15-319 / 15-619
Cloud Computing

Recitation 11

March 29th, 2016

Overview

- **Administrative issues**
  - Tagging, 15619Project, project code
- **Last week's reflection**
  - Project 3.4
  - Quiz 9
- **This week's schedule**
  - Project 3.5
  - Unit 5 - Module 18
  - 15619Project Phase 2
  - Quiz 10
- **Twitter Analytics: The 15619Project**

Reminders

- Monitor AWS expenses regularly and tag all resources
  - Check your bill (Cost Explorer > filter by tags).
- Piazza Guidelines
  - Please tag your questions appropriately
  - Search for an existing answer first
- Provide clean, modular and well documented code
  - Large penalties for not doing so.
  - **Double check** that your code is submitted!! (verify by downloading it from TPZ from the submissions page)
- Utilize Office Hours
  - We are here to help (but not to give solutions)
- Use the team AWS account and tag the 15619Project resources carefully

Project 3.4 FAQs

Problem 1: Database connection issues

- Network issues / Security groups/ version consistent/typo
- For HBase:@2294

Problem 2: Loading, sorting and processing the data correctly.

- MySQL “order by” is case-insensitive
- Java String sorting is case-sensitive
- Or use utf8mb4_bin collation for MySQL, which is case-sensitive

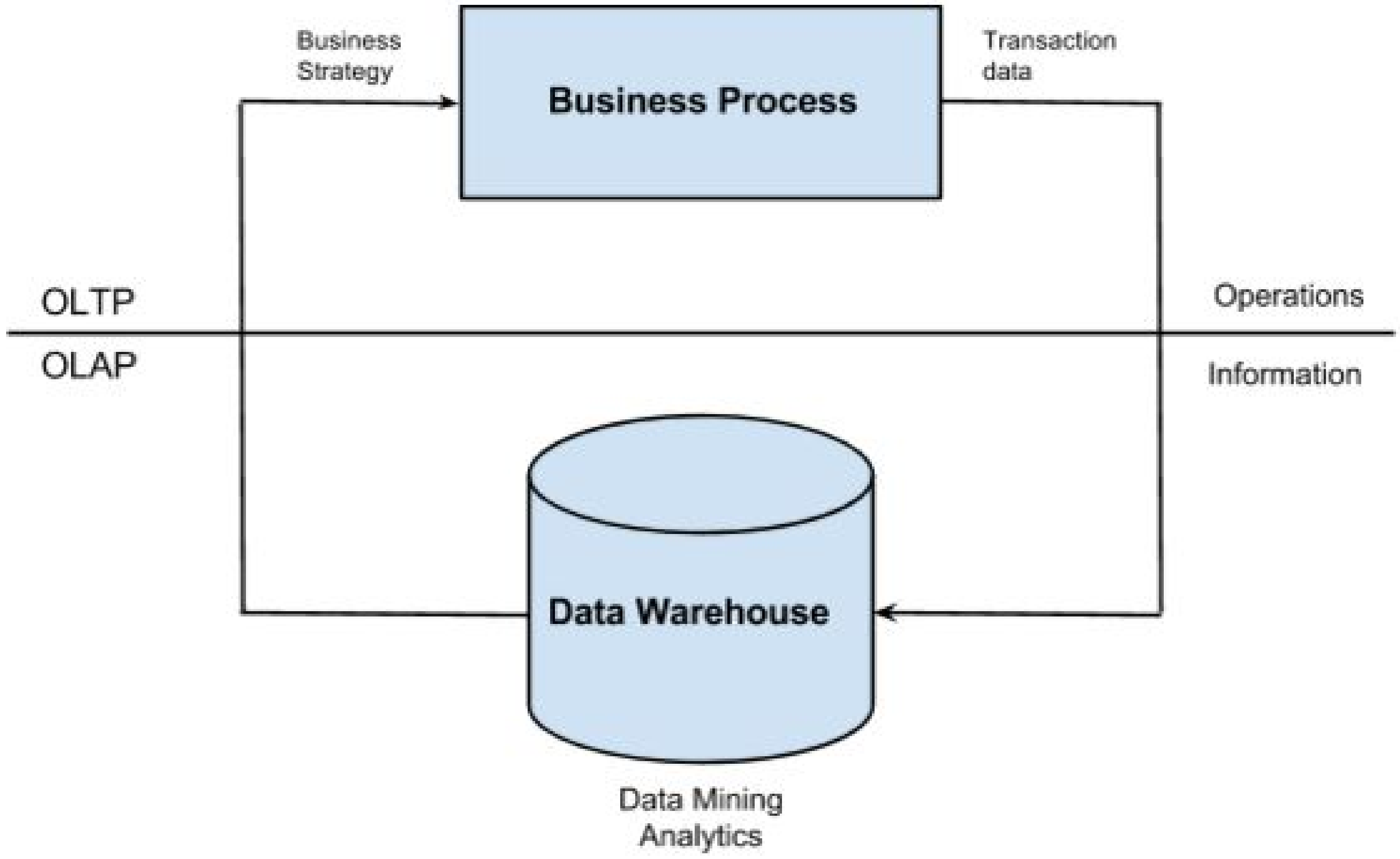
This week: Project 3.5

- P3.1: Files, SQL and NoSQL
- P3.2: Partitioning (Sharding) and Replication
- P3.3: Consistency in Distributed Key-Value Stores
- P3.4: Social network with Heterogeneous Backends
- **P3.5: OLAP with Cloud Data Warehousing**

P3.5: Background

- Carnegie Eagle(CE), a supermarket chain wishes to expand their business further into other markets and wants a decision support system
- The CTO of CE wants you to analyze various data warehouses and pick the best one for the job
- The CTO subscribes to the “Hottest Data Warehouses” weekly and decides she wants you to analyze the following:
    - Hive
    - Impala
    - Redshift

OLTP vs OLAP


Business Strategy
Business Process
Transaction data
OLTP
Operations
OLAP
Information
Data Warehouse
Data Mining
Analytics

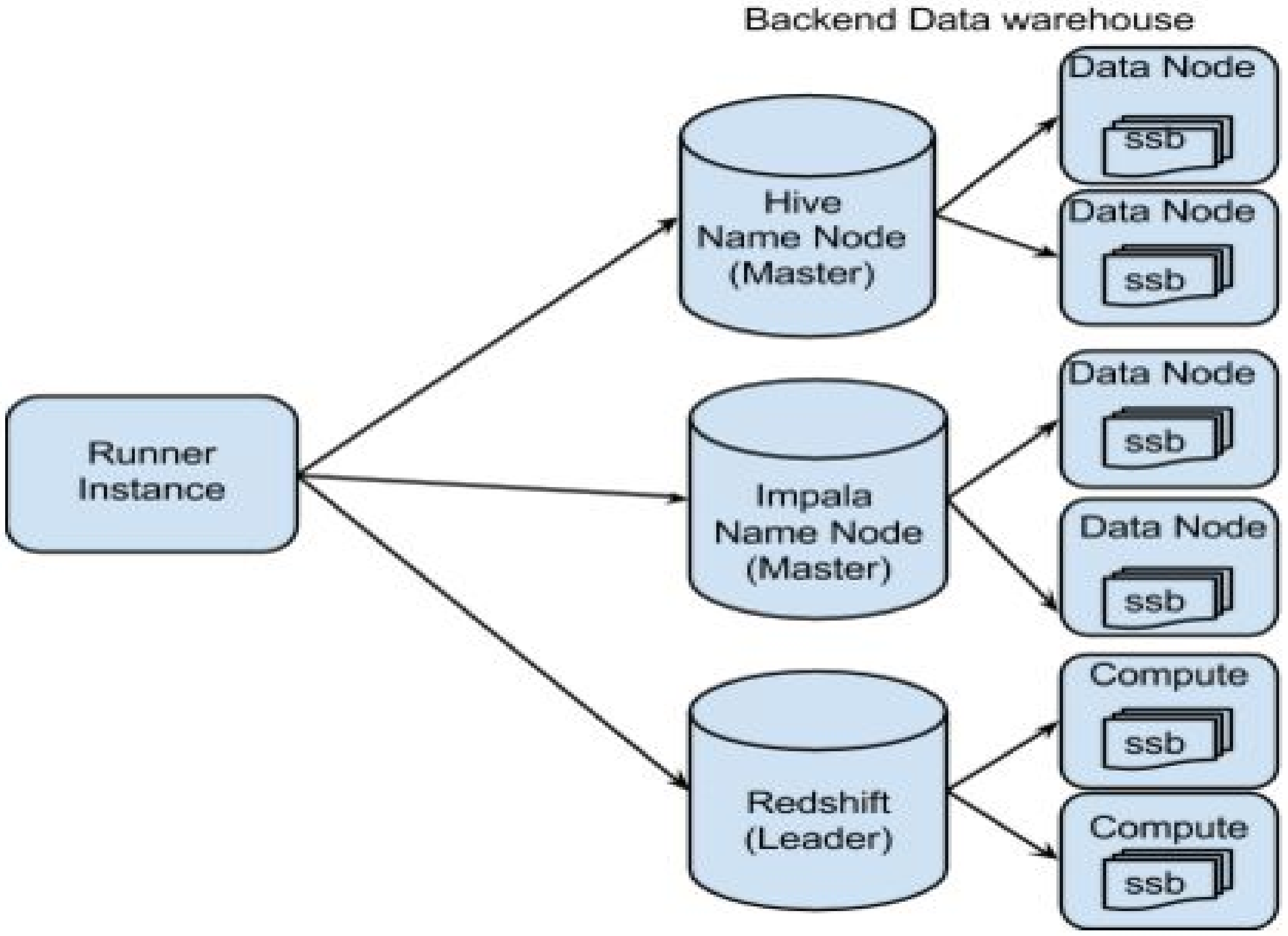
Data warehousing and OLAP

- OLAP (Online Analytical Processing) queries deals with historical/archived data
- OLAP warehouses optimized for reads and aggregations
- Rarely perform updates
- Tables in OLAP are denormalized as compared to normalized tables in OLTP (Online Transaction Processing)
- Data warehouses tuned for high throughput since they process large amounts of data
- OLTP databases are more tuned for smaller updates and lower latency

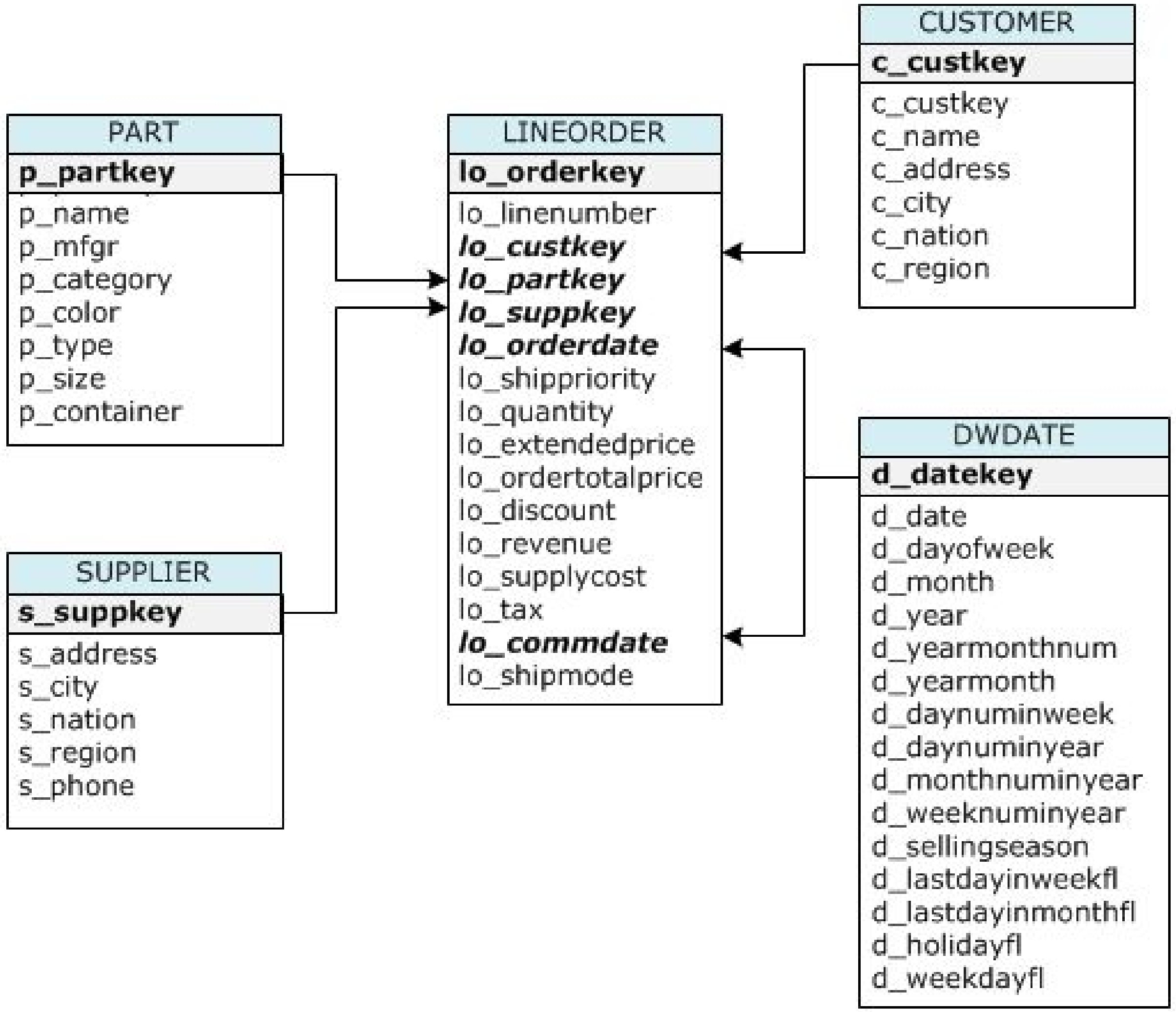
OLAP data warehouses

- Hive
  - Built on Hadoop and provides a SQL-like interface to query the data
  - Translates user SQL query to MapReduce jobs
- Impala
  - Based on Hadoop as well and provides a SQL-like interface similar to Hive
  - Uses its own engine to translate user queries and directly access data on the cluster (hence lower latency)
- Redshift
  - Data-warehouse-as-a-Service provided by Amazon Web Services
  - Used for real time analytics

Data Warehousing Benchmark



OLAP Star Schema



Data Warehousing Benchmark

- Hive
  - No optimization required
  - Follow the instructions and you are done

- Impala and Redshift
  - Load data, execute unoptimized queries
  - Optimize table schemas and/or queries
  - The evaluation will be on both correctness and response time

Notes

- Hive
  - May take an hour to run, be patient.
- Impala
  - Some unoptimized queries may throw exceptions.
- Redshift
  - Be aware of the high expenditure! Think before you start.

Module to Read

- UNIT 5: Distributed Programming and Analytics Engines for the Cloud
  - Module 18: Intro to distributed programming for the Cloud
  - Module 19: Distributed analytics engines: MapReduce
  - Module 20: Distributed analytics engines: Spark
  - Module 21: Distributed analytics engines: GraphLab
  - Module 22: Message Queues and Stream Processing

Distributed Programming

- **Taxonomy of Programs:**
  - Sequential
  - Concurrent
  - Parallel

- **Challenges in programming the cloud:**
  - Scalability
  - Communication overhead
  - Heterogeneity
  - Synchronization
  - Fault Tolerance
  - Scheduling

Upcoming Deadlines

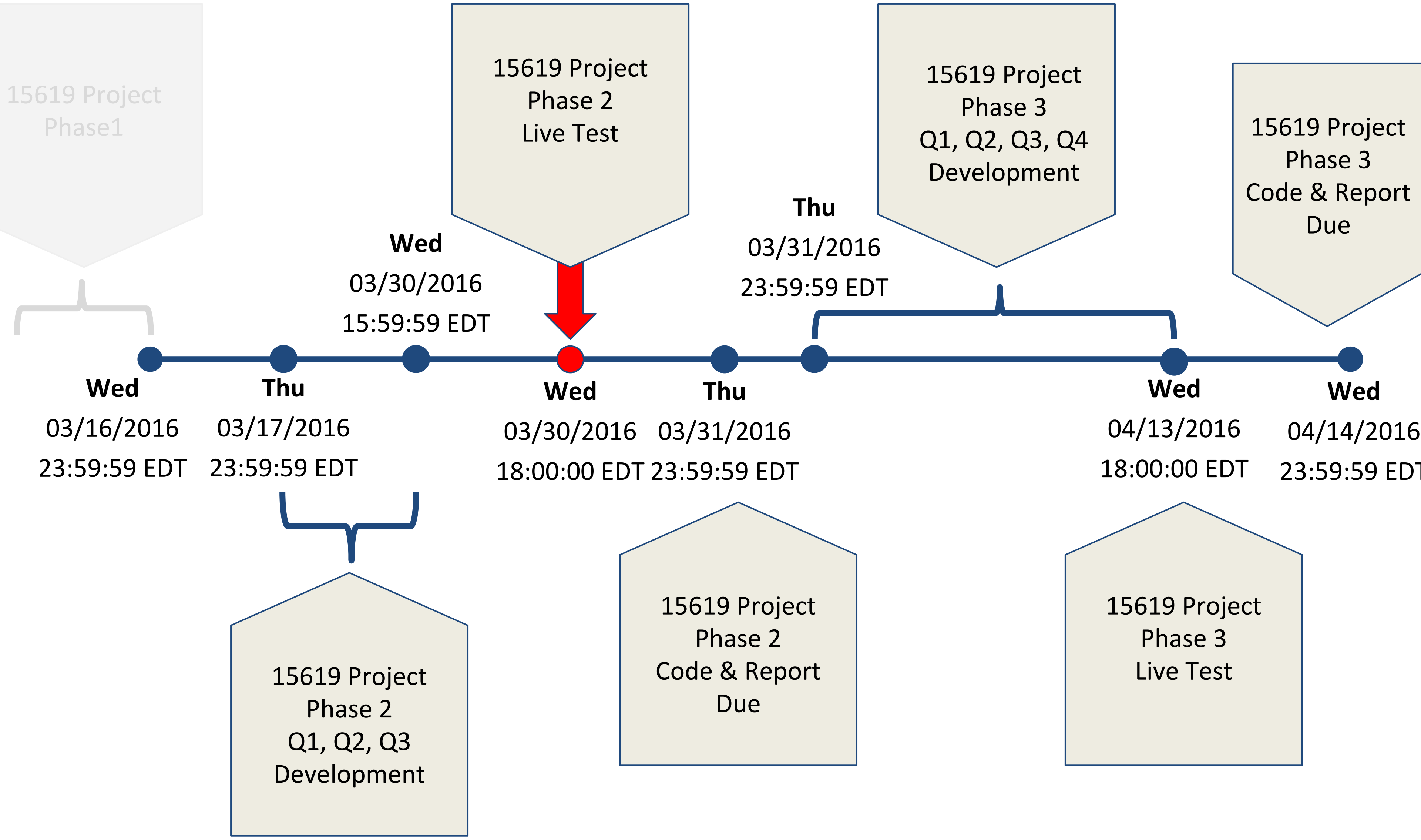
- Quiz 10 : Unit 5 - Module 18
  - Due: 04/01/2016 11:59 PM Pittsburgh
- Project 3.5 : Data warehousing and OLAP
  - Due: 04/03/2016 11:59 PM Pittsburgh
- 15619Project : Phase 2
  - Live-test DNS due: 03/30/2016 3:59 PM Pittsburgh
  - Code and report due: 03/31/2016 11:59 PM Pittsburgh

Busy Weeks Coming Up!

Wednesday	Thursday	Friday	Sunday
Wednesday 03/30/2016 18:00:01 E**<u>D</u>**T ● Phase 2 Live Test ● Submit DNS by 15:59 pm EDT	Thursday 03/31/2016 23:59:59 E**<u>D</u>**T ● Phase 2 Code & Report Due	Friday 04/01/2016 23:59:59 E**<u>D</u>**T ● Quiz 10	Sunday 04/03/2016 23:59:59 E**<u>D</u>**T ● P3.5 Due
Wednesday 04/13/2016 20:00:01 E**<u>D</u>**T ● Phase 3 Live Test	Thursday 04/14/2016 23:59:59 E**<u>D</u>**T ● Phase 3 Code & Report Due		Sunday 04/17/2016 23:59:59 E**<u>D</u>**T ● P4.2 Due

TWITTER DATA ANALYTICS: 15619 PROJECT

15619 Project Phase 2 Deadlines


15619 Project Phase1
Wed 03/16/2016 23:59:59 EDT
Thu 03/17/2016 23:59:59 EDT
15619 Project Phase 2 Q1, Q2, Q3 Development
Wed 03/30/2016 15:59:59 EDT
15619 Project Phase 2 Live Test
Wed 03/30/2016 18:00:00 EDT
Thu 03/31/2016 23:59:59 EDT
15619 Project Phase 2 Code & Report Due
Thu 03/31/2016 23:59:59 EDT
15619 Project Phase 3 Q1, Q2, Q3, Q4 Development
Wed 04/13/2016 18:00:00 EDT
15619 Project Phase 3 Live Test
Wed 04/14/2016 23:59:59 EDT
15619 Project Phase 3 Code & Report Due

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

15619Project Report Tips

- Look at the feedback of your Phase 1 report!
  - Feedback and suggestions are provided
- Learn to document your system design and correctly reason about it
  - This will help you improve your system
- Use tools to gather evidence to identify issues in your system's performance
  - Check previous recitations for hints
- General Tips
  - Create an AMI to automate your processes
  - Evaluate which HBase distribution to use and which parameters to configure

Live Test!

- Live Test!
  - Warmup, Q1, Q2, Q3, Mixed Q1-Q3
    - Each for 30 min
  - Submit your team’s web service DNS
    - Both HBase and MySQL
    - Two DNS before 4:00 PM

Phase 2 Live Test

HBase Live Test

Time	Value	Target	Weight
6:00 pm - 6:30 pm	Warm-up (Q1 only)	-	0%
6:30 pm - 7:00 pm	Q1	27000	5%
7:00 pm - 7:30 pm	Q2	10000	5%
7:30 pm - 8:00 pm	Q3	6000	10%
8:00 pm - 8:30 pm	Mixed Reads(Q1,Q2,Q3)	6000/3000/2000	5+5+5 = 15%

Half Hour Break

MySQL Live Test

Time	Value	Target	Weight
9:00 pm : 9:30 pm	Warm-up	-	0%
9:30 pm - 10:00 pm	Q1	27000	5%
10:00 pm - 10:30 pm	Q2	10000	10%
10:30 pm - 11:00 pm	Q3	6000	10%
11:00 pm - 11:30 pm	Mixed Reads (Q1,Q2,Q3)	6000/3000/2000	5+5+5 = 15%

Tips for Live Test

- Watch your budget: $60 = Phase 2 + Live Test
- Preparing for the live test
  - Submit two URLs, MySQL & HBase.
  - Budget limited to $.85/hr for MySQL and HBase web service separately.
    - No extra machines during live test except the cluster (FE, DB). Tag all your instances!
  - Need to have all Q1-Q3 running at the same time.
  - Queries will be mixed.
  - Do not use spot instances.
  - Teams need to monitor the whole live test.
  - FULLY warm up ELB and EBS.
  - You can terminate HBase machines immediately after the HBase live test.

Phase 3

- One last query (Q4)
  - No ETL!
  - Serving write requests
  - Front end caching will not work during the live test
- **Live Test!**
    - **Warmup, Q1, Q2, Q3, Q4, Mixed Q1-Q4**
      - **Each for 30 min**
    - **Choose HBase or MySQL**
      - **Submit One DNS**

Query 4: Tweet Server

There are five different parameters in the request URL for a request to /q4.

- tweetid (tweet ID)
- op (operation type)
- seq (sequence number)
- fields (comma separated fields involved in the request)
- payload (comma separated payload in Base64)

Execute the requests of a tweetid by the seq#

Query 4: Tweet Server

field	type	example
tweetid	long int	15213
userid	long int	156190000001
username	string	CloudComputing
timestamp	string	Mon Feb 15 19:19:57 2016
text	string	Welcome to P4!#CC15619#P3
hashtag	comma separated string	CC15619,P3
ip	string	128.2.217.13
coordinates	string	-75.14310264,40.05701649
repliedby	comma separated userid	156190000001,156190000002,156190000003
reply_count	long int	3
mentioned	comma separated userid	156190000004,156190000005,156190000006
mentioned_count	long int	3
favoritedby	comma separated userid	156190000007,156190000008,156190000009
favorite_count	long int	3
useragent	string	Mozilla/5.0 (iPhone; CPU iPhone OS ...)
filter_level	string	PG-13
lang	string	American

Query 4: Tweet Server

- SET Request `/q4?tweetid=15213&op=set&seq=1&fields=repliedby,reply_count&payload=MzM2NDE5MzE2NjUsMTc0Mjg5OTA1OTksOTQ5MDczNzc5NjQsMzkzMjIxMzU4NjQsMTg0NDA4MDg5NTUsNTE2MjU1MzMxOTgsOTI4MzA3NTgwNzQ=,Nw==`

- Response

```
TEAMID,TEAM_AWS_ACCOUNT_ID\n
success\n
```

Query 4: Tweet Server

- GET Request `/q4?tweetid=15213&op=get&seq=2&fields=repliedby,reply_count&payload=`

- Response

```
TEAMID,TEAM_AWS_ACCOUNT_ID\n
MzM2NDE5MzE2NjUsMTc0Mjg5OTA1OTksOTQ5MDczNzc5NjQsM
zkzMjIxMzU4NjQsMTg0NDA4MDg5NTUsNTE2MjU1MzMxOTgsOT
I4MzA3NTgwNzQ=\n
Nw==\n
```

What’s due soon?

- Report at the end of Phase 2
  - **Submission by 23:59 ET (Pittsburgh) Thur 03/31**
  - Make sure you highlight failures and learning
  - If you didn’t do well, explain why
  - If you did, explain how
  - Cannot begin to stress how critical this is!!!!
- Phase 3 Development
  - **Submission by 16:59 ET (Pittsburgh) Wed 04/13**
    - **Live Test from 6 PM to 10 PM ET**
  - Fix Q1 - Q3 if you did not go well
  - New query Q4.
  - Heads up: Phase 3 counts for **60%** of 15619Project grade

General Tips

- Don’t blindly optimize for every component, identify the bottlenecks using fine-grained profiling.
- Use caches wisely: caching in HBase and MySQL is obviously important, but front-end cache will most likely fail during the Live test.
- Get the whole picture of the database you are using, don’t just Google and adopt “HBase/MySQL optimization techniques” blindly.
- Review what we have learned in previous project modules
    - Scale out
    - Load balancing
    - Replication and sharding

Upcoming Deadlines

- Quiz 10 : Unit 5 - Module 18
  - Due: 04/01/2016 11:59 PM Pittsburgh
- Project 3.5 : Data warehousing and OLAP
  - Due: 04/03/2016 11:59 PM Pittsburgh
- 15619Project : Phase 2
  - Live-test DNS due: 03/30/2016 3:59 PM Pittsburgh
  - Code and report due: 03/31/2016 11:59 PM Pittsburgh