



**Build. Unify. Scale.**

WIFI SSID:SparkAISummit | Password: UnifiedAnalytics

ORGANIZED BY  
 databricks



SPARK+AI  
SUMMIT 2019

# Cloud Storage Spring Cleaning: A Treasure Hunt

Zach Musgrave and Steven Moy  
Yelp Inc

**#UnifiedAnalytics #SparkAISummit**

# Agenda today

- What is our treasure?
- Where can we find some clues?
- How do we uncover the treasure map?
- What can we find with it?

# Motivation

- Insight into cloud storage patterns
- Justification to delete or transition data
- Classification of data via real-world accesses

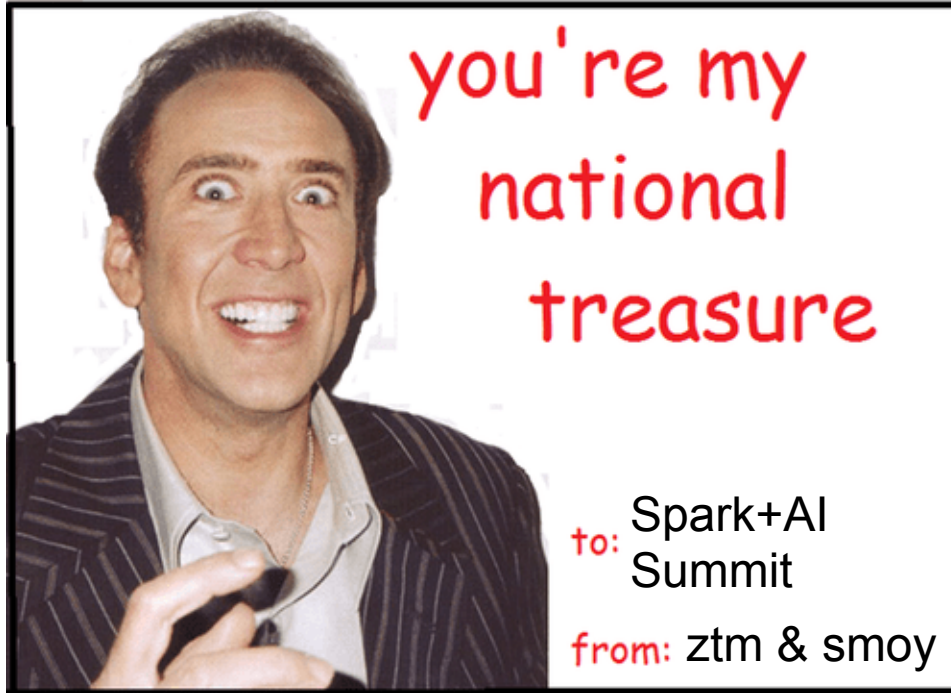
# Motivation

- In the beginning, there was infinite storage.
- Yelp said, "Yes, please!"
- Now, we don't have a problem, *per se...*

# Motivation

- Now, we don't have a problem, *per se*...
  - We have dozens of PBs worth of problems :/

# What is our treasure?



# What is our treasure?

*We keep many things in S3, and we want to lifecycle data into the right cost tier. How do we make the right decision?*

*For example, should we archive after 365 days? 400 days?*



# Motivation

- Cloud object storage: It's everywhere!
  - S3 continues to eat the world's storage workloads
  - No more filing tickets to add hard drives
- As long as you pay the monthly bill, it's easy...
  - Put(key, object)
  - Get(key)

# Eventually, leadership says...

***Why** are we spending so much money?  
Do we **really** need tens of petabytes **stored in the cloud**?*

# Eventually, leadership says...

***Why** are we spending so much money?*

*Do we **really** need tens of petabytes **for immediate access at all times**?*

# Eventually, leadership says...

*Why are we spending so much money?*  
*Do we **really** access all data equally all the time?*

# And then we say...

*If only we knew something - **anything!** - about engineers' access patterns...*

# We need a map...



# What is the treasure?

- Large storage savings, measured in \$\$\$
  - You may end up in your org's quarterly slide deck
- Security implications
  - Principle of least privilege made easy easier
- Access trails
  - Make decisions without rolling a 12-sided dice

# Where are the clues?

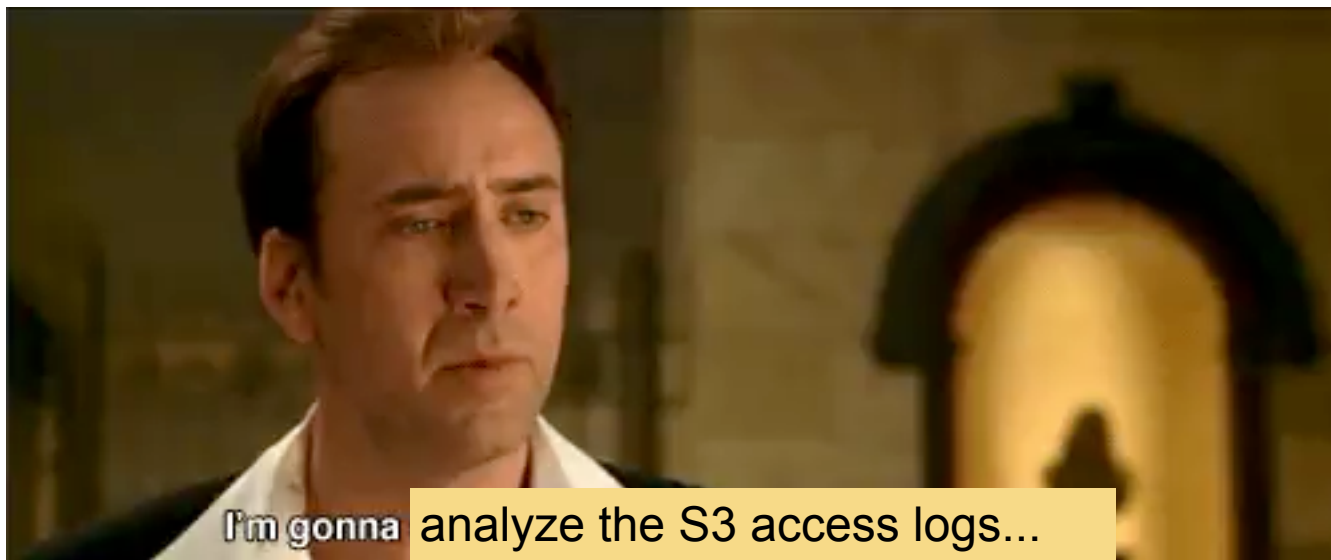
- We use AWS S3 as our example
  - Your storage provider likely has similar features
- S3 Server Side Logging
  - Apache-like access logs for every operation



# Where are the clues?

- Use S3?
  - Get access logs.
- Where do those go?
  - ... Inside S3 ...
  - If you enable them.

# Where are the clues?



# How do we do that?

Let's look at the contents of these logs.

# S3 Access Logs

- Non-compressed
- Row-based format
- Files can be very small
  - Range from KB to MB
  - Dozens - to many thousands - of rows
- Require a complex regex to parse

# S3 Access Logs: Example

```
79a59df900b949e55d96a1e698fbacedfd6e09d98eacf8f8d521
8e7cd47ef2be awsexamplebucket [06/Feb/2019:00:00:38
+0000] 192.0.2.3
79a59df900b949e55d96a1e698fbacedfd6e09d98eacf8f8d521
8e7cd47ef2be 3E57427F3EXAMPLE REST.GET.VERSIONING -
"GET /awsexamplebucket?versioning HTTP/1.1" 200 -
113 - 7 - "-" "S3Console/0.4" -
s91zHYrFp76ZVxRcpX9+5cjAnEH2ROuNkd2BHfIa6UkFVdtjf5mK
R3/eTPFvsiP/XV/VLi31234= SigV2 ECDHE-RSA-AES128-GCM-
SHA256 AuthHeader awsexamplebucket.s3.amazonaws.com
```

# Persistence and structure

- Conversion is a bit expensive
  - No need to repeat conversion over 10+ years of data
  - May as well store a queryable result
  - Determinism: Easy to replicate prior results.

# Before we go further

*All queries, code fragments, snippets, etc. we show you  
Are included in our Github repository!*

[github.com/Yelp/  
aws\\_logs\\_to\\_parquet\\_converter](https://github.com/Yelp/aws_logs_to_parquet_converter)

*So, no need to photograph each slide :)*

# Where are the clues?

The floor is the Declaration of Independence





## SQL Analytics (AWS Athena)

winces a design to reduce them under absolute Despotism, it is their right, it is their duty, to throw off such Government, and to provide new Guards for their future security. — Such has been the patient sufferance of these Colonies; and such is now the necessity which constrains them to alter their former Systems of Government. The history of the present King of Great Britain is a history of repeated injuries and usurpations, all having in direct object the establishment of an absolute Tyranny over these States. To prove this, let Facts be submitted to a candid world. — He has refused his Assent to Laws, the most wholesome and necessary for the public good. — He has forbidden his Governors to pass Laws of immediate and pressing importance, unless suspended in their operation till his Assent should be obtained; and when so suspended, he has utterly neglected to attend to them. — He has refused to pass other Laws for the accommodation of large districts of people, unless when such Laws were so framed as to serve his private turn, or to clothe his Ministers with arbitrary powers, by which they might oppress the people, or to establish new Offices, or to increase the Number of their Officers, or to make themselves by the Title of Chancery, a right ineffable to them and formidable to the People. — He has refused for the People at large for their exercise; the State remaining in the mean time exposed to all the dangers of Invasion from without, and convulsions within. — He has endeavoured to prevent the Population of these States; for that purpose obstructing the Laws for Naturalization of Foreigners; refusing to pass others to encourage their migrations hither, and raising the conditions of new Appropriations of Lands. — He has obstructed the Administration of Justice, by refusing his Assent to Laws for establishing Judiciary powers. — He has made Judges dependent on his Will alone, for the tenure of their offices, and the amount and payment of their salaries. — He has erected a multitude of New Offices, and sent hither swarms of Officers to harass our people, and eat out their substance. — He has kept among us, in times of peace, Standing Armies without the Consent of our Legislatures. — He has affected to render the Military independent of and superior to the Civil power. — He has combined with others to subject us to a jurisdiction foreign to our constitution, and unacknowledged by our Laws; giving his Assent to their Acts of pretended Legislation: — For quartering large bodies of armed troops among us: — For protecting them, by a mock Trial, from punishment for any Murders which they should commit on the Inhabitants of these States: — For cutting off our Trade with all parts of the world: — For imposing Taxes on us without our Consent: — For depriving us in many cases, of the benefits of Trial by jury: — For transporting us beyond Seas to be tried for pretended offences. — For abolishing the free System of English Laws in a neighbouring Province, establishing therein an Arbitrary government, and enlarging its Boundaries so as to render it at once an example and fit instrument for introducing the same absolute rule into these Colonies: — For taking away our Charters, abolishing our most valuable Laws, and altering fundamentally the Forms of our Governments: — For suspending our own Legislatures, and declaring themselves invested with power to legislate for us in all cases whatsoever. — He has abdicated Government here, by declaring us out of his Protection and waging War against us. — He has plundered our seas, ravaged our Coasts, burnt our towns, and destroyed the lives of our people. — He is at this time transporting large Armies of foreign Mercenaries to complete the works of death, desolation and tyranny, already begun with circumstances of Cruelty & perfidy scarcely paralleled in the most barbarous ages, and totally unworthy the Head of a civilized nation. — He has constrained our fellow Citizens taken Captive on the high Seas to bear Arms against their Country, to become the executioners of their friends and Brethren, or to fall themselves by their Hands. — He has excited domestic insurrections amongst us, and has endeavoured to bring on the inhabitants of our frontiers, the merciless Indian Savages, whose known rule of warfare, is an undistinguished destruction of all ages, sexes and conditions. In every stage of these Oppressions We have petitioned for Redress in the most humble terms: Our repeated Petitions have been answered by repeated injury. A Prince, whose character is thus marked by every act which may define a Tyrant, is unfit to be the ruler of a free people. Nor have We been wanting in attentions to our British brethren. We have warned them from time to time of attempts by their Legislature to extend an unwarrantable jurisdiction over us. We have reminded them of the circumstances of our emigration and settlement here. We have appealed to their native justice and magnanimity, and we have conjured them by the ties of our common kindred to disavow these usurpations, which, would inevitably interrupt our connections and correspondence. They too have been deaf to the voice of justice and of consanguinity. We must, therefore, acquiesce in the necessity, which denounces our Separation, and hold them, as we hold the rest of mankind, Enemies in War, in Peace Friends. — We, therefore, the Representatives of the united States of America, in General Congress, Assembled, appealing to the Supreme Judge of the world for the rectitude of our intentions, do, in the Name and to the effect of the said United States, hereby publish and declare: That these United Colonies are, and of Right ought to be, free and independent

# Clue #1 - SQL declarative analysis

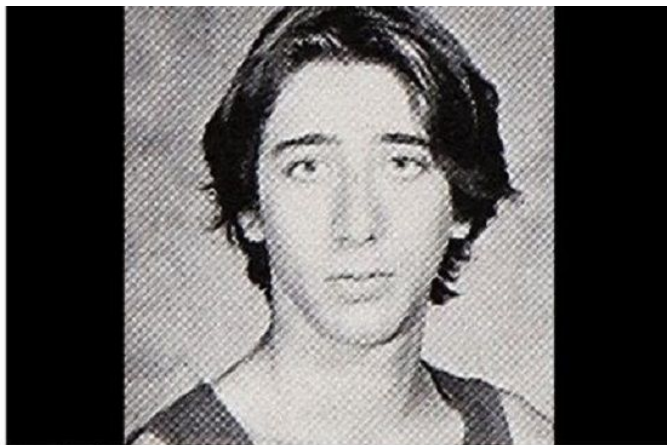
- Our developers and analysts speak SQL
- We can execute SQL against flat files
  - Lots of different ways to do this
  - AWS Athena is easiest for us

# Clue #1(a) Nomenclature

- Use your naming conventions
  - `s3://bucketname/logs/2019/03/04/any_file_name_here.tgz`
- Extract object's **creation date**:

```
date_parse(  
    array_join(  
        regexp_extract_all(key, '/(\d+)', 1), '-'  
    ),  
    '%Y-%m-%d'  
) AS dt_written    ==> date(2019, 03, 04)
```

# Clue #1(a) Nomenclature



Nicholas Cage

Hobbies: Acting

Future goals: "I'm going to steal the  
Declaration of Independence."

WeKnowMemes

# Clue #1(b) Days Apart

- We don't care **when** accesses occur
- We do care **how far in the past** the access is

```
date_diff(  
    'day',  
    date_parse(array_join(regexp_extract_all(..... ,  
    date_trunc('day', request_time)  
) AS days_apart,
```



# Clue #1(c) ARN Bingo

- Carefully break cardinality? Meaningful results!

	requester	
1	arn:aws:sts::	assumed-role/mrjob-ec2/i-00aa2e7cc1cd03b20
2	arn:aws:sts::	assumed-role/mrjob-ec2/i-0f26f91605a6408c5
3	arn:aws:sts::	assumed-role/mrjob-ec2/i-0af17382a31413147
4	arn:aws:sts::	assumed-role/mrjob-ec2/i-05ea56d0ca67476e8
5	arn:aws:sts::	assumed-role/mrjob-ec2/i-0aaa490443938b9c4
6	arn:aws:sts::	assumed-role/mrjob-ec2/i-0859888b9f7dc8a57
7	arn:aws:sts::	assumed-role/mrjob-ec2/i-0525d799d4cf98a48
8	arn:aws:sts::	assumed-role/mrjob-ec2/i-0a9cac4bbe4ecbf60
9	arn:aws:sts::	assumed-role/mrjob-ec2/i-0052def119a2e6c5b
10	arn:aws:sts::	assumed-role/mrjob-ec2/i-01d49e1b77775244f

# Clue #1(d) Group to Understand

- Any large org has **billions** of accesses per year
  - Need to aggregate accesses carefully

```
SELECT
    requester,
    log_name,
    count(*) AS access_count,
    sum(bytes_sent) AS total_bytes
FROM tmp_workspace WHERE
    days_apart > 365
GROUP BY 1, 2
ORDER BY access_count DESC
```

SQL Analytics (AWS Athena)

Columnar Files (Parquet)

Batch Conversion (Spark + Jupyter)



# Clue #2 Columnar File Format

- Many small files? List them all? Impossible.
  - Our access logs have **over a million** keys per prefix!
- Queries don't use all possible columns:
  - Amazon's schema has 19 columns
  - We usually reference 7 of those

# Clue #2 Columnar File Format

- Convert row-based access logs
  - Make them snappy-compressed Parquet files!
- Write a converter in Spark:
  - Good scalability
  - Great support for Parquet file format

SQL Analytics (AWS Athena)

Columnar Files (Parquet)

Batch Conversion (Spark + Jupyter)

S3 Prefix Scan (boto3 + RDD -> dataframe)

# Clue #3 S3 is not a filesystem

- Spark's S3 integration is *sort of* like HDFS
  - Many entries at a *directory*? Can't list them all.
  - `OutOfMemoryException` :(
- Access log keys
  - Prefixed with date information
  - Use a prefix-based S3 query
  - Paginate? Paginate.

# Clue #3 S3 is not a filesystem

```
127 def list_bucket_with_prefix(s3_client, bucket, prefix):
128     token = None
129     more_keys = True
130     keys = []
131     while more_keys:
132         kwargs = {
133             'Bucket': bucket,
134             'Prefix': prefix,
135         }
136         if token is not None:
137             kwargs['ContinuationToken'] = token
138         response = s3_client.list_objects_v2(
139             **kwargs
140         )
141         token = response.get('NextContinuationToken', None)
142         more_keys = (token is not None)
143         keys.extend([content['Key'] for content in response['Contents']])
144     return keys
```

# Clue #3 S3 is not a filesystem

- List all log files with a date-based prefix
  - Limit the number of results from S3
  - Constrain your memory utilization
- Generate RDD from the previous results
  - Convert the RDD to a dataframe?
  - Better ecosystem integrations

# Two-step analysis?

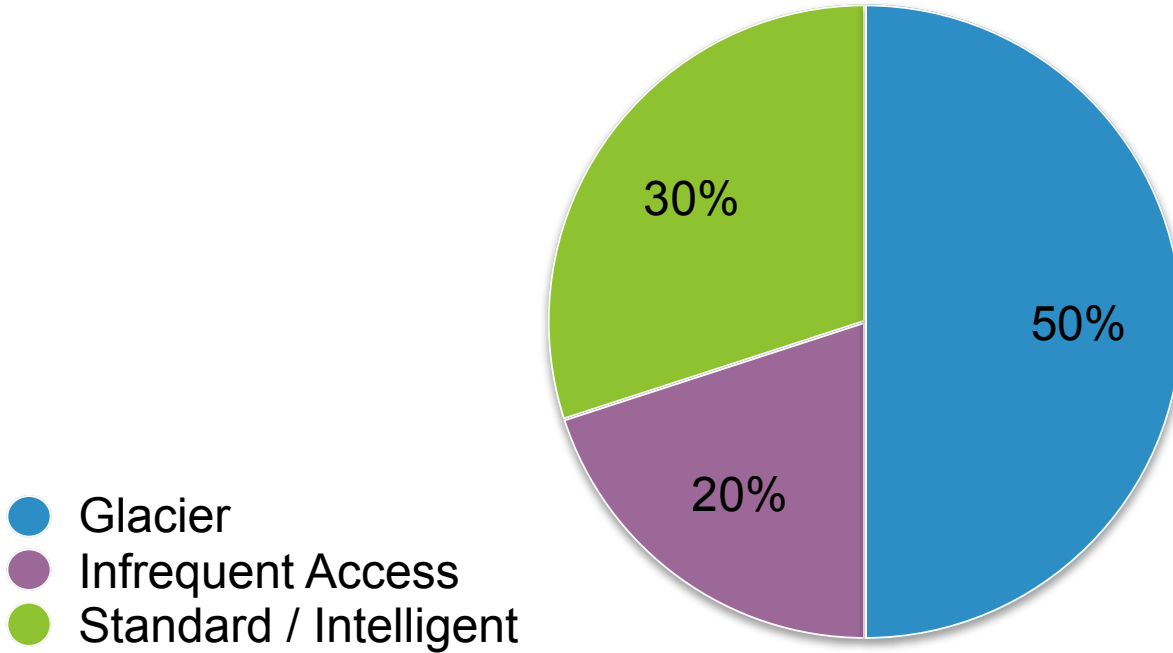
- Self-contained Spark job
  - Process S3 access logs
  - Parquet-ify, then write back to S3
- Query Parquet files with AWS Athena
  - We can all write SQL against them
  - Accessible in the org-wide Data Lake

# Org-wide Considerations

- First time at Yelp all data isn't instantly available
  - Glacier bulk restores require ~12 hours
  - Faster (15 minute) restores possible for small files
- Resulting developer impact
  - Very minor! Under ten restore operations so far.
  - Processing time for large computations **already** measured in days.



# What did we find?



# What did we find?

- ~20% data - 90 to 400 days old - rarely accessed
  - S3 Infrequent Access
    - Pay per download
    - Still immediately available
    - Developer impact: None!

# What did we find?

- ~50% data - over 400 days old - archivable
  - AWS S3 Glacier!
    - Pay to archive (once)
    - Pay to retrieve (rarely, usually never)
    - Need to wait ~12 hours between request and usage
    - Developer impact: Need to plan ahead.

# What did we find?

- Total savings to the org
  - **About 25%** of our monthly S3 bill. Gone for good!
- Total developer impact
  - **Very little.** Estimated at ~20 hours per year.
- Total pushback when implementing
  - **None.** Because we had the treasure map!
  - *Please sign off on this... unless you dislike money.*

# What did we do?

**We applied Spark, Parquet, Jupyter, and Athena to S3 access logs.**

**We wrote SQL to compute basic statistics.**

**We decreased Yelp's S3 bucket cost by ~25% going forward.**

# Did we steal the treasure?



# Thanks so much!

*All queries, code fragments, snippets, etc.  
Are included in our Github repository!*

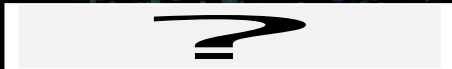
[github.com/Yelp/  
aws\\_logs\\_to\\_parquet\\_converter](https://github.com/Yelp/aws_logs_to_parquet_converter)



SPARK+AI  
SUMMIT 2019

**DON'T FORGET TO RATE  
AND REVIEW THE SESSIONS**

**SEARCH SPARK + AI SUMMIT**



SPARK+AI

