CS15-319 / 15-619
Cloud Computing

Recitation 10

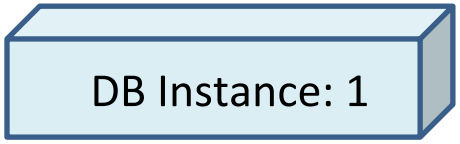
October 28th and October 30th , 2014

Announcements

- Encounter a general bug:
  - Post on Piazza
- Encounter a grading bug:
  - Post Privately on Piazza
- Don’t ask if my answer is correct
- Don’t post code on Piazza
- Search before posting
- Post feedback on OLI
- **OLI does not show timer for Quiz! You have to maintain your own timer!**

Project 3, Module 3 Reflections

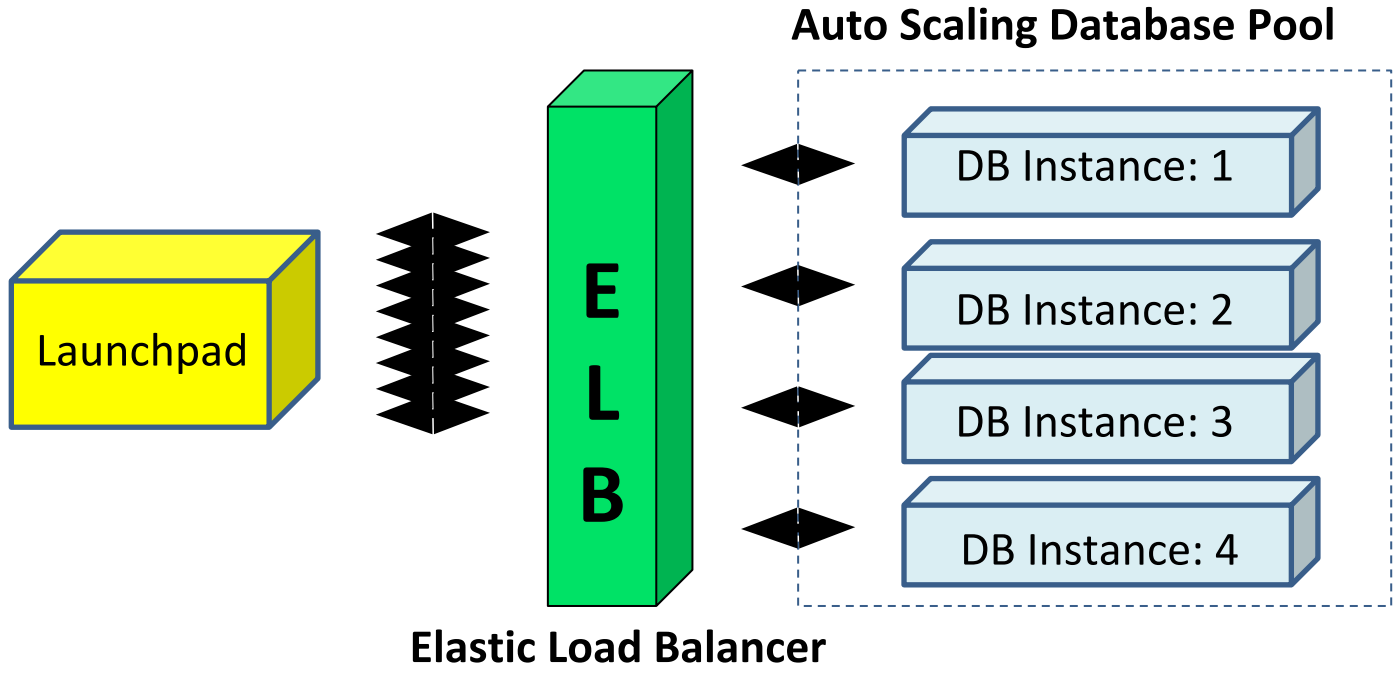
- Cost is different between running a fixed number of instances vs autoscaling
- Custom Metrics



  - Queries per Second = Queries / (Uptime1 – Uptime2)
  - Transactions Per Second (TPS)
    - Factor in the two queries used to find the number of queries and uptime.
    - *sysbench* transaction equals to 16 queries
  - Send TPS information to CloudWatch using API

Project 3, Module 3 Reflections

- Horizontal Scaling of Databases



- Horizontal Scaling of SQL Database is not easy to implement.

Project 3, Module 3 Problems

- The **/etc/init.d** directory contains a number of start/stop scripts for various services on your system.
- **/etc/rc.local** file runs after all other init level scripts have run. You can put commands that you want to have issued upon startup.
- **cron:** enable users to execute commands or scripts automatically at a specified time/date.

Project 3, Module 3 Problems

@1640 - Unable to SSH to AMI

- When copying mysql backup, it should go to /storage/mountpoint;
- Some students copied to root folder, and when they executed ‘chown’, it caused mysql to own the root folder, thus making ubuntu user unable to access the folder that contains the key (*.pem) file.

Module to Read

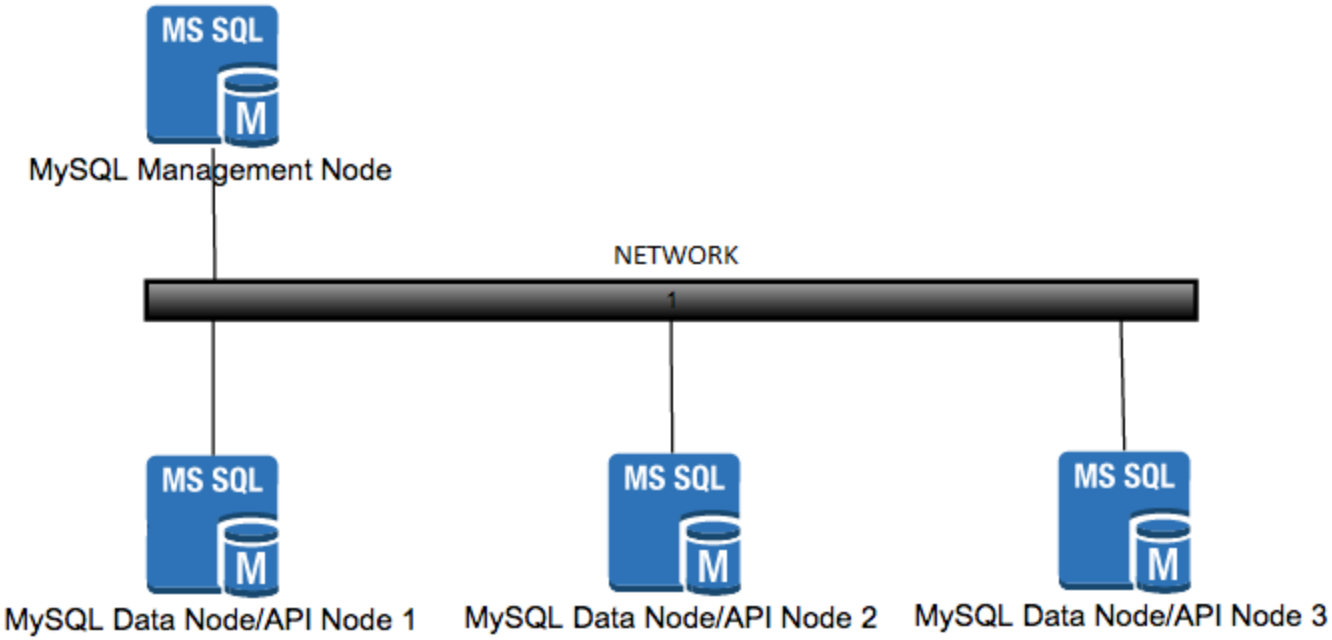
- UNIT 4: Cloud Storage
  - Module 12: Cloud Storage
  - Module 13: Case Studies: Distributed File Systems
  - Module 14: Case Studies: NoSQL Databases
  - Module 15: Case Studies: Cloud Object Storage
  - Quiz 4: Cloud Storage

Project 3

- Files vs. Databases
  - File vs. Database
- Vertical Scaling in Databases
  - Vertical Scaling
- Horizontal Scaling in Databases
  - Horizontal Scaling
- Working with NoSQL: DynamoDB / Hbase
  - Write Scalability
  - DynamoDB vs. HBase

Sharding

- Horizontal partitioning (i.e. partitioning of the rows)



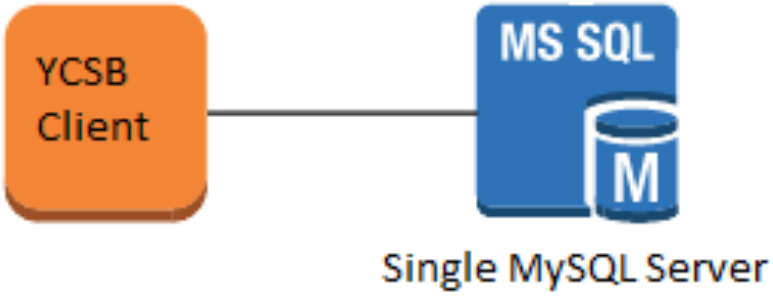
Sharding

- Steps

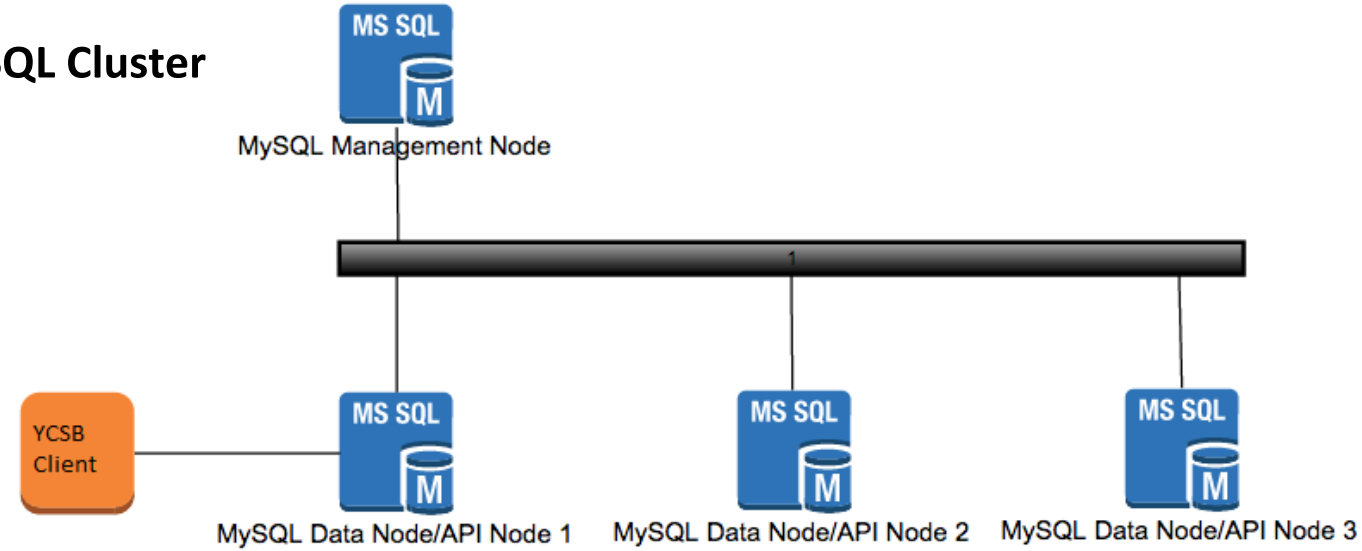
1. Configure management node;
2. Configure database nodes;
3. Testing.

YCSB Tests

1st: Single Node



2nd: MySQL Cluster



Upcoming Deadlines

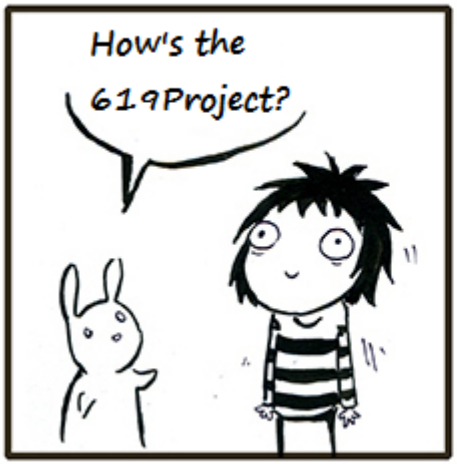
- Project 3:

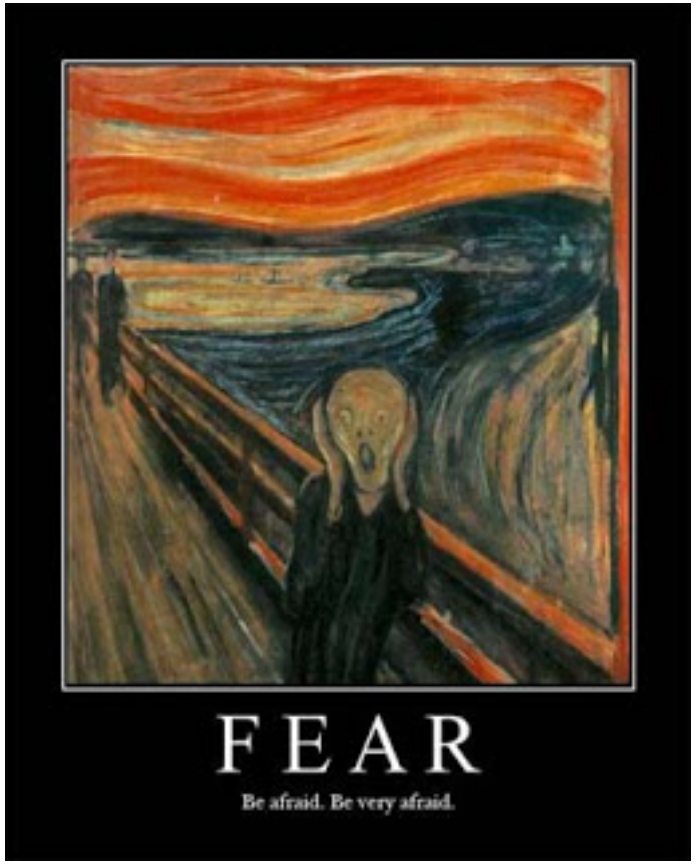
Working with NoSQL: DynamoDB / HBase (Gradebook) (Learning Dashboard)		
Write Scalability	Checkpoint	**Available Now** Due 11/2/14 11:59 PM
DynamoDB vs. HBase	Checkpoint	Not yet assigned Due date TBD by instructor

- Unit 4:

Module 15: Case Studies: Cloud Object Storage (Gradebook) (Learning Dashboard)		
Quiz 4: Cloud Storage	Checkpoint	**Available Now** Due 10/30/14 11:59 PM

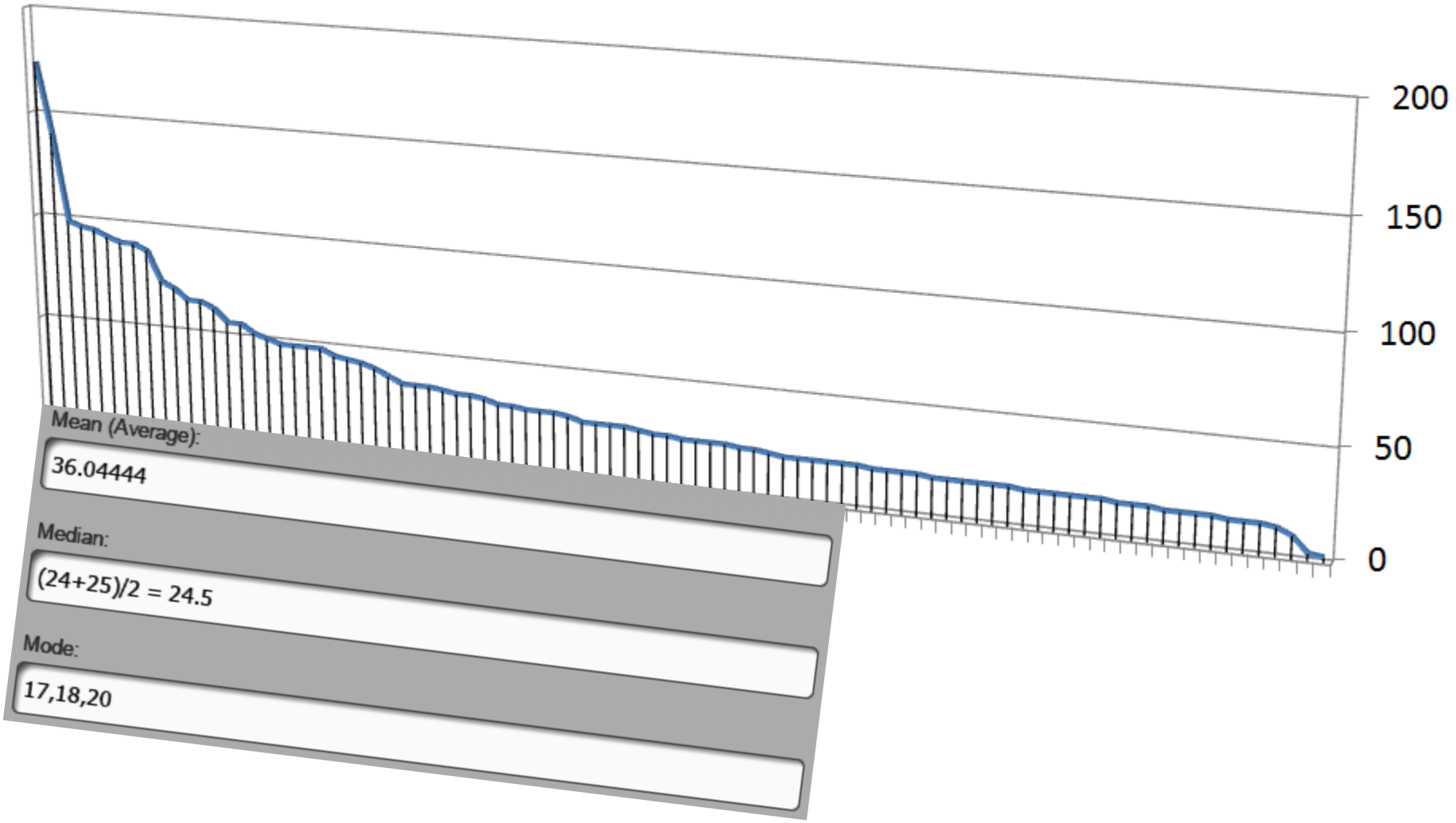
TWITTER ANALYTICS:
THE 619 PROJECT


How's the
619Project?


FEAR
Be afraid. Be very afraid.

Phase Score	Q1 Best	Q1 Correctness	Q1 HTTP Error Rate	Q1 Throughput	Q1 Latency	Q2 Best	Q2 Correctness	Q2 HTTP
2	11	100.00	0.00	1670.7	58	–	–	–
3	19	100.00	0.00	2827.8	33	0	0	100.00
10	47	100.00	0.00	7064.8	13	2	92.00	0.01
13	67	100.00	0.00	10001.0	9	–	–	–
14	73	100.00	0.00	10887.0	8	0	0	100.00
14	72	100.00	0.00	10871.4	8	–	–	–
14	71	100.00	0.00	10576.6	9	–	–	–
15	74	100.00	0.00	11058.0	8	2	46.00	0.04
15	78	100.00	0.00	11649.2	8	0	0	0.17
15	78	100.00	0.00	11671.2	8	–	–	–
15	75	100.00	0.00	11308.8	8	–	–	–
16	84	100.00	0.00	12542.2	7	0	0	0.27
16	83	100.00	0.00	12487.0	7	–	–	–
16	83	100.00	0.00	12492.0	7	–	–	–
17	84	100.00	0.00	12664.5	7	1	7.00	0.01
17	87	100.00	0.00	13037.0	7	0	0	100.00
17	89	100.00	0.00	13365.9	7	–	–	–
17	88	100.00	0.00	13262.6	7	–	–	–
17	86	100.00	0.00	12835.6	7	–	–	–
17	84	100.00	0.00	12632.5	7	–	–	–
18	73	100.00	0.00	10959.8	8	9	44.00	12.67
18	79	100.00	0.00	11807.1	8	4	46.00	0.02
18	84	100.00	0.00	12564.3	7	3	66.00	0.02
18	93	100.00	0.00	14009.3	6	0	0	0.03
18	90	100.00	0.00	13505.1	6	0	0	100.00
18	90	100.00	0.00	13487.7	7	0	0	100.00
19	83	100.00	0.00	12491.5	7	8	72.00	13.12

Phase 1 Score Distribution



Congratulations Team FDU

- The early bird gets the worm

Team	Phase Score	*Q1 Best*	Q1 Throughput	*Q2 Best*
FDU	174	127	19082.7	218
xnoobs	140	121	18174.2	151
WaterPig	98	111	16634.1	99
spartans	96	121	18148.3	124
LanXiang	95	106	15861.3	96
CMUETC	92	111	16590.0	141
aaa	90	113	16980.6	95
KuantumKoalas	90	110	16530.6	91
whatever	87	126	18925.4	139

Welcome to Phase 2

- Is Q1 easier? Why is the rps reduced?
- Is Q2 easier? Why is the rps reduced?

Q2 Recap

- Don’t understand encoding? See:

https://piazza.com/class/hxs4b3o2lox5f9?cid=1595

- Still don’t understand encoding? Meet a TA!!!

- Question: Explain:

“q2.sample as only 499161 rows whereas 15619f14twitter-parta-aa has 500000 twwets”

Q3

- Use the ['retweeted_status']['user']['id'] and ['user']['id']

- Sort response numerically in order of user_id
- Ignore parentheses during sort

- What if user A retweeted two of user X's tweets?

- Q3 required rps is fairly high. Why?

Q4

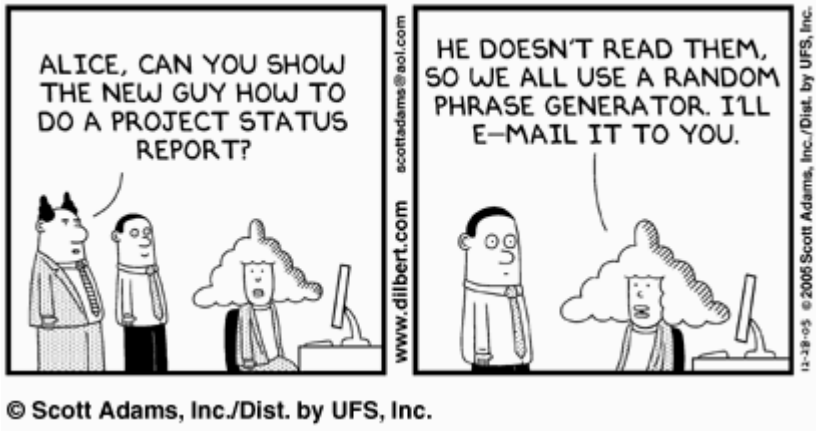
- Hashtags, places are case sensitive

- Read the rules carefully to find location
- Ignore “location” field

- Q4 required rps is fairly low. Why?

Phase 2 Report [VERY IMPORTANT]

- Start early
- Document your steps



- Identify and isolate the performance impact of each change you make
- Document your ideas and experiments

MAKE A QUANTITATIVE, DATA-DRIVEN REPORT

Live Test

- No benefit of database pre-caching
  - (unless you're really smart)

- 30 minute warm-up
- 2 hours Q1-Q4
- 30 minutes mix-Q1Q2Q3Q4
  - Avoid bottlenecks

- Lather, rinse, repeat

Grading

- The report really matters!!
  - Dense, not long
  - What you tried and what you measured matters
- Choose which run you get graded on
  - BUT,
    - You cannot use that DB for Phase 3
- How do we decide which DB?

Suggestions / Improvements

- UI
- Favicon
- Design
- Report comments
- Features
- Bugs

http://bit.ly/1roJsvU

Any questions?