Goizueta Business School
MSBA 2018

Google BigQuery
Final Project

Lancy Mao, Athena Li, Elena Lopez

ISOM 671: Managing Big Data

Professor Panos

11/15/2017

Table of Contents

I. Introduction

This report serves to introduce Google's BigQuery Platform by outlining the multiple functions and extensions you can achieve using the Google Cloud Big Data Platform. In our tutorial we will provide the fundamentals in using this cloud based data warehouse including:

- ❏ Creating a database
- ❏ Loading the data into a table
- ❏ Querying a public dataset
- ❏ Previewing and querying the table
- ❏ Retrieving useful information via SQL

We were motivated to study Google's BigQuery platform because of its querying strengths compared to our current capabilities of MySQL that are limited by Workbench. By using BigQuery, we can extract key specifications in massive datasets and explore record level data in real time, due to its highly efficient and speedy output performance (MSV, 2014). The goal of this project is to present a clear understanding of BigQuery by presenting a made-up case answering questions related to stack overflow data, which we believe is relevant to our course studies.

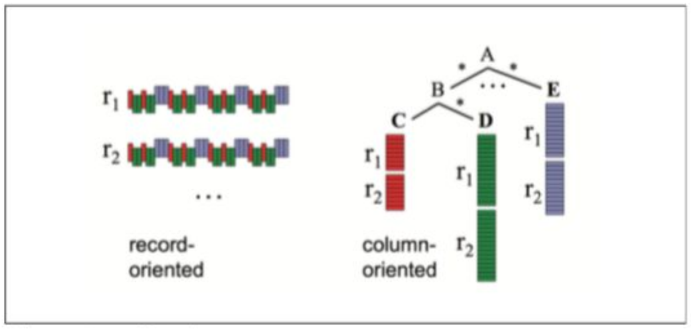
A. Conceptual Foundation

BigQuery was originally developed under the code name "Dremel" as one of Google's in-house core technologies. Dremel was designed to resolve the issues posed with MapReduce, such as its inability to handle large distributed datasets at an acceptable rate (MSV, 2014). Since BigQuery's public launch in 2012, users have been able to experience, first-hand, Dremel's high speed scalable query engine on data stored across thousands of servers (MSV, 2014).

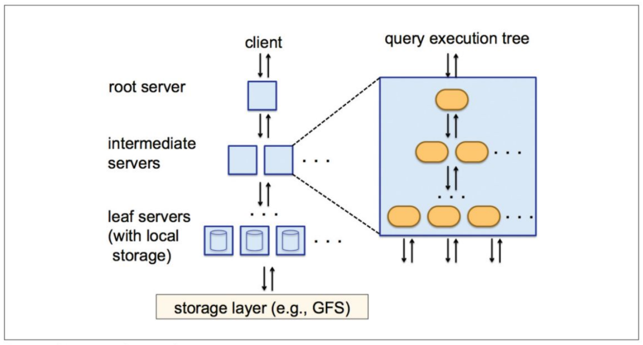
What is Dremel?
Dremel's claim to fame is inherent in its economies of scale. Back in 2006, when Google's internal engineers created the cloud-powered infrastructure that hosts a multitude of capabilities, most notably its parallelized and simultaneous streamlined query service, difficult tasks, such as scanning 223 million rows of Wikipedia titles containing a numeric character, became commonplace with most queries having an interactive output response time of 10 seconds (Sato, 2012). Dremel runs SQL-like queries on datasets of any magnitude and scale.

How does Dremel work?
There are two major sub-technologies within Dremel that are pivotal to its groundbreaking performance. First, is its columnar data storage function. Columnar storage accumulates each record into separate column values based on specific storage volume and allows for extremely high compression ratio and scan output (Sato, 2012). Dremel is a pioneer in delivering cloud service analytics that implements columnar storage. Previous databases, traditionally store every record into one equal volume. However, one disadvantage that lies in columnar storage is its inability to perform well when existing records need to be updated.

Dremel's Columnar Storage



Dremel's Tree Architecture



Source: Inside Look Google BigQuery

Next, is Dremel's collection of results through its tree architecture. This type of design facilitates the dispatched queries by forming parallel distributed trees. The queries are pushed through the tree to retrieve the results residing at the leaves, which contain aggregated results from thousands of machines (Sato, 2012). Together, the two reinforce each other's' capacities by unifying the cloud platform with a massively parallel columnar database.

BigQuery v. Dremel v. MapReduce

Essentially, BigQuery is the public form of Dremel, sharing both its high-caliber performance evaluation and foundational architectural design. In 2012, BigQuery was introduced externally to allow users outside of Google to harness the power of Dremel's infrastructure in their everyday Big Data processing and operating needs (Sato, 2012). The unprecedented and unique query performance of Dremel's features are made available through BigQuery by means of a command line interface, a Web UI, REST API, access control and more. This propelled developers to capitalize on Google's immense computational groundwork.

Both MapReduce and BigQuery consist of massive parallelized computing infrastructures. Even so, the fundamental differences between MapReduce and BigQuery are found in their common-case usages and running speeds of each technology. MapReduce is a programming model for processing large datasets and BigQuery is a query service for large datasets (Sato, 2012). MapReduce could take anywhere from minutes to days to run simple to longer jobs, while BigQuery is known for its very fast output response, regardless of the vastness of the dataset. BigQuery is easy to learn and employ for non-programmers, while MapReduce requires extensive programming knowledge that analysts or programmers are expected to know. While BigQuery may seem like it always outperforms, there are some disadvantages when compared to MapReduce. MapReduce is much better at handling and processing unstructured data, updating existing data, running extensive batch processes such as data mining, and programming complex data processing logic than BigQuery (Sato, 2012).

BigQuery Common Usages
When deciding whether or not to use BigQuery, the following criteria should be imposed:

	BigQuery	MapReduce
Finding particular records with specified conditions	✔	
Executing large join operations across huge datasets		✔
Gathering a quick aggregation of statistics with dynamically changing conditions	✔	
Performing trial-and-error data analysis to identify the root of the problem	✔	
Export and updating large amounts of data after processing		✔
Creating multiple iterations and paths of data processing with pre-programmed algorithms needed to be executed on Big Data		✔
Executing multiple data conversions and large join operations, but also quickly aggregating and performing ad-hoc data analysis on the result dataset	✔	✔

(Sato, 2012).

B. Background

Our made-up case utilizes the Stackoverflow dataset found on BigQuery's public databases. Our focus consists of three tables within the dataset: post_questions, post_answers, and users. There are 21.8 GB (14mm rows), 17.7 GB (22mm rows), and 1.1 GB (7mm rows) in each table, respectively. The goal of the analysis is to present a real-life example that is relevant to our course, and offers the reader valuable insight into BigQuery's capabilities.

In our problem set, we pose the following 6 questions and provide solutions for each:

1. Posts questions count and growth rate by year
2. Posts questions count changes over time by tags (tools covered in class; relational DBMS)s
3. The most active user posting questions and Hadoop-related questions in 2017
4. The most active user answering questions in 2017
5. Correlation between user reputation and accepted answers
6. The post with the most favorite/ view count of Hadoop tag

Our findings are presented through a series of plots, tables, and explanations, to help visualize the answers for the questions above.

II. Methodology

Initial Setup

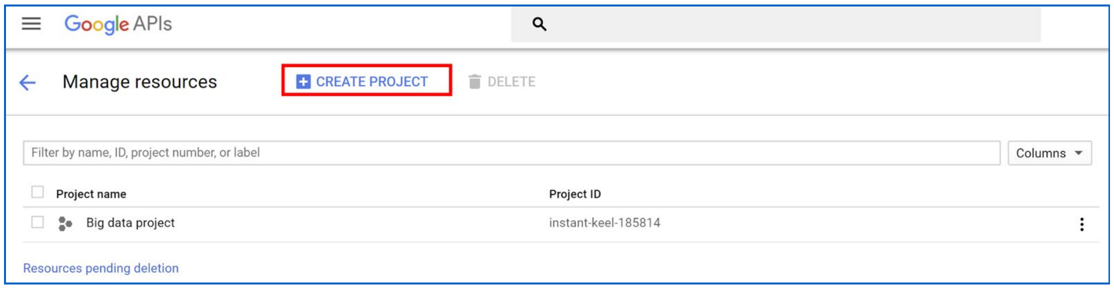
Before you are ready to start a BigQuery Project, there are a couple more steps you need to do:

- ❏ Go to the website cloud.google.com and sign into your Google account.
- ❏ Click "TRY IT FREE" to navigate to the next page.
- ❏ You will be prompted to a webpage where Google requests your billing information. Enter the billing information and enable billing.
- ❏ Click the "CONSOLE" next to your avatar. You are now at your main dashboard. In this page, you can do things such as manage your projects and jobs, monitor your compute engine, and enable APIs.
- ❏ BigQuery is automatically enabled in a new project. To activate BigQuery in a pre-existing project, go to Enable the BigQuery API.

Start a BigQuery Project on StackOverflow data

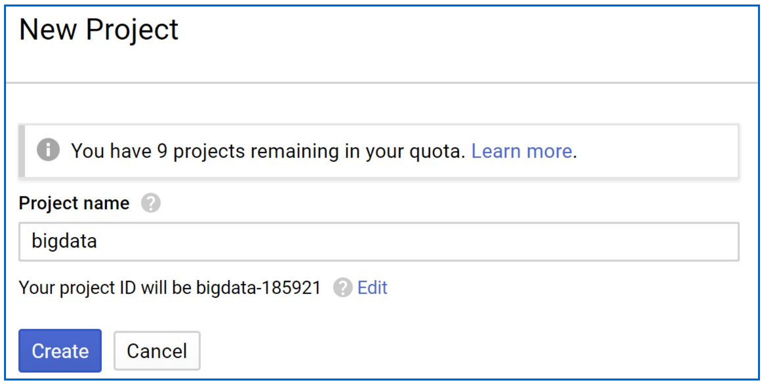
Now you are all set to initiate your first BigQuery project on Google Cloud Platform.

1. Go to Manage resources and create a new project


Google APIs
Manage resources
CREATE PROJECT
DELETE
Filter by name, ID, project number, or label
Columns
Resources pending deletion

Project name	Project ID
Big data project	instant-keel-185814

2. Type in a project name and the system will assign you a project ID
   a. The project ID is the globally unique identifier for your project. You cannot change the project ID after the project is created. It will be used in SQL query to refer to the database and table.

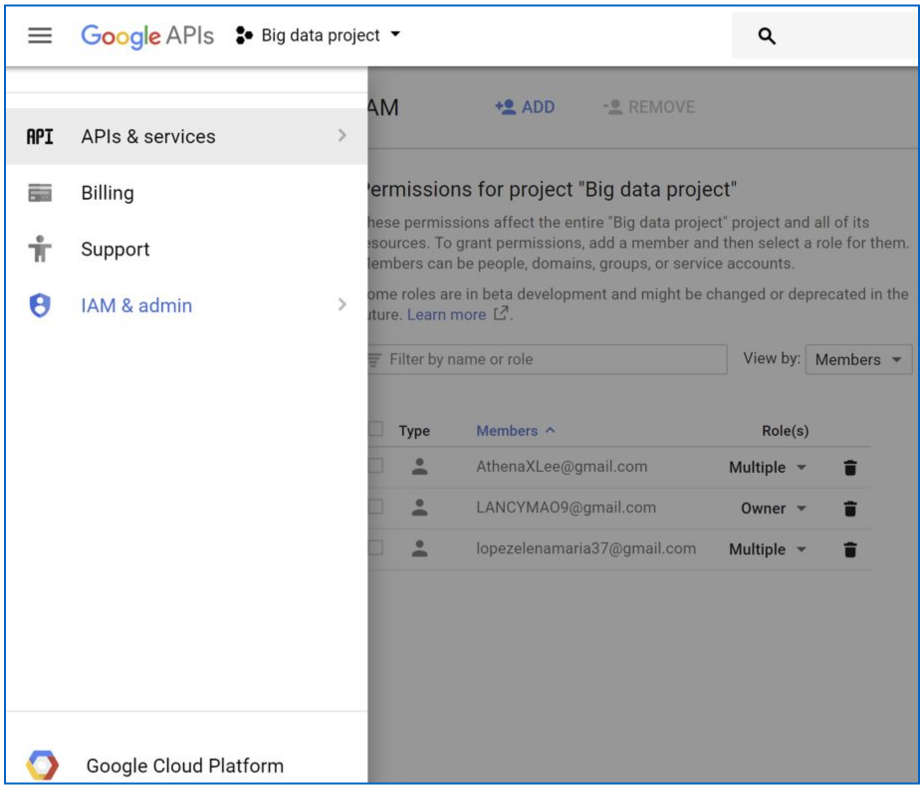


You have 9 projects remaining in your quota.

Project name: bigdata

Your project ID will be bigdata-185921

3. Click on the three stripes next to Google's logo. Click Google Cloud Platform to go back to the Home page.



Permissions for project "Big data project"

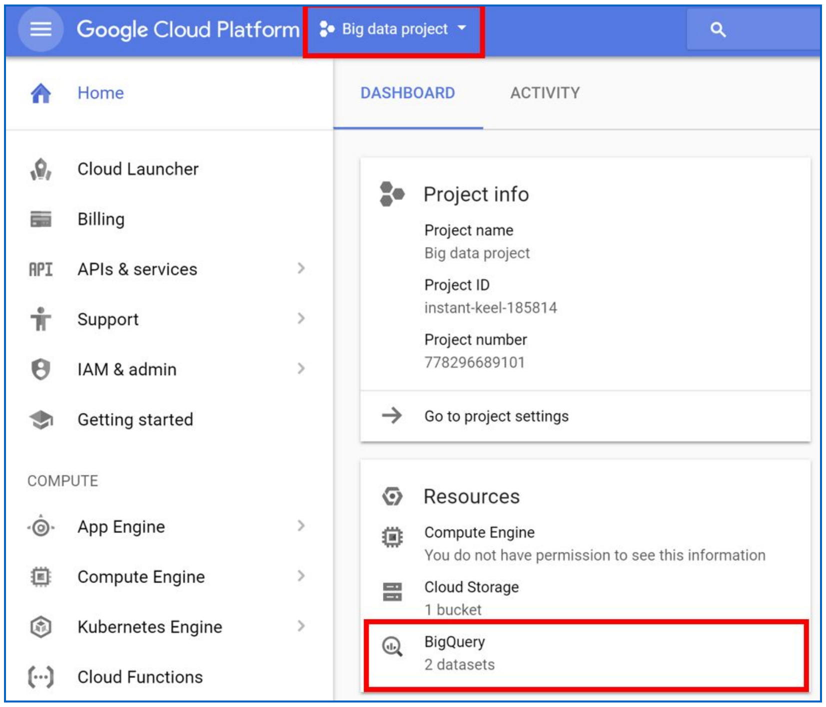
These permissions affect the entire "Big data project" project and all of its resources. To grant permissions, add a member and then select a role for them. Members can be people, domains, groups, or service accounts.

Some roles are in beta development and might be changed or deprecated in the future. Learn more.

Filter by name or role — View by: Members

Type	Members	Role(s)
	AthenaXLee@gmail.com	Multiple
	LANCYMAO9@gmail.com	Owner
	lopezelenamaria37@gmail.com	Multiple

4. Click BigQuery in Resources section to go to the BigQuery Web UI



Project info

Project name
Big data project

Project ID
instant-keel-185814

Project number
778296689101

Go to project settings

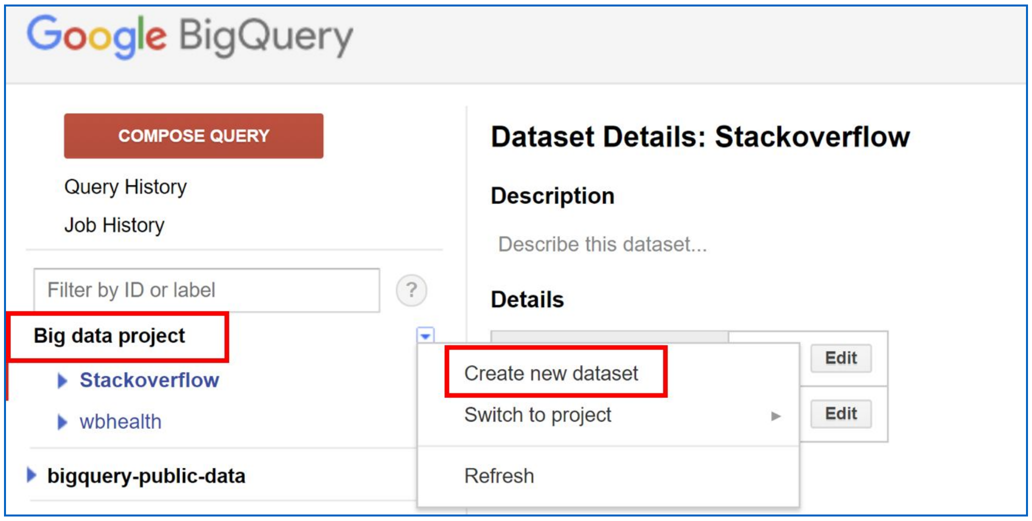
Resources

Compute Engine
You do not have permission to see this information

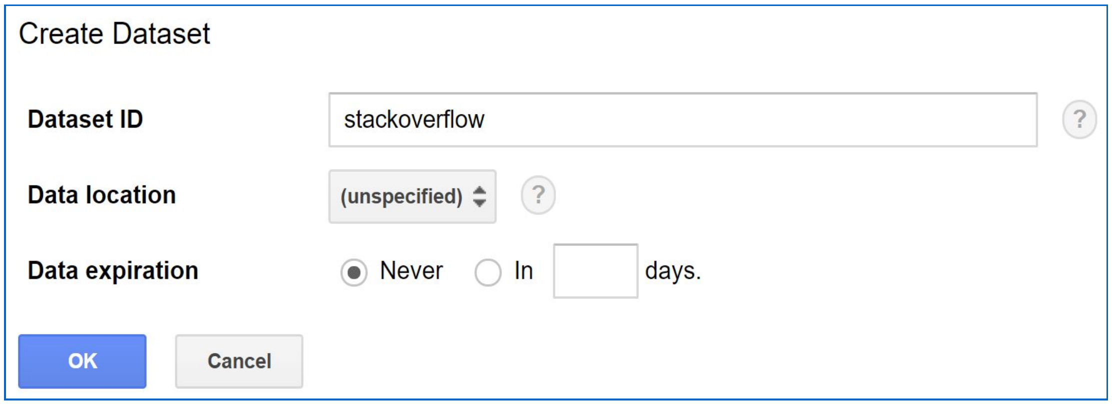
Cloud Storage
1 bucket

BigQuery
2 datasets

5. Click on the little triangle shape button next to your project name and click "Create new dataset.

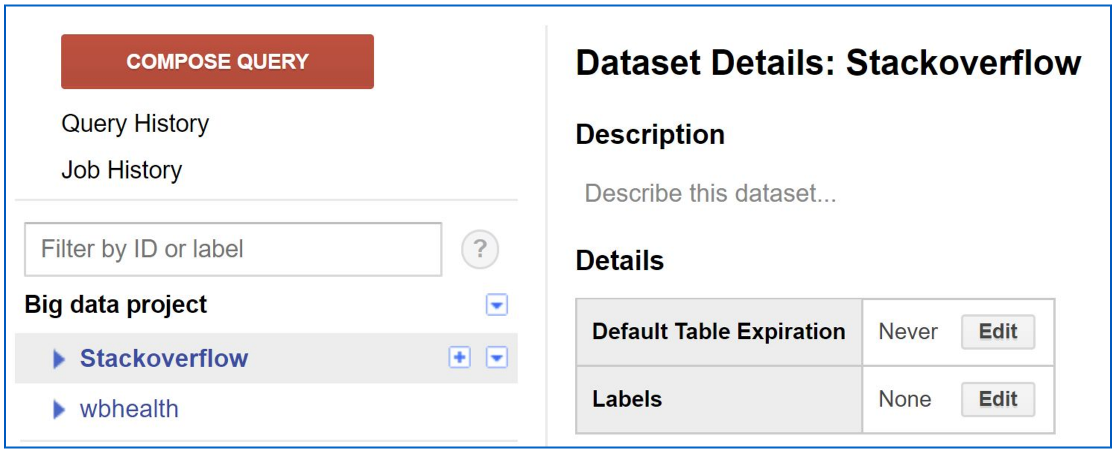


6. In the prompt window, create your Dataset ID in this case, we named it as stackoverflow



Dataset ID	stackoverflow
Data location	(unspecified)
Data expiration	[✓ Never] In days.

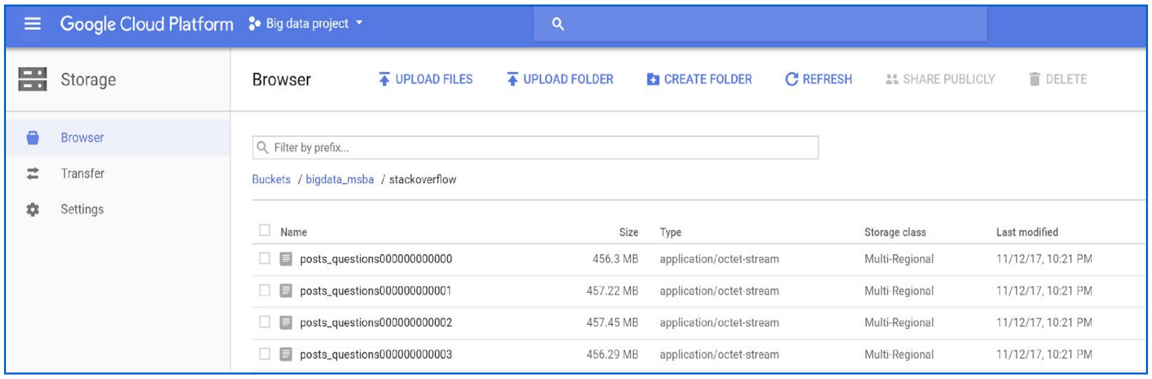
7. After database is created, you can click "+" after the database name to create as many table as you want.



Default Table Expiration	Never	Edit
Labels	None	Edit

8. Now you can load your data into the created dataset.
   a. There are several ways to load data into your Google cloud project: from Google Cloud Storage, a readable data source (such as your local machine), streaming inserts, by using DML statements to perform bulk inserts. We can use a Google Cloud Dataflow pipeline to write data to BigQuery.
   b. There are also several ways that do not require manually loading the data. You can use the publicly shared datasets that are stored in BigQuery. If another user has shared a dataset with you, you can run queries on that dataset without loading the data. You can skip the data loading process by creating a table that is based on an external data source.
   c. You can also use stackdriver log file. Cloud Logging provides an option to export log files into BigQuery. Streaming the data one record at a time is another way to avoid manually loading the data. This method is typically used when you need the data to be available immediately.

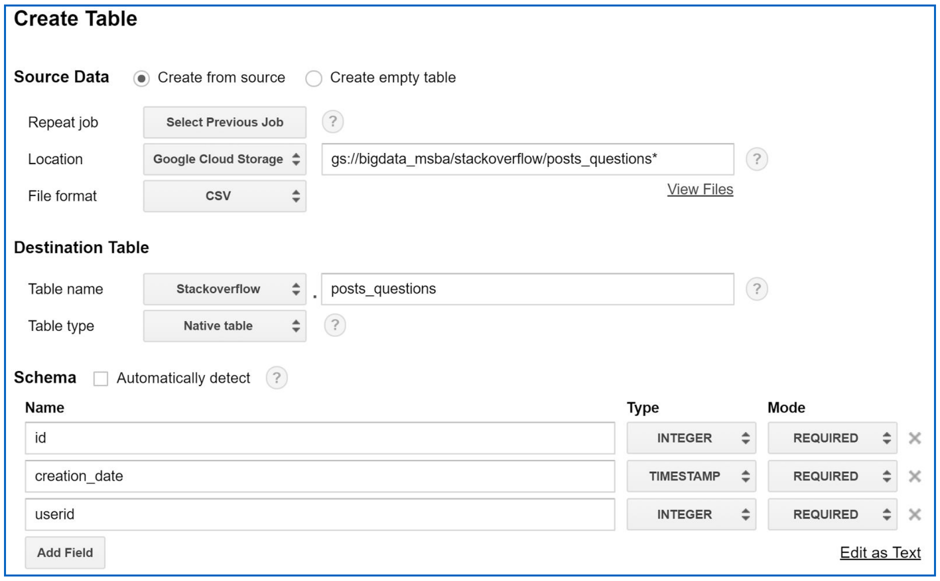
9. In this demonstration, we are going to create a data table by fetching data from the Google Cloud Storage, where you store your raw data sets. Copy the path of the data. You will need this later.



Buckets / bigdata_msba / stackoverflow

Name	Size	Type	Storage class	Last modified
posts_questions000000000000	456.3 MB	application/octet-stream	Multi-Regional	11/12/17, 10:21 PM
posts_questions000000000001	457.22 MB	application/octet-stream	Multi-Regional	11/12/17, 10:21 PM
posts_questions000000000002	457.45 MB	application/octet-stream	Multi-Regional	11/12/17, 10:21 PM
posts_questions000000000003	456.29 MB	application/octet-stream	Multi-Regional	11/12/17, 10:21 PM

10. Paste the path in the location. The "Schema" section takes your table attribute names. Iterate this process for multiple times until you get all your tables.
    a. Google Cloud Storage URIs begin with "gs://" and specify the bucket and object you want to load.
       i. Example: gs://mybucket/path/to/mydata.csv. You can use a wildcard to load multiple files



Source Data [✓ Create from source] Create empty table

Repeat job: Select Previous Job
Location: Google Cloud Storage — gs://bigdata_msba/stackoverflow/posts_questions*
File format: CSV

Destination Table
Table name: Stackoverflow . posts_questions
Table type: Native table

Schema ☐ Automatically detect

Name	Type	Mode
id	INTEGER	REQUIRED
creation_date	TIMESTAMP	REQUIRED
userid	INTEGER	REQUIRED

11. Now Click on the “Compose Query” button. You can now follow the steps and create your own queries with SQL.

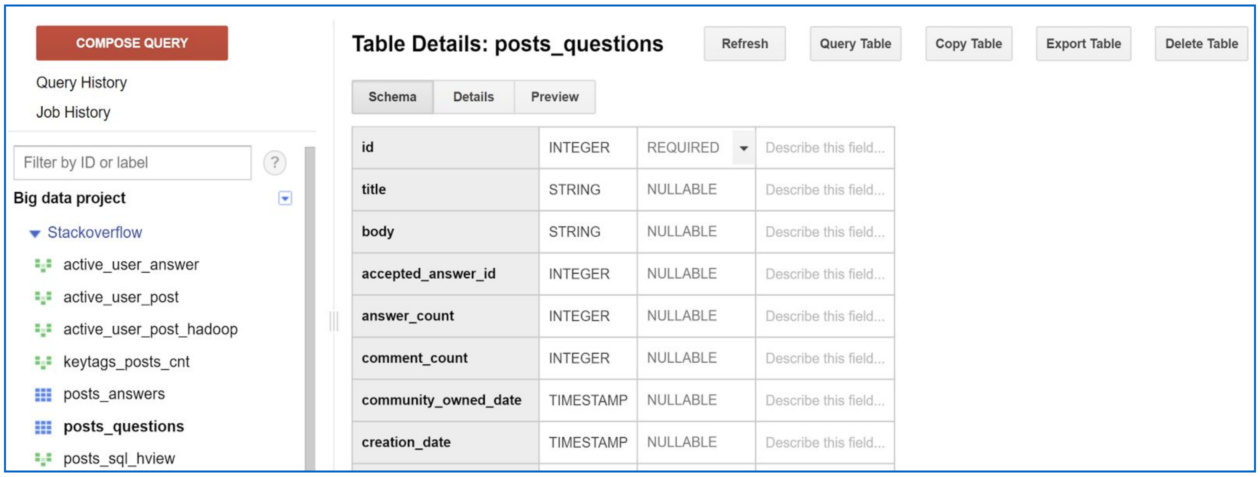
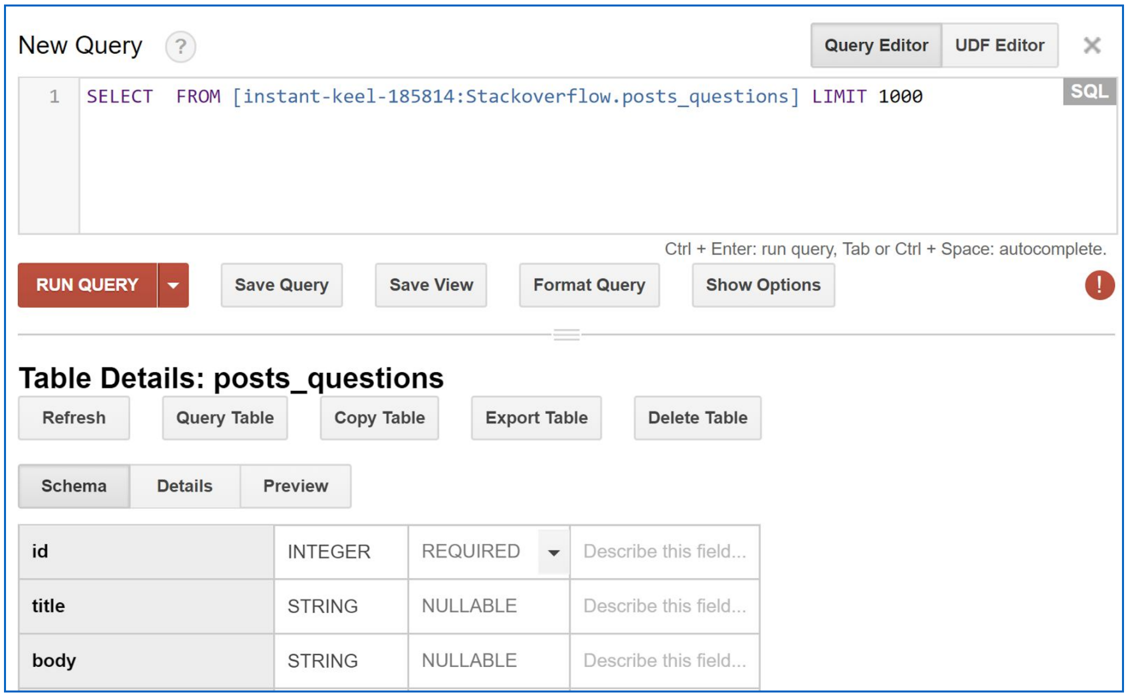


Table Details: posts_questions

id	INTEGER	REQUIRED	Describe this field...
title	STRING	NULLABLE	Describe this field...
body	STRING	NULLABLE	Describe this field...
accepted_answer_id	INTEGER	NULLABLE	Describe this field...
answer_count	INTEGER	NULLABLE	Describe this field...
comment_count	INTEGER	NULLABLE	Describe this field...
community_owned_date	TIMESTAMP	NULLABLE	Describe this field...
creation_date	TIMESTAMP	NULLABLE	Describe this field...

12. New Query will be on the top of the screen with table schema below, so it will be more convenient to check fields in tables while writing queries. We can save query and view it.



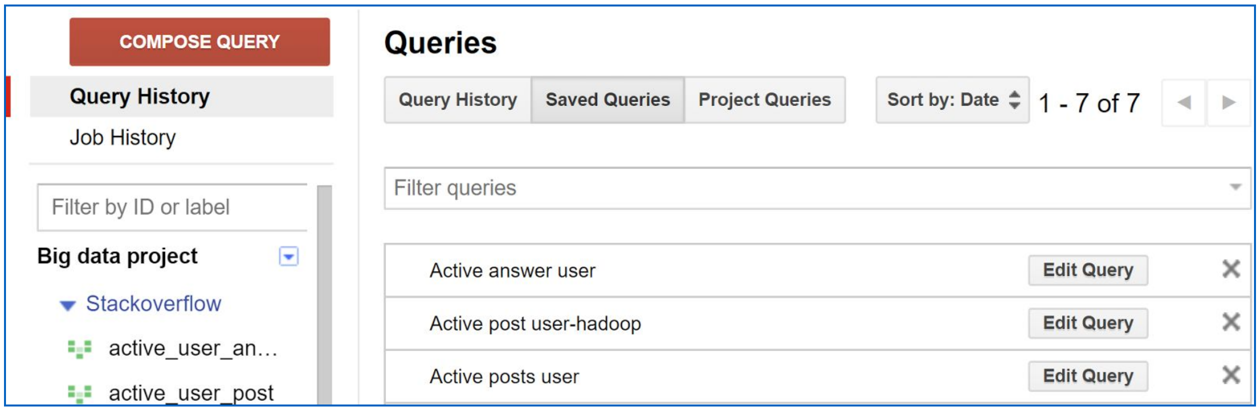
```
SELECT  FROM [instant-keel-185814:Stackoverflow.posts_questions] LIMIT 1000
```

Ctrl + Enter: run query, Tab or Ctrl + Space: autocomplete.

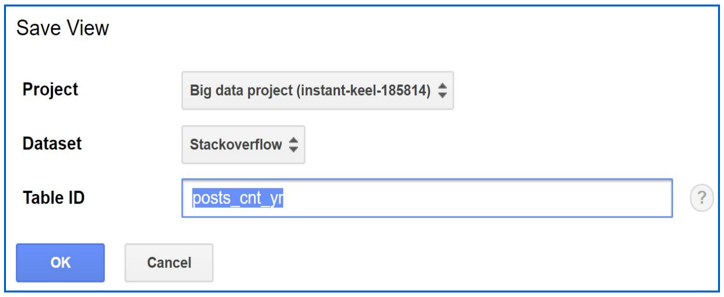
Table Details: posts_questions

id	INTEGER	REQUIRED	Describe this field...
title	STRING	NULLABLE	Describe this field...
body	STRING	NULLABLE	Describe this field...

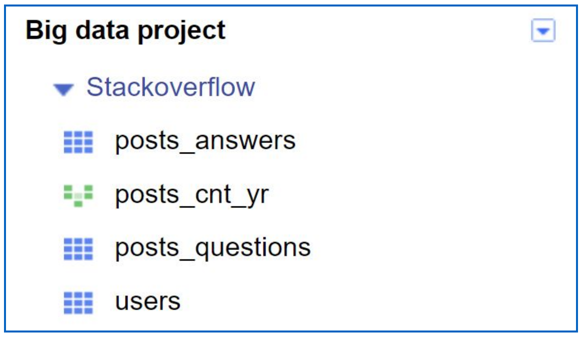
13. You can also save query for future use. If you click "Query history" on the upper left side, you can see query history, saved queries, and project queries. edit the saved queries and rerun them.



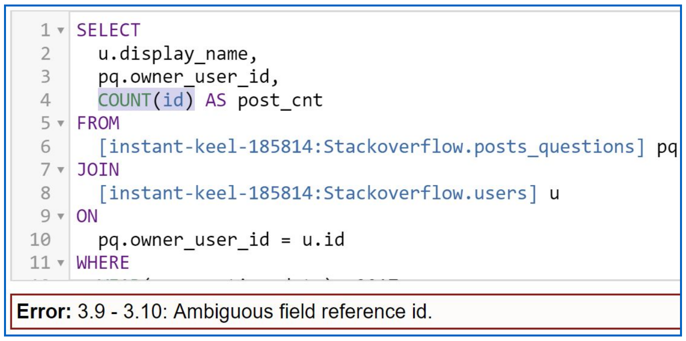
14. Save view: if you want to create a table based on queries on other tables, you can just click "save view" to create a view without coding.

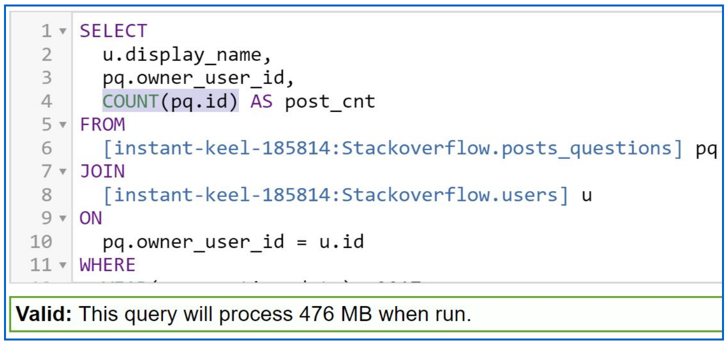

Save View

Project	Big data project (instant-keel-185814)
Dataset	Stackoverflow
Table ID	posts_cnt_yr

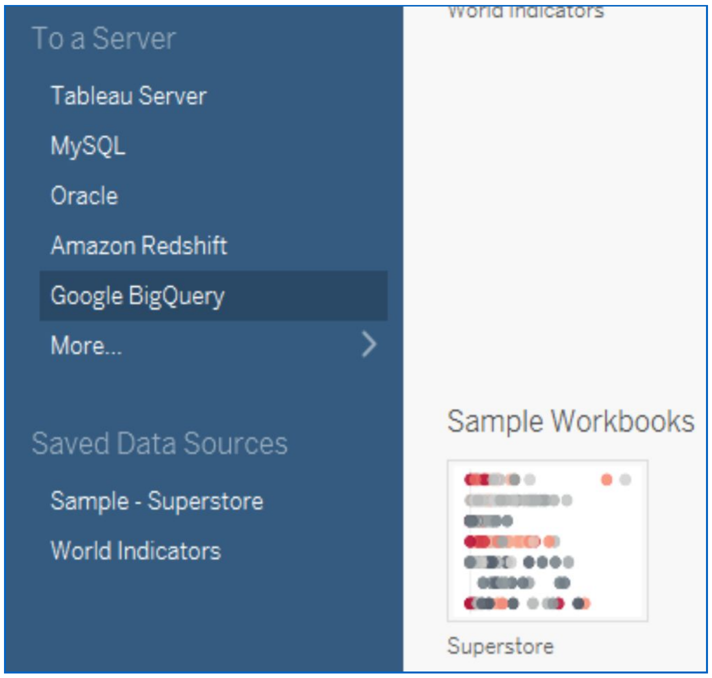


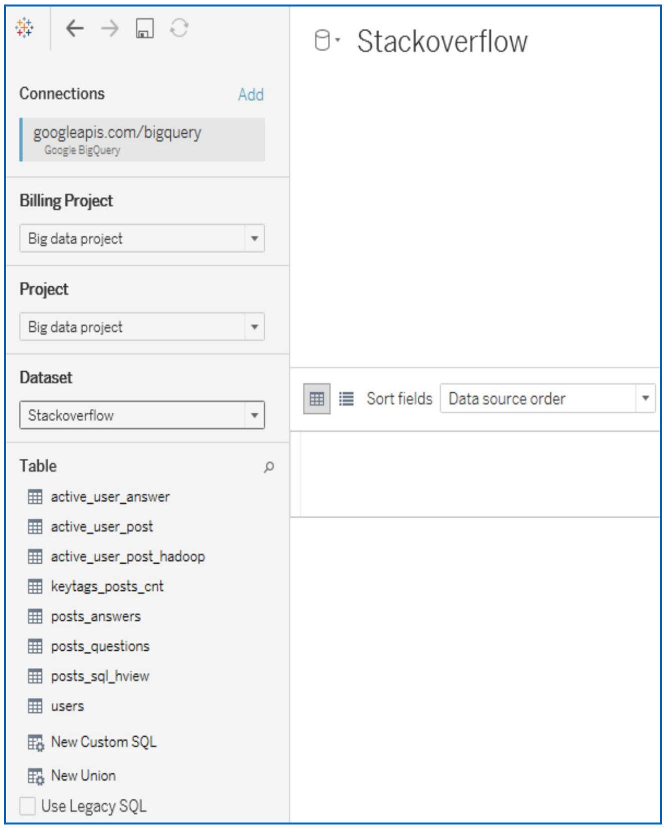
15. Error message will change once you correct the error, so it will avoid running incorrect queries several times. Also you can see how much data the query will process.


```
SELECT
  u.display_name,
  pq.owner_user_id,
  COUNT(id) AS post_cnt
FROM
  [instant-keel-185814:Stackoverflow.posts_questions] pq
JOIN
  [instant-keel-185814:Stackoverflow.users] u
ON
  pq.owner_user_id = u.id
WHERE
```
Error: 3.9 - 3.10: Ambiguous field reference id.


```
SELECT
  u.display_name,
  pq.owner_user_id,
  COUNT(pq.id) AS post_cnt
FROM
  [instant-keel-185814:Stackoverflow.posts_questions] pq
JOIN
  [instant-keel-185814:Stackoverflow.users] u
ON
  pq.owner_user_id = u.id
WHERE
```
Valid: This query will process 476 MB when run.

16. Another powerful feature of Google BigQuery is to link the query result with other software such as Tableau and SAP.





Billing Project: Big data project
Project: Big data project
Dataset: Stackoverflow

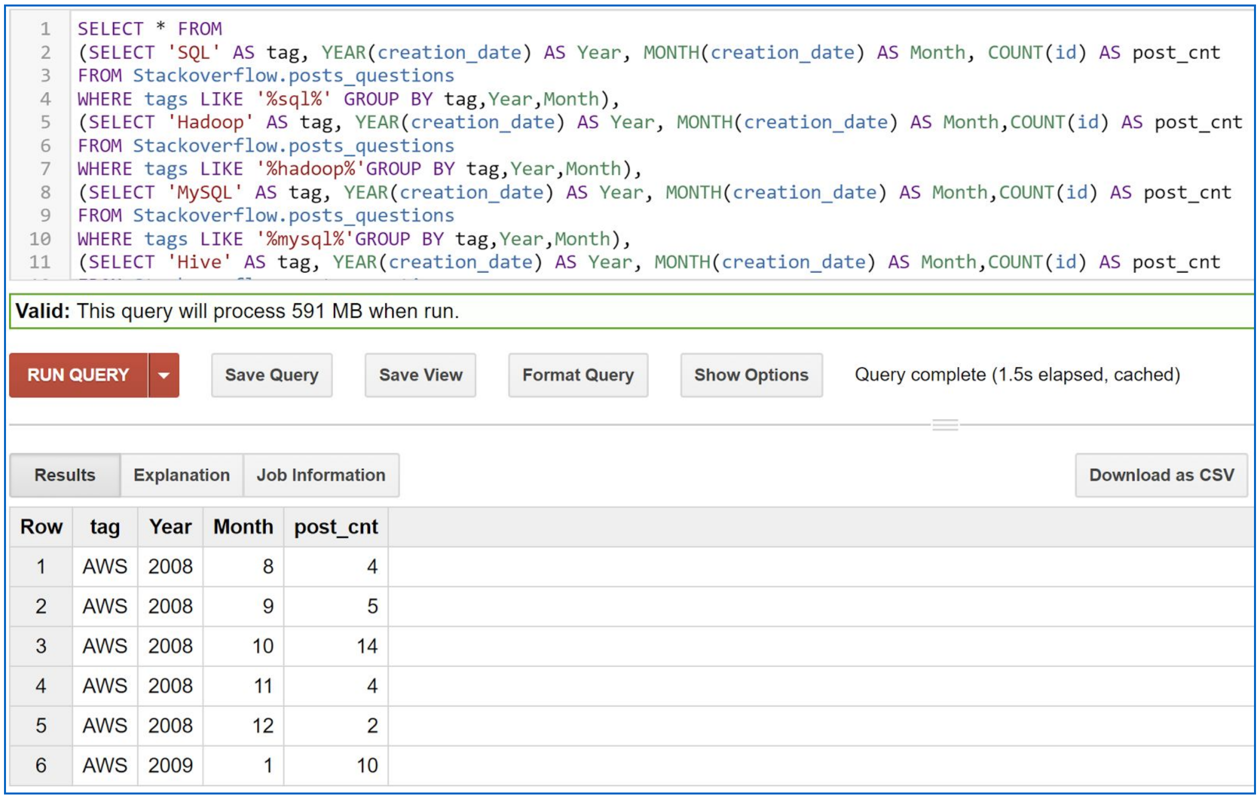
Table:
- active_user_answer
- active_user_post
- active_user_post_hadoop
- keytags_posts_cnt
- posts_answers
- posts_questions
- posts_sql_hview
- users

III. Application and Findings

1. We use Stackoverflow data for this demonstration. This dataset was last updated August, 2017. Originally, there are 16 tables, and we chose the following three to include in our database. Now we can start to create our queries.

Table ID	**posts_questions**	**posts_answers**	**users**
Table Size	21.8 GB	17.7 GB	1.10 GB
Number of Rows	14,458,875	22,668,556	7,617,191

2. First we want to find the frequency of big data family words such as “hive”, “hadoop”, “ SQL”, “SQL-Server” “AWS”, “ Pig”, “NoSQL” “ Spark”, “Impala”, “Teradata” that were tagged in the question posts over the last 10 years. The SQL code for the query is as follow:



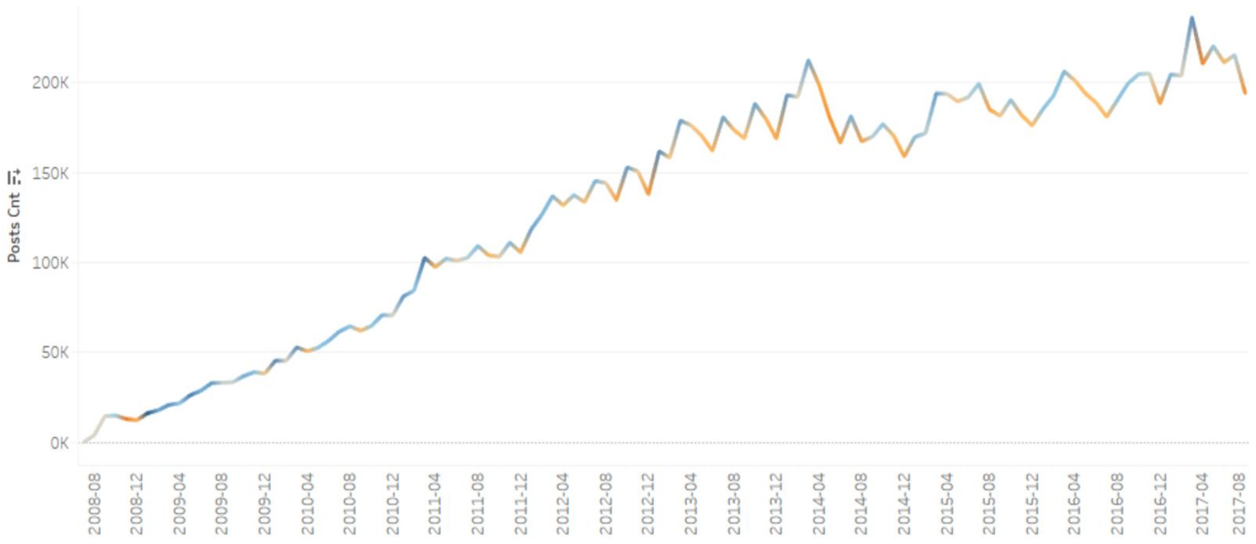
```
SELECT * FROM
(SELECT 'SQL' AS tag, YEAR(creation_date) AS Year, MONTH(creation_date) AS Month, COUNT(id) AS post_cnt
FROM Stackoverflow.posts_questions
WHERE tags LIKE '%sql%' GROUP BY tag,Year,Month),
(SELECT 'Hadoop' AS tag, YEAR(creation_date) AS Year, MONTH(creation_date) AS Month,COUNT(id) AS post_cnt
FROM Stackoverflow.posts_questions
WHERE tags LIKE '%hadoop%'GROUP BY tag,Year,Month),
(SELECT 'MySQL' AS tag, YEAR(creation_date) AS Year, MONTH(creation_date) AS Month,COUNT(id) AS post_cnt
FROM Stackoverflow.posts_questions
WHERE tags LIKE '%mysql%'GROUP BY tag,Year,Month),
(SELECT 'Hive' AS tag, YEAR(creation_date) AS Year, MONTH(creation_date) AS Month,COUNT(id) AS post_cnt
```

Valid: This query will process 591 MB when run.

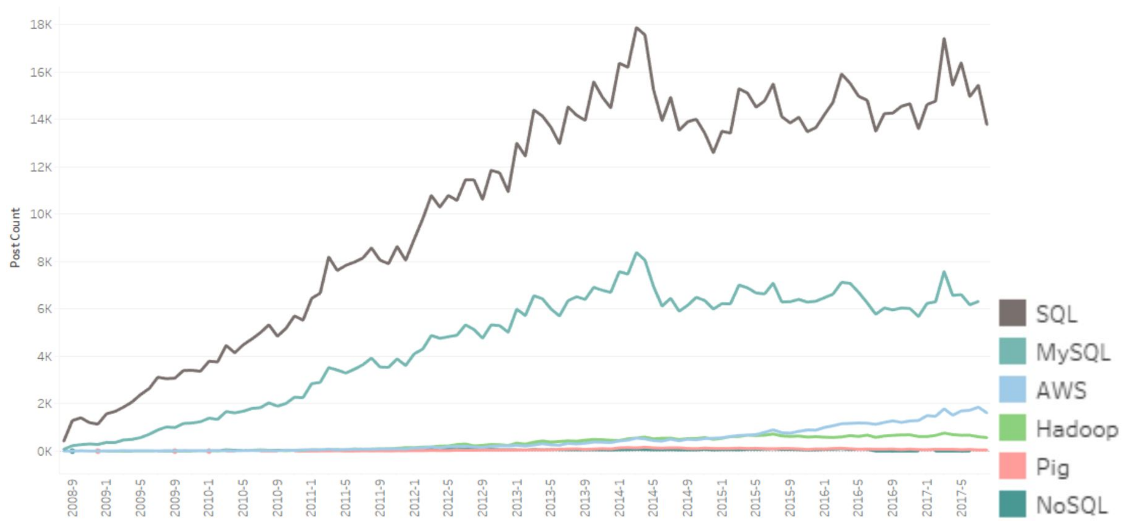
Query complete (1.5s elapsed, cached)

Row	tag	Year	Month	post_cnt
1	AWS	2008	8	4
2	AWS	2008	9	5
3	AWS	2008	10	14
4	AWS	2008	11	4
5	AWS	2008	12	2
6	AWS	2009	1	10

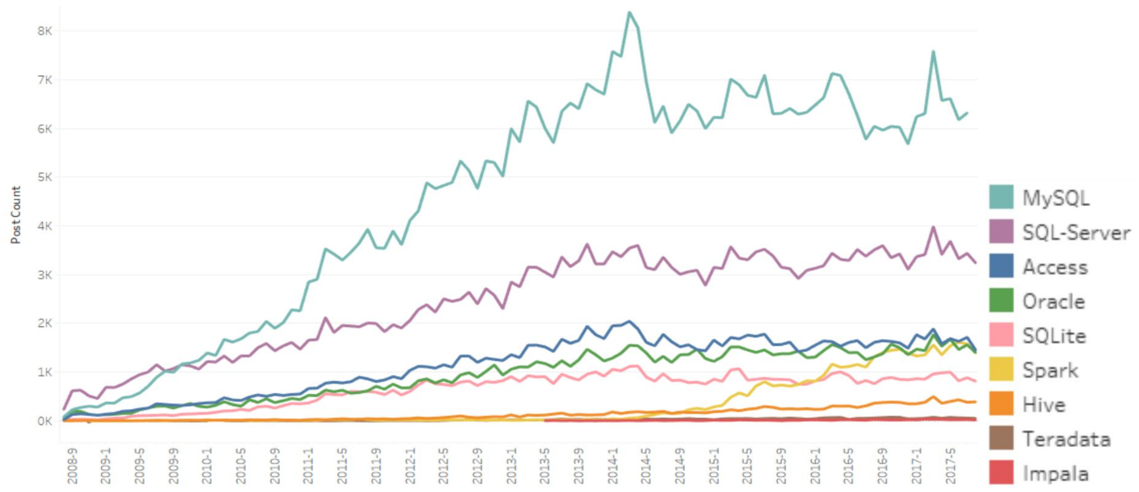
3. From this table, we plotted the yearly trend of total count for all of the tagged question posts



Finding 1: We can see that there is a clear upward trend of how frequently these buzz words are tagged over years.

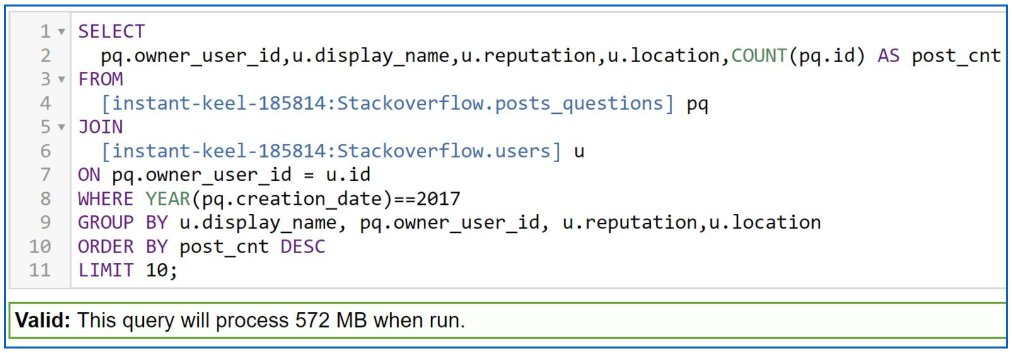


Finding 2: This chart highlighted big data tools that we have covered in class. We found that SQL is still one of the most mentioned buzz word out there. AWS is also leading the way.



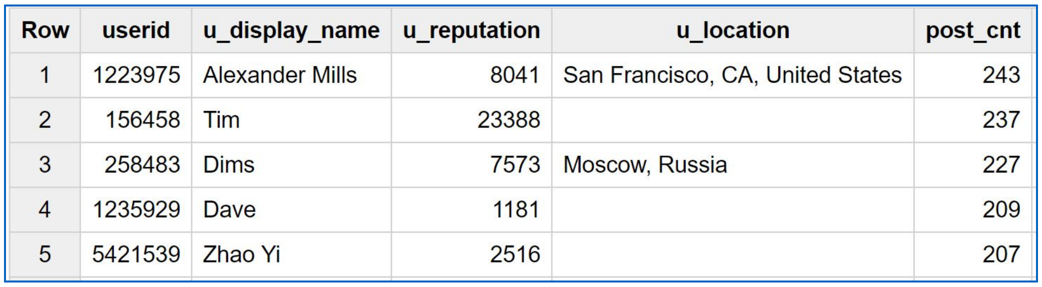
Finding 3: Among all of the Database Management Systems, MySQL and SQL-Server are two of the most mentioned tags. This implies the current industrial usage of such tool.

1. For our second query, we want to explore the users in stackoverflow. We want to find active users with the most question posts in 2017.



```
SELECT
  pq.owner_user_id,u.display_name,u.reputation,u.location,COUNT(pq.id) AS post_cnt
FROM
  [instant-keel-185814:Stackoverflow.posts_questions] pq
JOIN
  [instant-keel-185814:Stackoverflow.users] u
ON pq.owner_user_id = u.id
WHERE YEAR(pq.creation_date)==2017
GROUP BY u.display_name, pq.owner_user_id, u.reputation,u.location
ORDER BY post_cnt DESC
LIMIT 10;
```

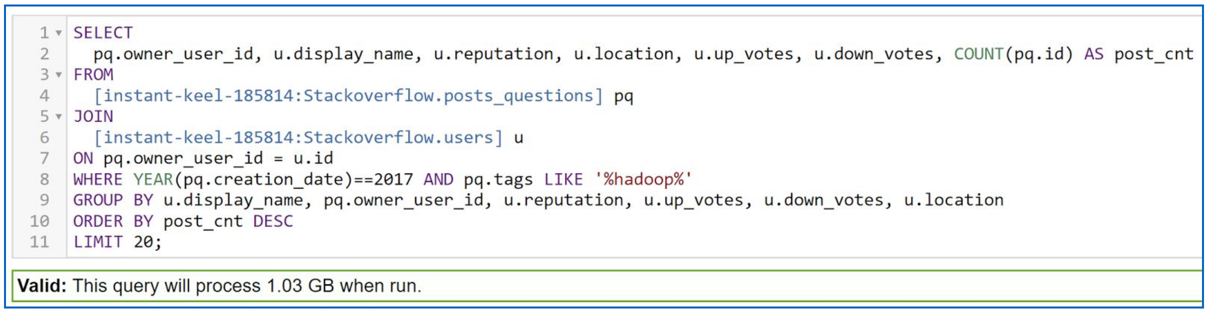
Valid: This query will process 572 MB when run.



Row	userid	u_display_name	u_reputation	u_location	post_cnt
1	1223975	Alexander Mills	8041	San Francisco, CA, United States	243
2	156458	Tim	23388		237
3	258483	Dims	7573	Moscow, Russia	227
4	1235929	Dave	1181		209
5	5421539	Zhao Yi	2516		207

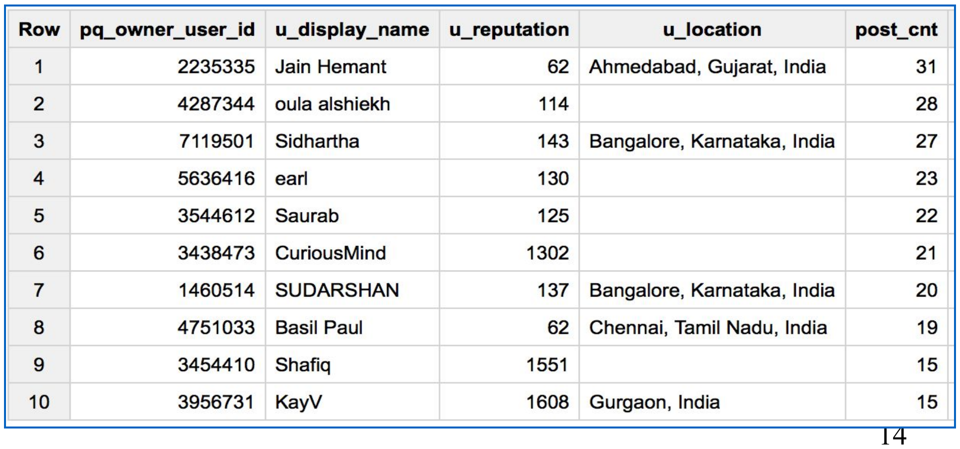
Finding 4: Alexander Mills has the most question posts as of 2017. A total of 243 question posted

2. Now we want to know which user has the most hadoop related question posts and their country of origin.



```
SELECT
  pq.owner_user_id, u.display_name, u.reputation, u.location, u.up_votes, u.down_votes, COUNT(pq.id) AS post_cnt
FROM
  [instant-keel-185814:Stackoverflow.posts_questions] pq
JOIN
  [instant-keel-185814:Stackoverflow.users] u
ON pq.owner_user_id = u.id
WHERE YEAR(pq.creation_date)==2017 AND pq.tags LIKE '%hadoop%'
GROUP BY u.display_name, pq.owner_user_id, u.reputation, u.up_votes, u.down_votes, u.location
ORDER BY post_cnt DESC
LIMIT 20;
```

Valid: This query will process 1.03 GB when run.



Row	pq_owner_user_id	u_display_name	u_reputation	u_location	post_cnt
1	2235335	Jain Hemant	62	Ahmedabad, Gujarat, India	31
2	4287344	oula alshiekh	114		28
3	7119501	Sidhartha	143	Bangalore, Karnataka, India	27
4	5636416	earl	130		23
5	3544612	Saurab	125		22
6	3438473	CuriousMind	1302		21
7	1460514	SUDARSHAN	137	Bangalore, Karnataka, India	20
8	4751033	Basil Paul	62	Chennai, Tamil Nadu, India	19
9	3454410	Shafiq	1551		15
10	3956731	KayV	1608	Gurgaon, India	15

Finding 5: We found that people who ask the most hadoop related questions are all from India.

3. Now we would like to find who are the users that answered the most questions in 2017.

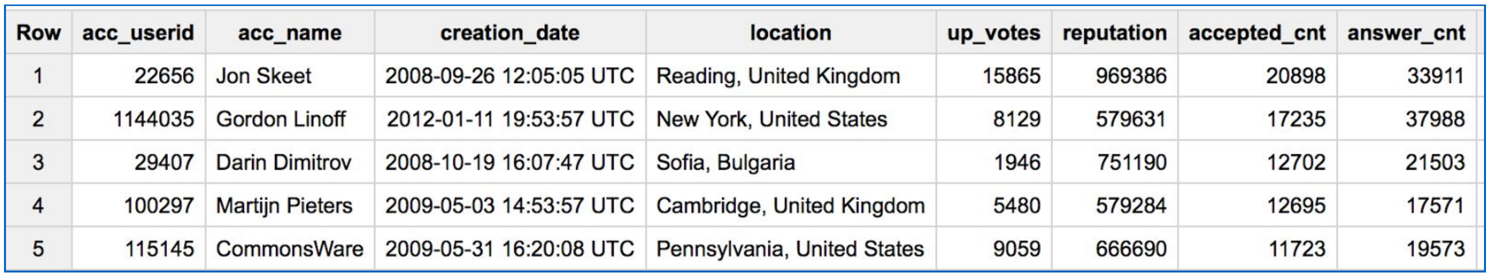
```
SELECT
  pa.owner_user_id AS userid, u.display_name, u.reputation, u.location, COUNT(pa.id) AS answer_cnt
FROM
  [instant-keel-185814:Stackoverflow.posts_answers] pa
JOIN
  [instant-keel-185814:Stackoverflow.users] u
ON pa.owner_user_id = u.id
WHERE YEAR(pa.creation_date)==2017
GROUP BY userid, u.display_name, pa.owner_user_id, u.reputation, u.location
ORDER BY answer_cnt DESC
LIMIT 20;
```

Finding 6: We found that Gordon Linoff is the one who answers the most questions (5940)

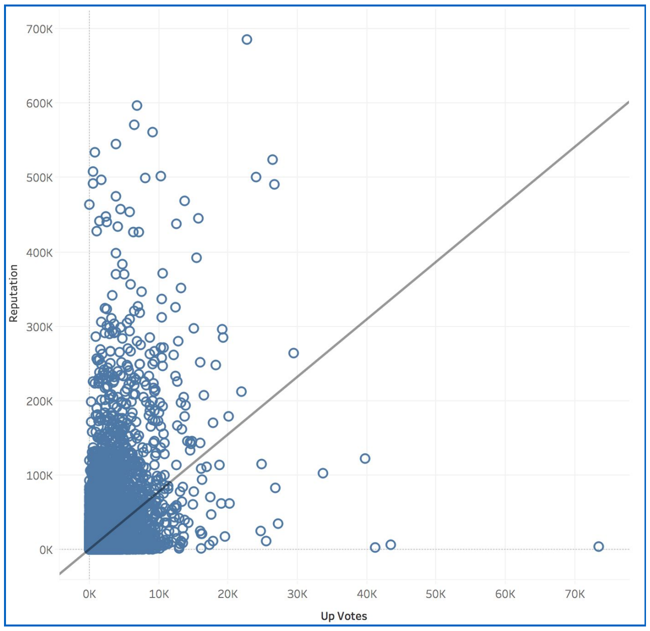
Row	userid	u_display_name	u_reputation	u_location	answer_cnt
1	1144035	Gordon Linoff	579631	New York, United States	5940
2	2901002	jezrael	153048	Bratislava, Slovakia	3084
3	3732271	akrun	285053		2632
4	3832970	Wiktor Stribiżew	211872	Warsaw, Poland	2084
5	2336654	piRSquared	84859	Bellevue, WA, United States	2042

4. As we can see, the users have attributes such as reputation and up vote. Reputation in stackoverflow is a rough measurement of how much the community trusts you. It can be gained primarily from posting good questions and answers. The upvote in stackoverflow is how the community indicates which questions and answers are most useful and appropriate. So for this part, we want to know whether people with more reputation will have more accepted answers, and we want to know whether people who has more upvotes will gain higher reputation.

```
SELECT acc.userid, acc.name, acc.u.creation_date AS creation_date, acc.u.location AS location,
acc.u.up_votes AS up_votes,
acc.u.reputation AS reputation, acc.acc_cnt AS accepted_cnt, a.a_cnt AS answer_cnt
FROM
(SELECT pa.owner_user_id AS userid, u.display_name AS name, u.creation_date, u.location,
u.up_votes, u.reputation, COUNT(pq.accepted_answer_id) AS acc_cnt
FROM [instant-keel-185814:Stackoverflow.posts_questions] pq
JOIN [instant-keel-185814:Stackoverflow.posts_answers] pa
ON pq.accepted_answer_id = pa.id
JOIN [instant-keel-185814:Stackoverflow.users] u
ON pa.owner_user_id   = u.id
GROUP BY userid, name, u.creation_date, u.location, u.up_votes, u.reputation
ORDER BY acc_cnt DESC) acc
JOIN
(SELECT pa.owner_user_id AS userid, COUNT(pa.id) AS a_cnt
FROM [instant-keel-185814:Stackoverflow.posts_answers] pa
GROUP BY userid
ORDER BY a_cnt DESC) a ON acc.userid=a.userid
ORDER BY acc.acc_cnt DESC;
```

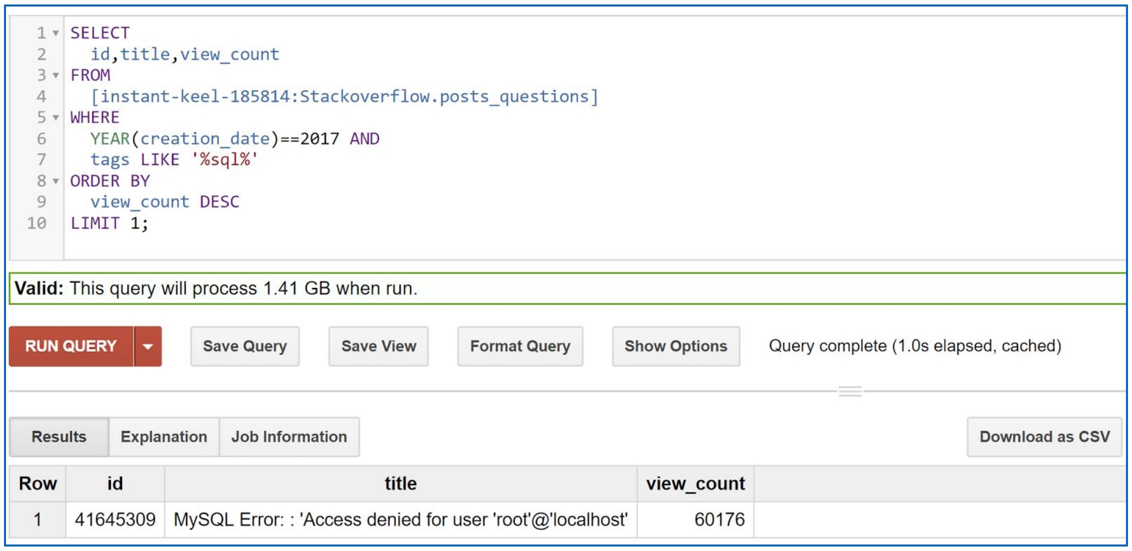


Row	acc_userid	acc_name	creation_date	location	up_votes	reputation	accepted_cnt	answer_cnt
1	22656	Jon Skeet	2008-09-26 12:05:05 UTC	Reading, United Kingdom	15865	969386	20898	33911
2	1144035	Gordon Linoff	2012-01-11 19:53:57 UTC	New York, United States	8129	579631	17235	37988
3	29407	Darin Dimitrov	2008-10-19 16:07:47 UTC	Sofia, Bulgaria	1946	751190	12702	21503
4	100297	Martijn Pieters	2009-05-03 14:53:57 UTC	Cambridge, United Kingdom	5480	579284	12695	17571
5	115145	CommonsWare	2009-05-31 16:20:08 UTC	Pennsylvania, United States	9059	666690	11723	19573



Finding 7: We found that people who have more accepted count tend to have higher reputation, and people who have more upvotes, in in general, tend to have higher reputation. However, there are outliers who have lots of upvotes but his/her reputation is low.

5. In the end, we want to know which SQL related question post has the highest view in 2017, and which hadoop related question post has the most favorite count.

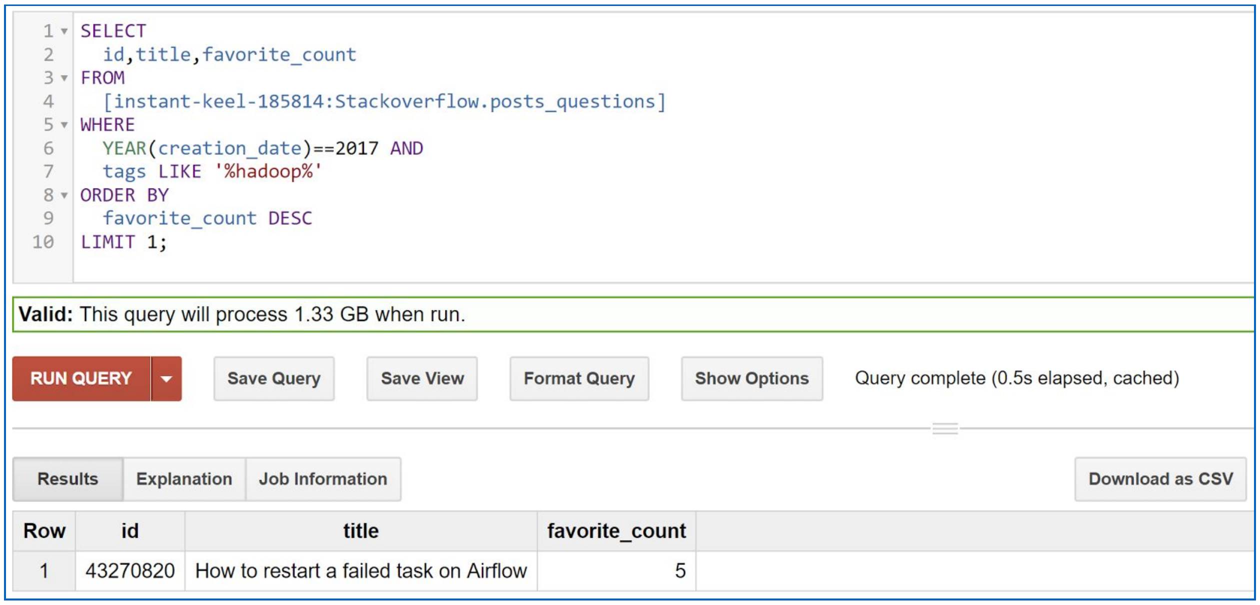


```
SELECT
  id,title,view_count
FROM
  [instant-keel-185814:Stackoverflow.posts_questions]
WHERE
  YEAR(creation_date)==2017 AND
  tags LIKE '%sql%'
ORDER BY
  view_count DESC
LIMIT 1;
```

Valid: This query will process 1.41 GB when run.

Query complete (1.0s elapsed, cached)

Row	id	title	view_count
1	41645309	MySQL Error: : 'Access denied for user 'root'@'localhost'	60176



```
SELECT
  id,title,favorite_count
FROM
  [instant-keel-185814:Stackoverflow.posts_questions]
WHERE
  YEAR(creation_date)==2017 AND
  tags LIKE '%hadoop%'
ORDER BY
  favorite_count DESC
LIMIT 1;
```

Valid: This query will process 1.33 GB when run.

Row	id	title	favorite_count
1	43270820	How to restart a failed task on Airflow	5

Finding 8: We found that the most viewed SQL question post is “Access denied for user ‘root’ @’localhost’ with 60,176 views and the most favorite hadoop question post is asking “How to restart a failed task on Airflow” with a favorite count of only 5. So after all, there are more people in the world asking and answering questions related to SQL than hadoop.

IV. Conclusion

In this project, we took a quick tour of how to use Google BigQuery, and applied stackoverflow dataset for querying on this google platform. Since we want to know where we stand in the market with what we have learned in the Managing Big Data class, we explore the stackoverflow dataset and have several interesting findings. We found that SQL is still one of the most common keyword in the big data circle. AWS is getting its attention over the years. Access is still having its share in the marketplace. There are more people asking and answering questions about SQL, but few show their interests in hadoop, hive, pig...etc. So this actually implies that there is a shortage of human resources with expertise in such area. Overtime, as cloud platform and Big data become more popular, we expect the big data tools that we learned to be the common topic on stackoverflow.

V. References

Google Cloud. (n.d.). Quickstart Using the Web UI | BigQuery | Google Cloud Platform. Retrieved from https://cloud.google.com/bigquery/quickstart-web-ui

MSV, Janakiram. (Jul 10, 2014). When to use Google BigQuery? Big Data in the Cloud. *CloudAcademy Blog*. Retrieved from: https://cloudacademy.com/blog/when-to-use-google-bigquery/

Sato, Kazunori. (2012). An Inside Look at Google BigQuery. *Google*. Retrieved form: https://cloud.google.com/files/BigQueryTechnicalWP.pdf

Stackoverflow. (n.d.). Privileges - vote up. Retrieved from https://stackoverflow.com/help/privileges/vote-up

Stackoverflow. (n.d.). What is reputation? How do I earn (and lose) it? - Help Center. Retrieved from https://stackoverflow.com/help/whats-reputation