards **$0.83/hour** even if you use spot instances.
  - EBS & ELB
  - Ignore data transfer and EMR cost
- Targets:
  - Query 2 - 10000 rps (for both MySQL and HBase)
  - Query 3 - 1500 rps (for both MySQL and HBase)

Reminder

Phase	Query	Start	Deadline	Code and Report Due
Phase 1	Q1	Monday 10/9/2017 00:00:01 EST	Sunday 10/22/2017 23:59:59 EST	-
	Q2	Monday 10/9/2017 00:00:01 EST	Sunday 10/29/2017 23:59:59 EST	Tuesday 10/31/2017 23:59:59 EST
Phase 2	**Q1, Q2, Q3**	**Monday 10/30/2017 00:00:01 EST**	**Sunday 11/12/2017 15:59:59 EST**	**-**
	Live Test Q1, Q2, Q3	**Sunday 11/12/2017 18:00:01 ET**	**Sunday 11/12/2017 23:59:59 EST**	**Tuesday 11/14/2017 23:59:59 EST**
Phase 3	**Q1, Q2, Q3, Q4**	**Monday 11/13/2017 00:00:01 ET**	**Sunday 11/26/2017 15:59:59 EST**	**-**
	Live Test Q1, Q2, Q3, Q4	**Sunday 11/26/2017 18:00:01 ET**	**Sunday 11/26/2017 23:59:59 EST**	**Tuesday 11/28/2017 23:59:59 EST**

Hints for the live test

- The request pattern will differ for Phase 2 and the live test so your solution should handle all types of load.
- Monitor your system efficiently during the live test to recover in case of a system crash.
- Your Phase 2 budget should take into account the cost for the live test.

Upcoming Deadlines

- Quiz 9: Unit 4 - Modules 16 & 17

  Due: Friday, Nov 3 2017 11:59PM Pittsburgh

- Project 2.3: Serverless Functions

  Due: Thursday, Nov 5 2017 11:59PM Pittsburgh

- Project 4.1: Batch Processing with MapReduce

  Due: Sunday, Nov 5 2017 11:59PM Pittsburgh

- Team Project - Phase 1

  Due: Tuesday, Oct31 2017 11:59PM Pittsburgh

- Team Project - Phase 2 - ongoing

Questions?