15-319 / 15-619
Cloud Computing

Recitation 7

February 23rd, 2016

Overview

- **Administrative issues**
  Office Hours, Piazza guidelines
- **Last week's reflection**
  Project 2.3, OLI unit 3 module 10, 11, 12, Quiz 5
- **This week's schedule**
  - Quiz 6 - Friday, February 26th (Module 13)
  - Project 3.1 - Sunday, February 28th
- **Twitter Analytics: The 15619 Project**

Last Week : A Reflection

- Content
  - Unit 3 - Modules 10, 11 and 12:
    Virtualizing Resources on the Cloud
  - Quiz 5 - completed
- You wrote your own cache!
  - Data locality
  - Adaptive caching strategy
  - Asynchronous vs synchronous

Project 2.3 Grading

Reminder!

- Manual Grading:
  - 20 Points are for the code, we will evaluate
    - Solution (15)
    - Style (5)

Project 2 Reflection

Topics explores in P2:

- AWS/Azure APIs
- AutoScaling
- Load balancing strategies
- Mitigating failure (health check)
- Caching strategy in multi-tiered applications

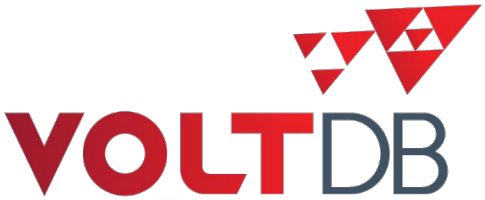
This Week: Content

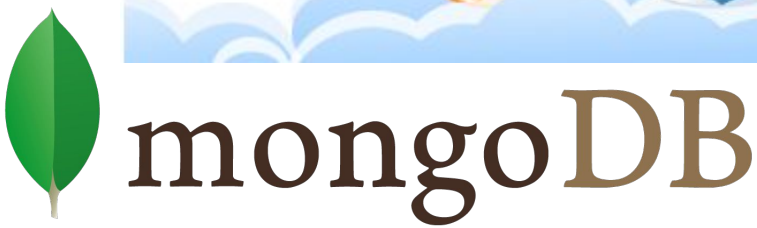
UNIT 3: Virtualizing Resources for the Cloud

- Module 10: Resource virtualization (memory)
- Module 11: Resource virtualization (I/O)
- Module 12: Case Study
- **Module 13: Storage and network virtualization**
  - Software Defined Data Center (SDDC)
  - Software Defined Networking (SDN)
    - Device virtualization (Router and NIC virtualization)
    - Link virtualization (Bandwidth/datapath virtualization)
  - Software Defined Storage (SDS)
    - IOFlow
- **Quiz 6, Friday, February 26th**

Project 3 - Storage

- Storage in the cloud (It's Hot!!!)





Project 3 Weekly Modules

- P3.1: Files, SQL and NoSQL
- P3.2: Replication and sharding
- P3.3: Consistency models
- P3.4: Social network with heterogeneous backend storage
- P3.5: Data warehousing and OLAP

Project 3.1 Overview

P3.1: Files, SQL and NoSQL:

- Run basic Unix commands like grep, awk etc to extract certain data from given datasets in flat files
- Use relational databases (MySQL)
  - load data, run basic queries
- Vertical scaling in storage technologies
  - Magnetic vs SSD
  - Instance types
- Use a NoSQL database (HBase)
  - load data, run basic queries

Flat Files

- Flat files, plain text or binary
  - Ex: A comma-separated ‘csv’ file.

    Ryan, 15619, A

    Wei, 15319, A
- Lightweight
- Flexible
- Performing analysis on data in files can be expensive
- Insert is expensive due to having to seek
- Managing a large number of files with different data types can become complex
- …

Databases

- A collection of organized data
- Database management system (DBMS)
  - Interface between user and data
  - Store/manage/analyze data
- Relational databases
  - Based on the relational model (schema): MySQL
- NoSQL Databases
  - Unstructured/semi-structured
  - DynamoDB, Cassandra, HBase

Databases

- Advantages
  - Logical and physical data independence
  - Concurrency control and transaction support
  - Easy and convenient querying (e.g. SQL)
  - ...

- Disadvantages
  - Cost (computational resources, fixed schema)
  - Maintenance and management
  - Complex and time-consuming to design
  - ...

Files vs. Databases

- Compare flat files to databases

- Think about:
  - What are the advantages and disadvantages of using flat files or databases?
  - In what situation would you use a flat file or a database?
  - How to design your own database? How to manipulate and query data in a database?

Flat File Tasks

- Analyze million_songs_metadata.csv and millions_songs_sales_data.csv

- Answer questions in runner.sh
  - Use tools such as awk, grep, …
  - Similar to what you did in Project 1.1

- Merge csv files by joining on a common field

MySQL Tasks

- Prepare table
  - The script to create the table is already provided
  - Find a way to load the data properly into MySQL
- Use MySQL queries to answer questions
  - Learn JDBC
  - Complete MySQLTasks.java
  - Aggregate functions, joins
- Learn how to use indexes to improve performance

Storage Vertical Scaling Tasks

Use the sysbench to benchmark for the following 4 scenarios:

Scenario	Instance Type	Storage Type
1	t1.micro	EBS Magnetic Storage
2	t1.micro	EBS General Purpose SSD
3	m3.large	EBS Magnetic Storage
4	m3.large	EBS General Purpose SSD

Performance Benchmarks

- Run sysbench - prepare data
  - remember to change to mounted directory
  - use the prepare option to generate the data

- Experiments
  - run sysbench with different storage systems and instance types
  - run sysbench multiple times

- Answer questions in runner.sh

HBase (NoSQL) Tasks

- Launch an EMR cluster with HBase installed.
- Follow the write-up to download and load the data into HBase.
- Try different querying commands in the HBase shell.
- Complete HBaseTasks.java using HBase Java APIs.

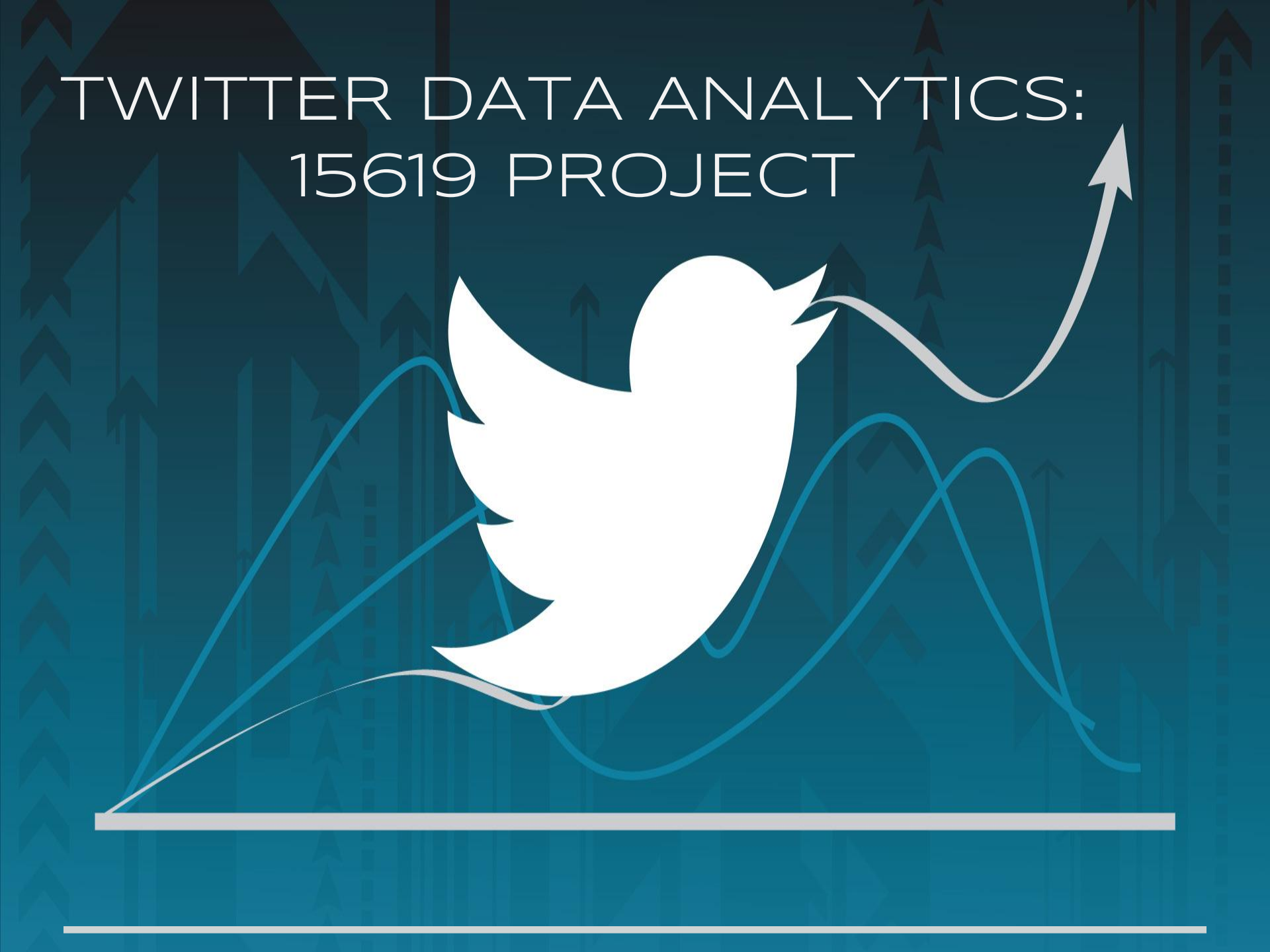
P3.1 Reminders

- Tag your resources with: Key: Project, Value: 3.1
  - manually tag your spot instances

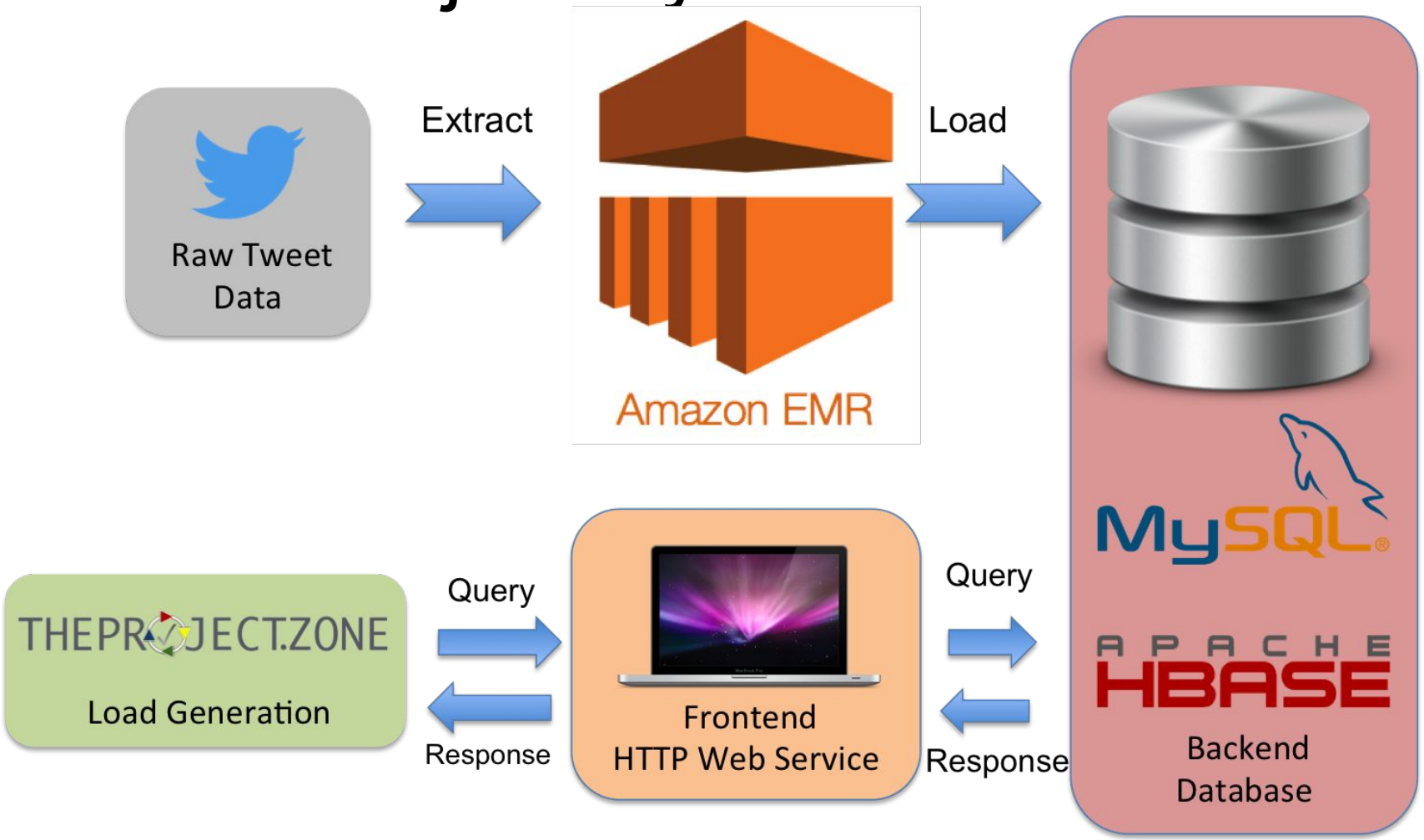
- You can save a copy of your runner.sh if you want to take a break and work on it later.

- Make sure to terminate the instance after finishing all questions and submitting your answers.

TWITTER DATA ANALYTICS: 15619 PROJECT



15619 Project System Architecture

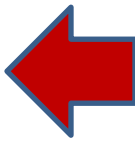


- Web server architectures
- Dealing with large scale real world tweet data
- HBase and MySQL optimization

15619 Project

- Phase 1:
  - Q1
  - Q2 (MySQL **AND** HBase)

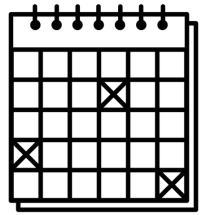
CHECK YOUR AWS ACCOUNT AND TEAM INFO



- Phase 2
  - Q1
  - Q2 & Q3 (MySQL **AND** HBase)

- Phase 3
  - Q1
  - Q2, Q3 & Q4 (MySQL **OR** HBase)

15619 Project Time Table



Phase (and query due)	Start	Deadline	Code and Report Due
Phase 1 Part 1 • Q1, Q2	Thursday 02/25/2016 00:00:01 EST	Wednesday 03/16/2016 23:59:59 EDT	Thursday 03/17/2016 23:59:59 EDT
Phase 2 • Q1, Q2, Q3	Thursday 03/17/2016 00:00:01 EDT	Wednesday 03/30/2016 15:59:59 EDT	
Phase 2 Live Test (Hbase/MySQL) • Q1, Q2, Q3	Wednesday 03/30/2016 18:00:01 EDT	Wednesday 03/30/2016 23:59:59 EDT	Thursday 03/31/2016 23:59:59 EDT
Phase 3 • Q1, Q2, Q3, Q4	Thursday 03/31/2016 00:00:01 EDT	Wednesday 04/13/2016 15:59:59 EDT	
Phase 3 Live Test • Q1, Q2, Q3, Q4	Wednesday 04/13/2016 18:00:01 EDT	Wednesday 04/13/2016 23:59:59 EDT	Thursday 04/13/2016 23:59:59 EDT

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)**

15619 Project Phase 1

- Two queries
  - Q1: Pure front end
  - Q2: ETL + back end + front end, do both MySQL (relational DBMS) and HBase (NoSQL)
- Grading
  - Submit on TPZ, you will get several numbers:
    - Error Rate, Correctness and RPS
  - Higher RPS, higher correctness, lower error rate ⇒ higher grade
  - Q1 is 25% of phase 1, Q2 MySQL is 25% of phase 1, Q2 HBase is 25% of phase 1, report is 25% of phase 1.
  - Phase 1 bonus:
    - 5% bonus if you succeed with Q1 in one week, 5% bonus if you a top 3 team on the scoreboard.

15619 Project, Phase 1, Q1

- **Step 1:** Compare different front-end frameworks
- **Step 2:** Deploy the front-end
- **Step 3:** Perform decryption of a secret message
  - You have the secret key (a big int)
  - We send you the secret message and the encryption key (another big int)
  - Perform the calculation of the secret key and encryption key (GCD) and decrypt the secret message

Pure front end, no database needed. Need to consider scaling

Meet the target of Q1 by the first week and earn a bonus!

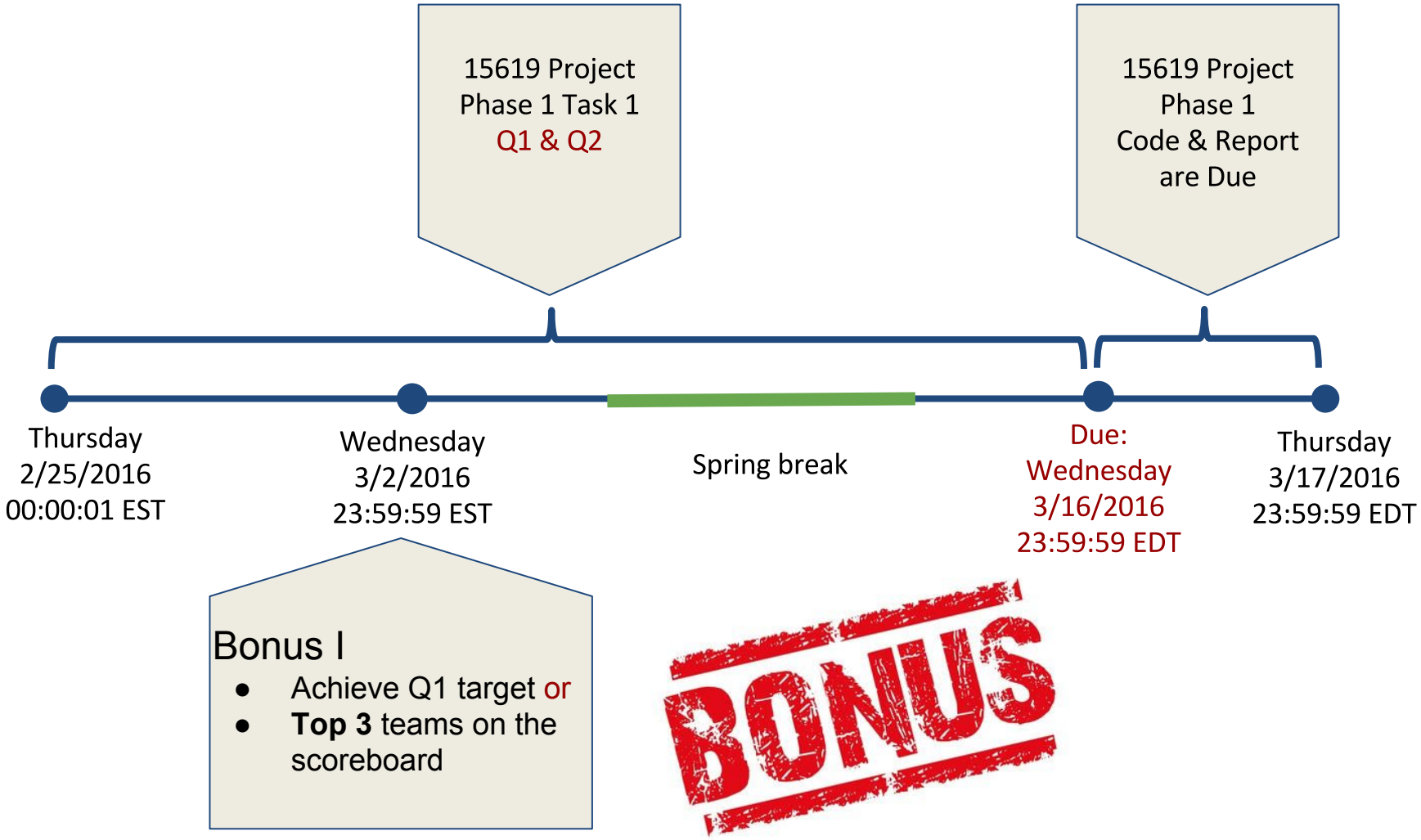
15619 Project, Phase 1, Q2

- **Step 1:** Extract tabular data from raw tweets
  - Input file: JSON Tweets (approx. 1 TB)
  - Consider using a MapReduce Job for ETL
    - ETL is expensive and there's the potential for errors, so plan carefully, test on smaller data sets
    - When you're stuck, look at the **reference files**
    - Start early, or no time to optimize the backend
- **Step 2:** Load the data into HBase **and** MySQL (both!)
- **Step 3:** Deploy
  - a web service for handling HTTP requests, responds with data from the backend

Higher throughput = More points

Winner gets grades, fame (?), job (?)

15619 Project Phase 1 Deadlines



Hints

- Read the write-up carefully (read more than once)
- You can test only if you have a **front end**
- ETL has many corner cases, can be time consuming and expensive
  - Start early (from the first day), your backend will be meaningless if you have incorrect data
  - Reference file is your friend
- Big data challenge will easily eat up your time and money if you are careless. Think, calculate, & test before you launch an EMR cluster with 20 machines

Reminder

- You have a total budget of $40 for Phase 1
- Your system should not cost more than $0.85 per hour, this includes (see write-ups for details):
  - EC2 on demand instance cost
    - even if you use spot instances, we will calculate your cost using the on demand instance price
  - Storage cost
  - ELB cost

Upcoming Deadlines

- Quiz 6: Unit 3 - Storage and network virtualization

  Due: 2/26/2016 11:59PM Pittsburgh

- Project 3.1: Files vs Databases

  Due: 2/28/2016 11:59PM Pittsburgh

- 15619 project: Phase 1 bonus

  Due: 3/2/2016 11:59PM Pittsburgh