# CIS5200 Term Project Tutorial https://avatars2.githubusercontent.com/u/4156894?v=3&s=100

#### Authors: Divya Pakhale, Sanchita Gawand, Siddhi Udani, Tanvi Gawade

#### Instructor: [Jongwook Woo](https://www.linkedin.com/in/jongwook-woo-7081a85)

#### Date: 12/10/2018

**Lab Tutorial**

Divya Pakhale ([dpakhal@calstatela.edu](mailto:dpakhal@calstatela.edu))

Sanchita Gawand ([sgawand@calstatela.edu](mailto:sgawand@calstatela.edu))

Siddhi Udani ([sudani2@calstatela.edu](mailto:sudani2@calstatela.edu))

Tanvi Gawade ([tgawade@calstatela.edu](mailto:tgawade@calstatela.edu))

12/10/2018

**New York Times Data Analysis using Hive**

**OBJECTIVE**

The New York Times (NYT) has a large reader base and plays an important role in shaping public opinion and outlook on current affairs and in setting the tone of the public discourse, especially in the U.S. The comments sections for articles in the NYT are quite active and gives insights to reader’s opinion on the subject matter of the articles. Each comment can receive other reader’s recommendations in the form of upvotes. This project aims at performing data analysis and sheds lights on New York Times Dataset using HIVEQL queries and presenting visualization to see the insights using Power BI, Tableau, & Excel 3-D Maps.

* To find out count of document type by type of material for the year 2017 and 2018.
* To determine reply count for the document type month wise for year 2017 and 2018.
* To find out reply count for each comment type for year 2017 and 2018.
* To uncover frequency of type of material with respect to article word count for both the years.
* To find out the degree of polarity to reveal the most positive as well negative headlines for the year 2017 based on public comments.
* To identify the most common words in the headlines for article year 2017 and 2018.
* Geo map to show recommendations by the user’s location for the year 2017 and 2018.
* Also, to show a comparative analysis for the above objectives between the year 2017 and 2018.

**INTRODUCTION**

This Project aims at performing data analysis and providing insights on New York Times Comments (NYT) using HIVE and presenting the visualization in Tableau and Microsoft Power BI.

In this hands-on lab, you will learn how to:

* Load data from local desktop(windows) to Linux shell.
* Download and upload files to HDFS.
* Extract TXT file using Hive.
* Data cleaning using Hive.
* Create Hive tables to query the NYT dataset for analysis.
* Create Hive queries to analyze the sentiment of data
* Use Tableau, Power BI, Excel 3D Maps for visualization of the analyzed data.

**PLATFORM SPECIFICATIONS**

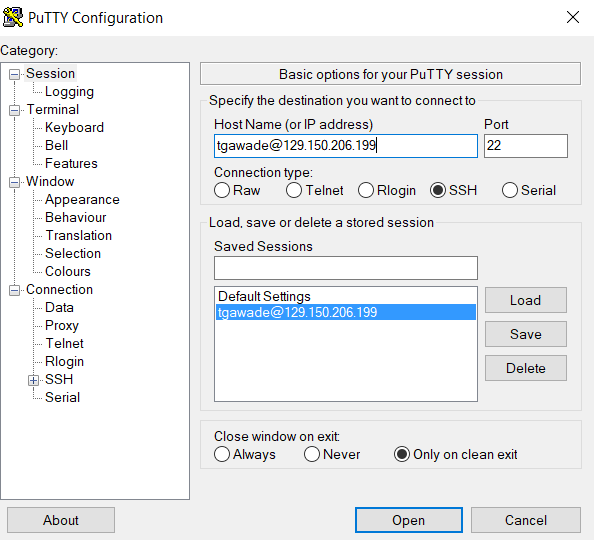
* Oracle Big Data Compute Edition: 5 nodes
* CPUs: 10
* CPU speed: 2.20GHz
* Memory: 150 GB
* Storage: 678 BG
* HDFS Capacity: 147 GB

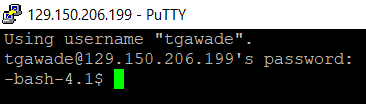
**PREREQUISITES**

* You must have Microsoft Excel 2010, 2013 or 2016 installed.
* You must have your Excel 3D-Map enabled.
* Tableau 10.3 installed for visualization of the analyzed data.
* Power BI Desktop Version
* Oracle Big Data Compute Edition: 5 nodes

**DOWNLOAD THE DATASET**

This step is to get data manually. You need to remotely access your Oracle Cloud Big Data Compute Editions that you executed in your Oracle Cloud account using ssh using the information - ip address and connect command in beeline CLI-





1. ArticleYear2017-<https://raw.githubusercontent.com/tanvigawade/April2017/master/ArticleYear2017.txt>
2. ArticleYear2018- <https://raw.githubusercontent.com/tanvigawade/April2017/master/ArticleYear2018.txt>
3. CommentYear2017- <https://www.dropbox.com/s/v0zqfog8pmque6g/CommentYear2017%20.txt?dl=0>
4. CommentYear2018- <https://www.dropbox.com/s/mj4by2421kptba2/CommentYear2018.txt?dl=0>

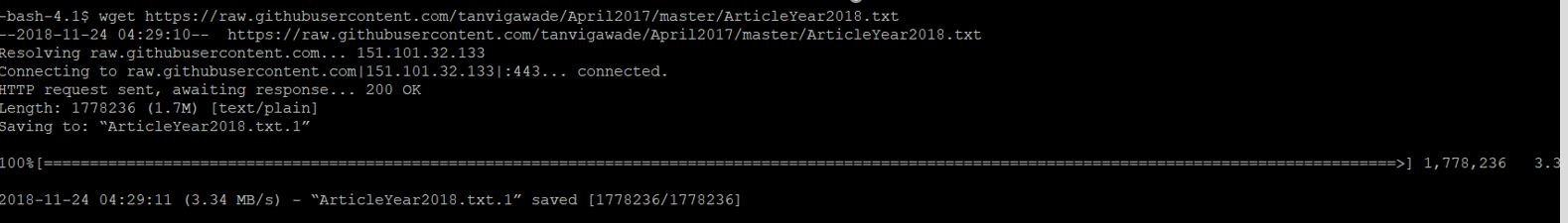
**UPLOAD TXT FILE TO HADOOP DIRECTORY**

Before uploading the TXT file to Hadoop directory, we need to first transfer it to local directory using below commands.

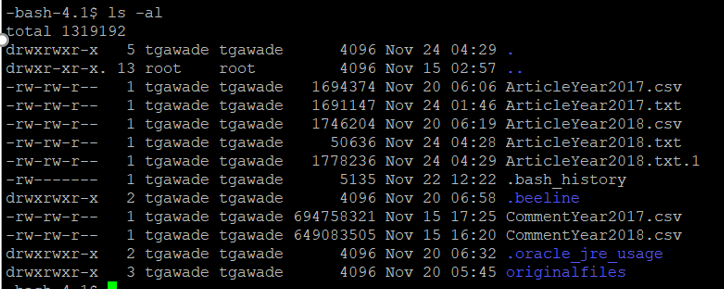
Note: Change the path and username.

wget https://raw.githubusercontent.com/tanvigawade/April2017/master/ArticleYear2017.txt

wget https://raw.githubusercontent.com/tanvigawade/April2017/master/ArticleYear2018.txt

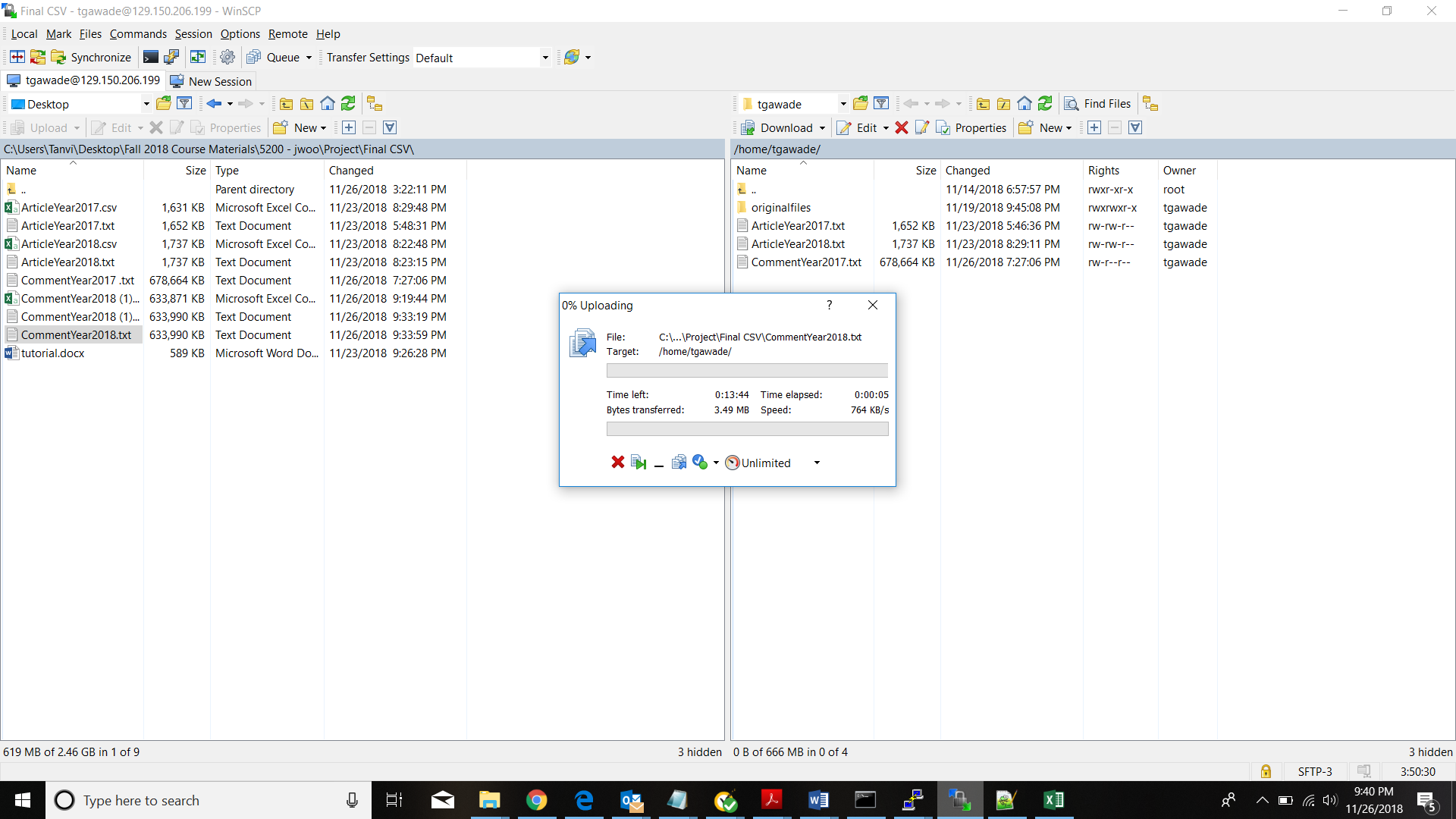


$ ls -al



Repeat “Step 2” for ArticleYear2018.

Since the CommentYear2017 and CommentYear2018 are more than 25MB we downloaded the datasets to local directory using WinSCP software. We also uploaded the dictionary data set using the same methodology.



Now we have to upload all the TXT files to HDFS folder. Run the following HDFS commands to create and list the a1,a2,c1,c2,d1 and d directories in HDFS.

Hdfs dfs -mkdir /user/tgawade/a1

Hdfs dfs -mkdir /user/tgawade/a2

Hdfs dfs -mkdir /user/tgawade/c1

Hdfs dfs -mkdir /user/tgawade/c2

Hdfs dfs -mkdir /user/tgawade/d1

hdfs dfs -put CommentYear2017.txt /user/tgawade/c1/

hdfs dfs -put CommentYear2018.txt /user/tgawade/c2/

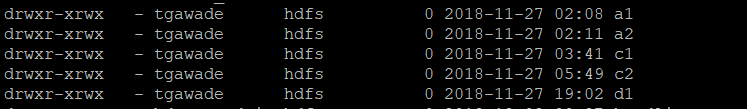
hdfs dfs -put ArticleYear2017.txt /user/tgawade/a1/

hdfs dfs -put ArticleYear2018.txt /user/tgawade/c2/

hdfs dfs -put dictionary.txt /user/tgawade/d1/

Now, you have to upload this CSV file to hdfs folder. Run the following HDFS commands to create and list the h1b\_data directory in HDFS





Give permissions

Run the following HDFS command to make your beeline command works:

-bash-4.1$ hdfs dfs -chmod -R o+w /user/tgawade/c1/

-bash-4.1$ hdfs dfs -chmod -R o+w /user/tgawade/c2/

-bash-4.1$ hdfs dfs -chmod -R o+w /user/tgawade/a1/

-bash-4.1$ hdfs dfs -chmod -R o+w /user/tgawade/a1/

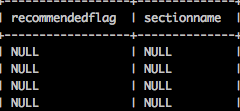
-bash-4.1$ hdfs dfs -chmod -R o+w /user/tgawade/d1/

**DATA CLEANING**

**Removing Null Values**

Null values were removed from tables. For example, section name and replycount columns had null values as shown below:

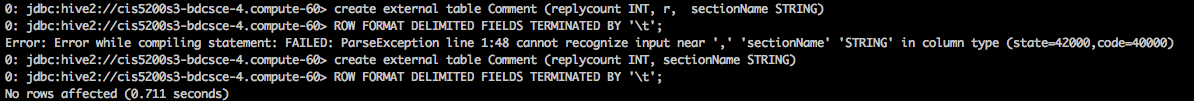
**Before:**

****

Below steps were performed to remove null values

**Step 1:**

|  |
| --- |
| #External Table was created  create external table Comment (replycount INT, sectionName STRING)  ROW FORMAT DELIMITED FIELDS TERMINATED BY '\t'; |

****

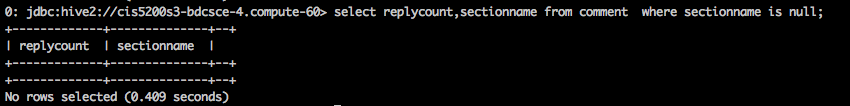
**Step 2:**

|  |
| --- |
| #Inserted data from original table  INSERT OVERWRITE TABLE Comment  Select replycount, sectionname  From commentyear2017  where replycount is not null and sectionname is not null; |

****

**All null values from columns were removed. Rechecked using below query:**

|  |
| --- |
| select replycount,sectionname from comment where sectionname is null; |

****

**CREATE HIVE TABLE TO QUERY NEW YORK TIMES DATA**

Open beeline CLI (Command Line Shell Interface) that is equivalent to hive CLI environment as follows, which you have done in the previous lab.

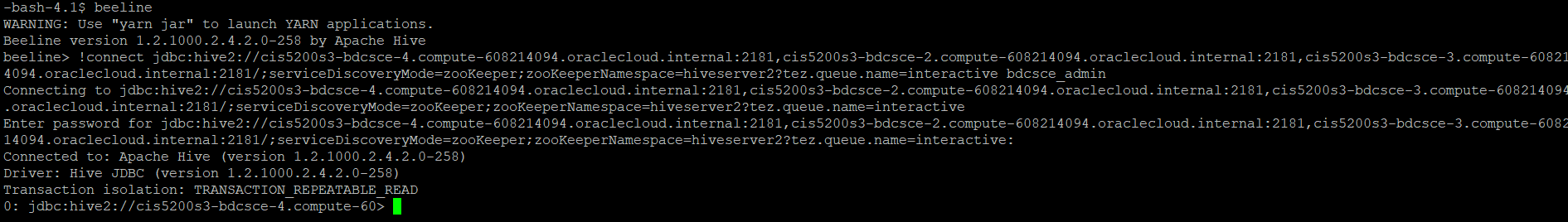
beeline

Beeline is for multiple user’s access to Hive Server 2 of a Hadoop cluster.

Use the below command to connect to beeline:

!connect jdbc:hive2://cis5200s3-bdcsce-4.compute-608214094.oraclecloud.internal:2181,cis5200s3-bdcsce-2.compute-608214094.oraclecloud.internal:2181,cis5200s3-bdcsce-3.compute-608214094.oraclecloud.internal:2181/;serviceDiscoveryMode=zooKeeper;zooKeeperNamespace=hiveserver2?tez.queue.name=interactive bdcsce\_admin.

NOTE: If you see “CLOSED” in the above beeline shell prompt, it is not connected to Hive Server 2.

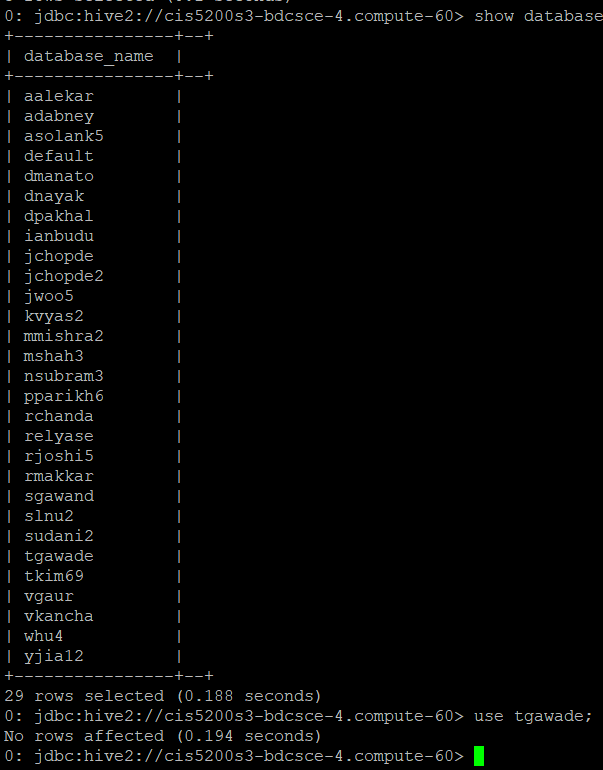


NOTE: Now we have to create your own database with your username to separate your tables with other users you have to use your username. For example, the user should run the following command.

0: jdbc:hive2://cis5200s3-bdcsce-4.compute-60>CREATE DATABASE tgawade;

No rows affected (0.277 seconds)

0: jdbc:hive2://cis5200s3-bdcsce-4.compute-60> show databases;



The following hive statement creates an external table for ArticleYear2017, ArticleYear2018. External tables preserve the data in the original file format, while allowing Hive to perform queries against the data within the file.

In the hive shell CLI, you need to copy and paste the following HiveQL code to create an external table CommentYear2017.

|  |
| --- |
| create external table if not exists CommentYear2017(Month\_Name STRING,approveDate STRING,articleID STRING,articleWordCount BIGINT,commentBody STRING,commentID STRING,commentSequence STRING,commentTitle STRING,commentType STRING,createDate STRING,depth INT,editorsSelection INT,inReplyTo STRING,newDesk STRING,parentID STRING,parentUserDisplayName STRING, permID STRING, picURL STRING, printPage INT, recommendations INT, recommendedFlag INT, replyCount INT, reportAbuseFlag INT, sectionName STRING, sharing INT, status STRING, timespeople INT, trusted INT, updateDate STRING, userDisplayName STRING, userID STRING, userLocation STRING, userTitle STRING, userURL STRING,typeofmaterial STRING)  ROW FORMAT DELIMITED FIELDS TERMINATED BY '\t'  STORED AS TEXTFILE location "/user/tgawade/c1/"  TBLPROPERTIES ('skip.header.line.count'='1'); |

In the hive shell CLI, you need to copy and paste the following HiveQL code to create an external table CommentYear2018.

create external table if not exists CommentYear2018(Month\_Name STRING,approveDate STRING,articleID STRING,articleWordCount BIGINT,commentBody STRING,commentID STRING,commentSequence STRING,commentTitle STRING,commentType STRING,createDate STRING,depth INT,editorsSelection INT,inReplyTo STRING,newDesk STRING,parentID STRING,parentUserDisplayName STRING, permID STRING, picURL STRING, printPage INT, recommendations INT, recommendedFlag INT, replyCount INT, reportAbuseFlag INT, sectionName STRING, sharing INT, status STRING, timespeople INT, trusted INT, updateDate STRING, userDisplayName STRING, userID STRING, userLocation STRING, userTitle STRING, userURL STRING,typeofmaterial STRING)

ROW FORMAT DELIMITED FIELDS TERMINATED BY '\t'

STORED AS TEXTFILE location "/user/tgawade/c2/"

TBLPROPERTIES ('skip.header.line.count'='1');

In the hive shell CLI, you need to copy and paste the following HiveQL code to create an external table ArticleYear2017.

create external table if not exists articleyear2017(Month\_Name STRING,articleID STRING,abstract STRING,byline STRING,documentType STRING,headline STRING,keywords STRING,multimedia INT,

newDesk STRING,printPage INT,pubDate TIMESTAMP,source STRING,

typeOfMaterial STRING,

webURL STRING,

articleWordCount BIGINT)

ROW FORMAT DELIMITED FIELDS TERMINATED BY '\t'

STORED AS TEXTFILE location "/user/tgawade/a1/"

TBLPROPERTIES ('skip.header.line.count'='1');

In the hive shell CLI, you need to copy and paste the following HiveQL code to create an external table ArticleYear2018**.**

create external table if not exists articleyear2018(Month\_Name STRING,articleID STRING,abstract STRING,byline STRING,documentType STRING,headline STRING,keywords STRING,multimedia INT,

newDesk STRING,printPage INT,pubDate TIMESTAMP,source STRING,

typeOfMaterial STRING,

webURL STRING,

articleWordCount BIGINT)

ROW FORMAT DELIMITED FIELDS TERMINATED BY '\t'

STORED AS TEXTFILE location "/user/tgawade/a2/"

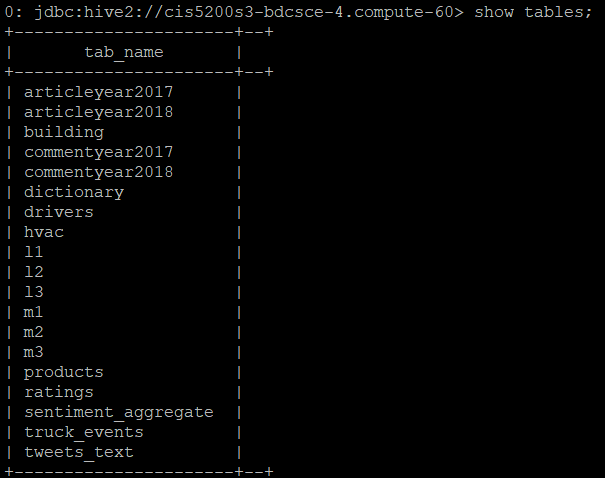
TBLPROPERTIES ('skip.header.line.count'='1');

In the hive shell CLI, you need to copy and paste the following HiveQL code to create an external table dictionary.

|  |
| --- |
| CREATE EXTERNAL TABLE if not exists dictionary (type string,length int,word string,pos string,  stemmed string,  polarity string )  ROW FORMAT DELIMITED  FIELDS TERMINATED BY '\t'  STORED AS TEXTFILE  LOCATION "/user/tgawade/d1/" |

Now you may see if those tables are created with “show tables”:

0: jdbc:hive2://cis5200-bdcsce-4.compute-6082> show tables;



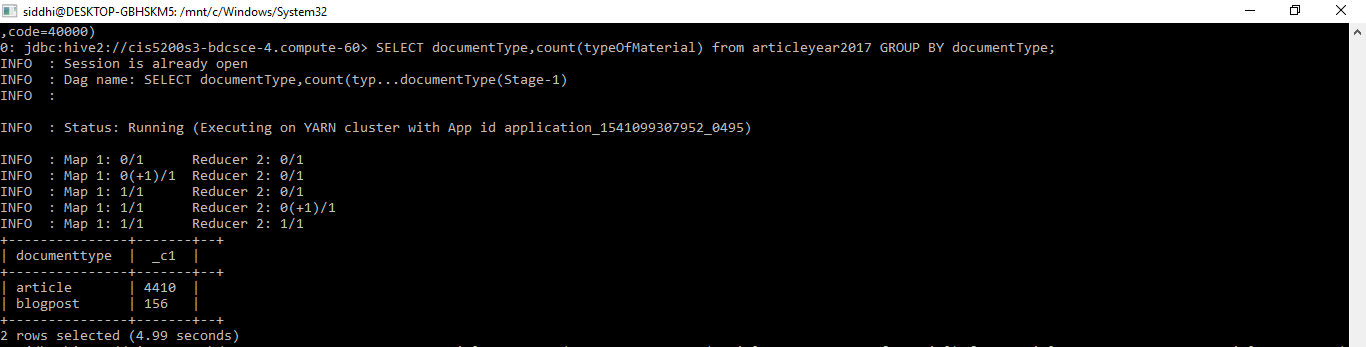
**QUERYING ON THE DATASET**

**Query 1: Show the count of document type by type of material for the year 2017 and 2018?**

In this query, we have tried to determine what number of articles and blogpost are present in NYT for both the years respectively.

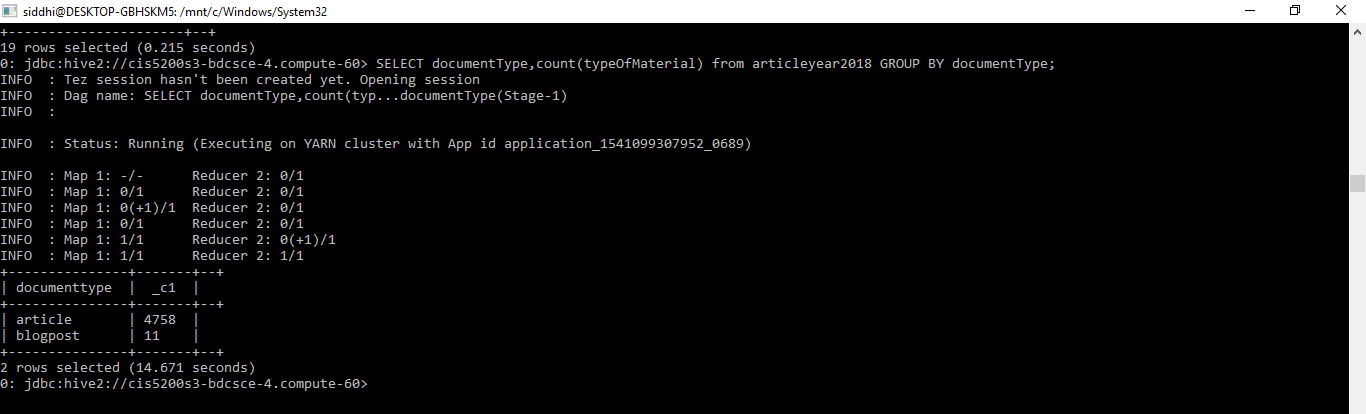
**For year 2017:**

|  |
| --- |
| SELECT documentType,count(typeOfMaterial) from articleyear2017 GROUP BY documentType; |



**For year 2018:**

|  |
| --- |
| SELECT documentType, count(typeOfMaterial) from articleyear2018 GROUP BY documentType; |



**Query 2: What is the reply count for the document type month wise for year 2017 & 2018?**

Below query shows the reply count for each document type for month January to June for year 2017 & 2018. Month name is from articleyear2017 and reply count is from commentyear2017 table. Left outer join is used to get the desired output and is Grouped by month.

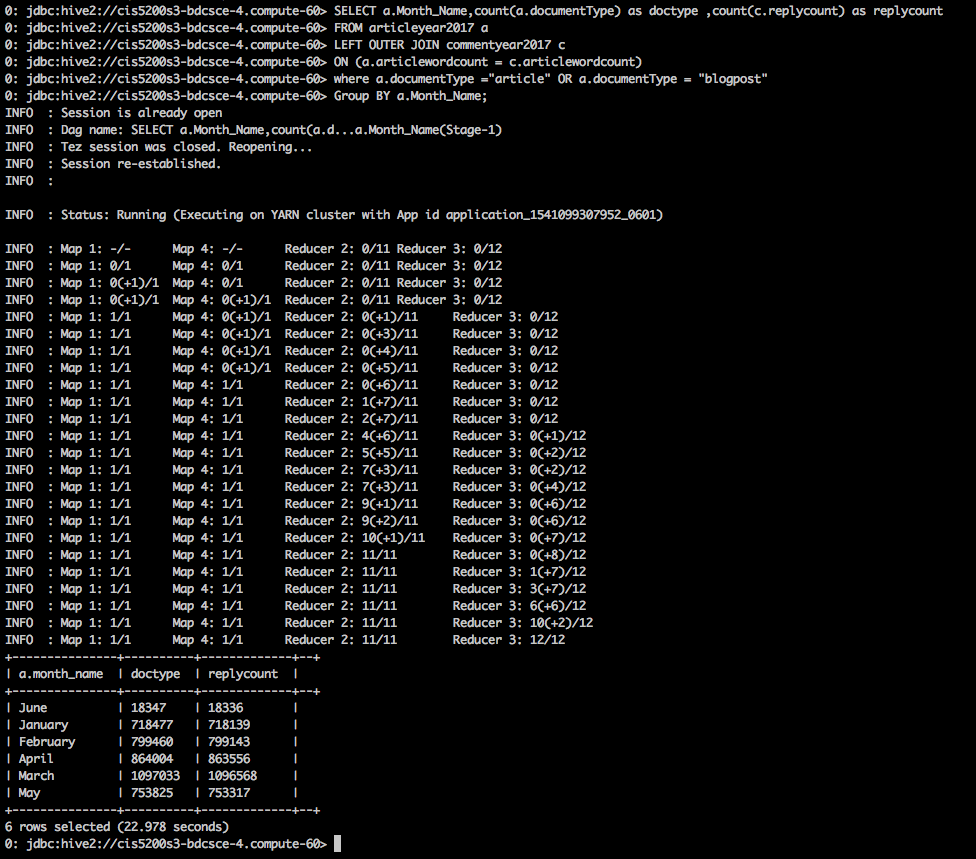
**For Year 2017:**

SELECT a.Month\_Name,count(a.documentType) as doctype ,count(c.replycount) as replycount

FROM articleyear2017 a  
LEFT OUTER JOIN commentyear2017 c   
ON (a.articlewordcount = c.articlewordcount)

where a.documentType ="article" OR a.documentType = "blogpost"

Group BY a.Month\_Name;

****

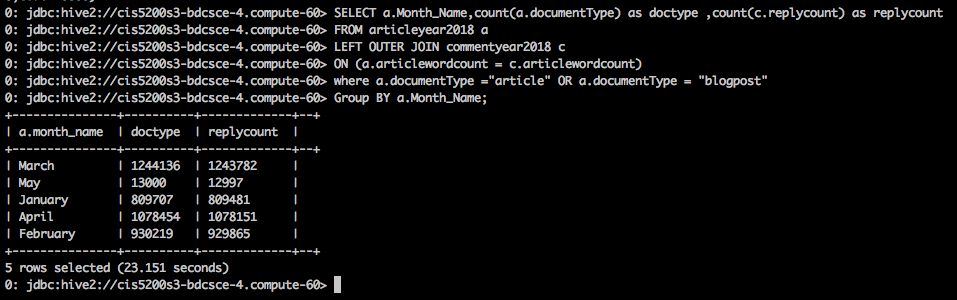
**For Year 2018:**

SELECT a.Month\_Name,count(a.documentType) as doctype ,count(c.replycount) as replycount

FROM articleyear2018 a  
LEFT OUTER JOIN commentyear2018 c   
ON (a.articlewordcount = c.articlewordcount)

where a.documentType ="article" OR a.documentType = "blogpost"

**Group BY a.Month\_Name;**

****

**Query 3: What is the reply count for each comment type.**

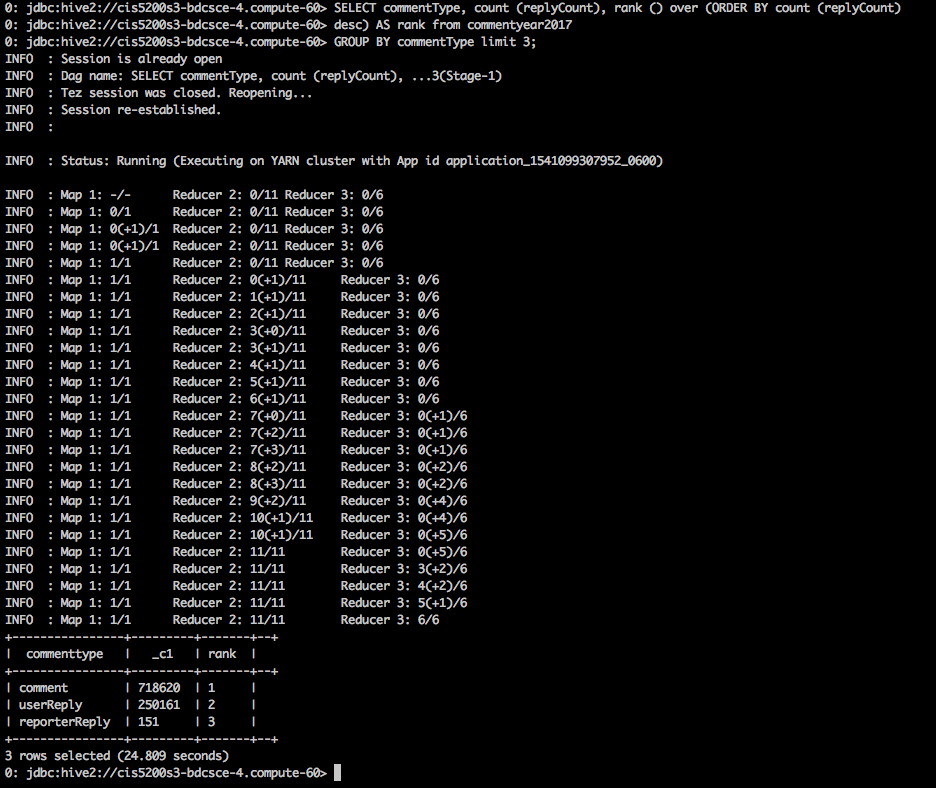
Below query shows the reply count received for top 3 comment type for year 2017 & 2018. Rank is used to get the desired output serially and is Grouped by comment type.

**For Year 2017:**

SELECT commentType, count (replyCount), rank () over (ORDER BY count (replyCount)

desc) AS rank from commentyear2017

GROUP BY commentType limit 3;

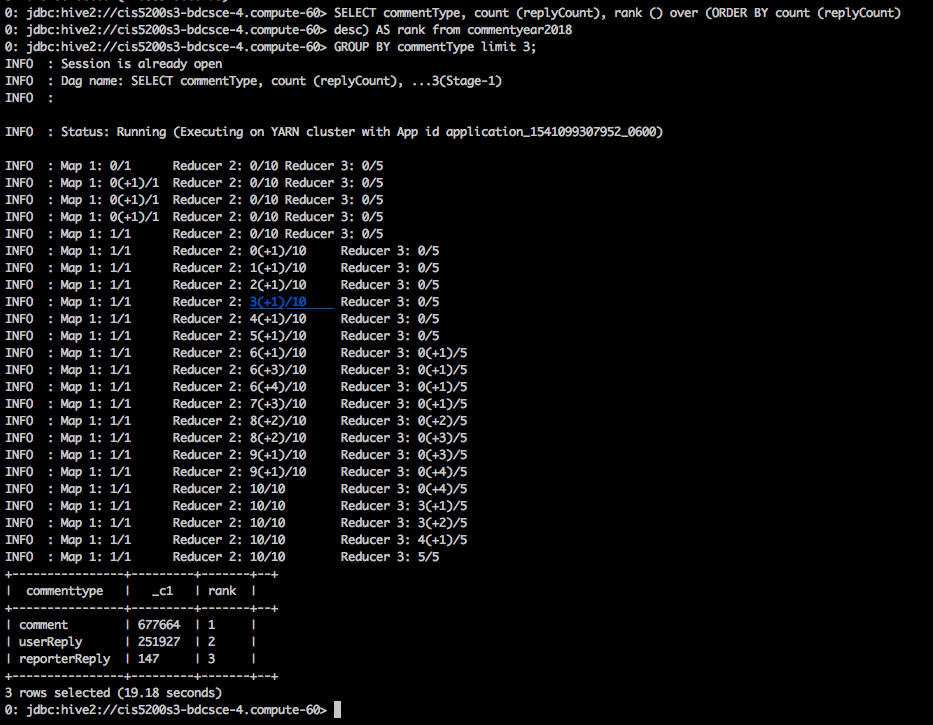
****

**For Year 2018:**

SELECT commentType, count (replyCount), rank () over (ORDER BY count (replyCount)

desc) AS rank from commentyear2018

GROUP BY commentType limit 3;

****

**Query 4: What is the count of new desk month wise?**

NewDesk is a column which has various field values like letter, foreign, editorial, brief, etc. In this query we have tried to find out the count of NewDesk received for both the years month wise.

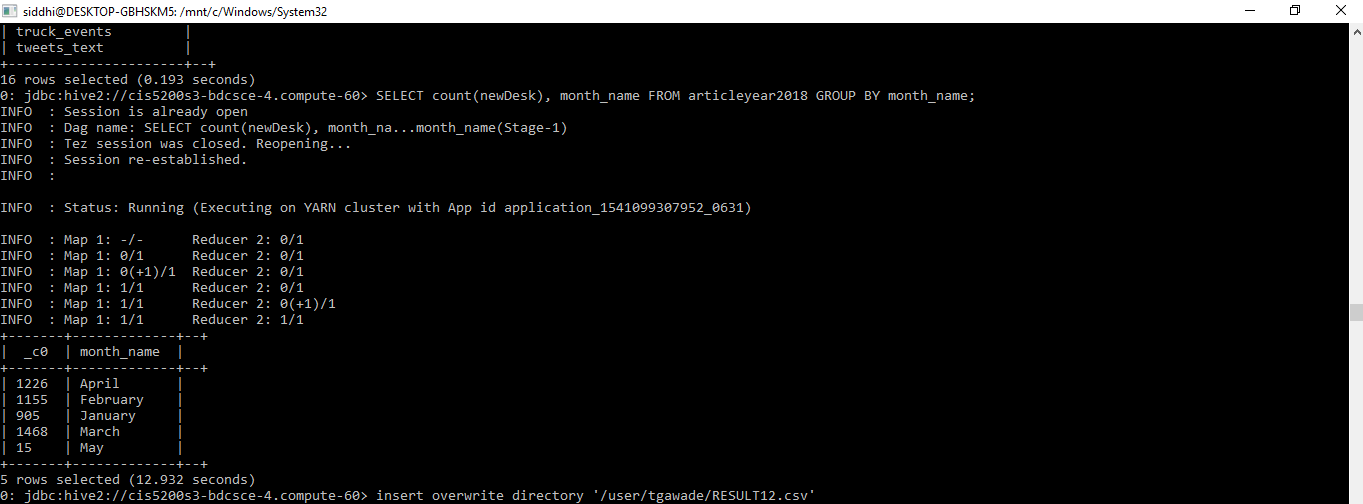
**For year 2017:**

|  |
| --- |
| SELECT count(newDesk),month\_name FROM articleyear2017 GROUP BY month\_name; |



**For year 2018:**

|  |
| --- |
| SELECT count(newDesk),month\_name FROM articleyear2018 GROUP BY month\_name; |

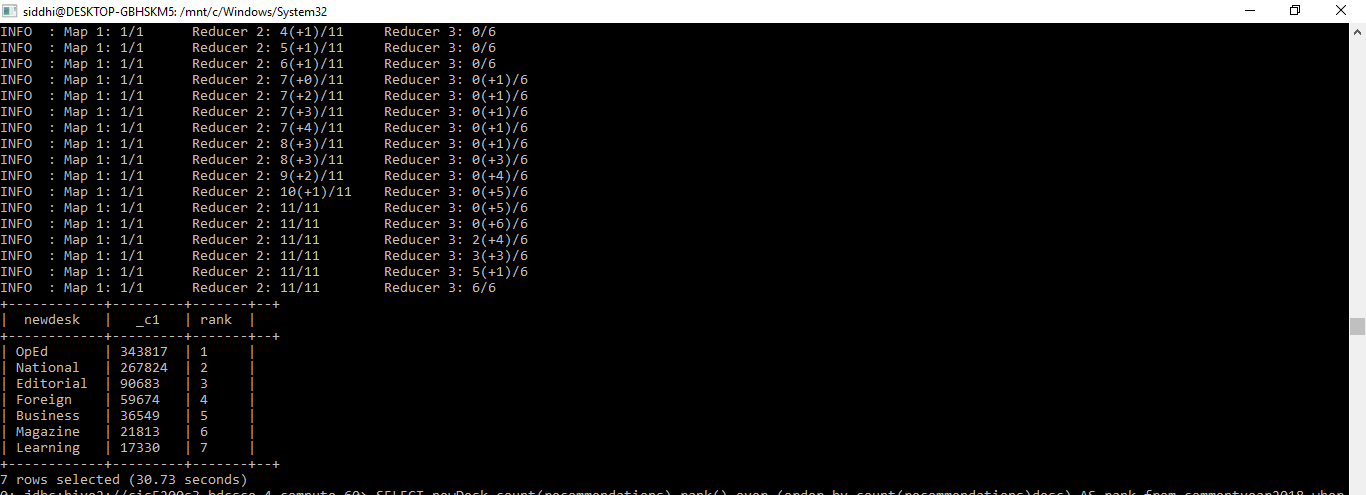


**Query 5: What is the count of new desk based on recommendations?**

As explained above that newDesk has various filed values and each of them receive some sort of recommendations from the people, which we have shown in the query below for both the years.

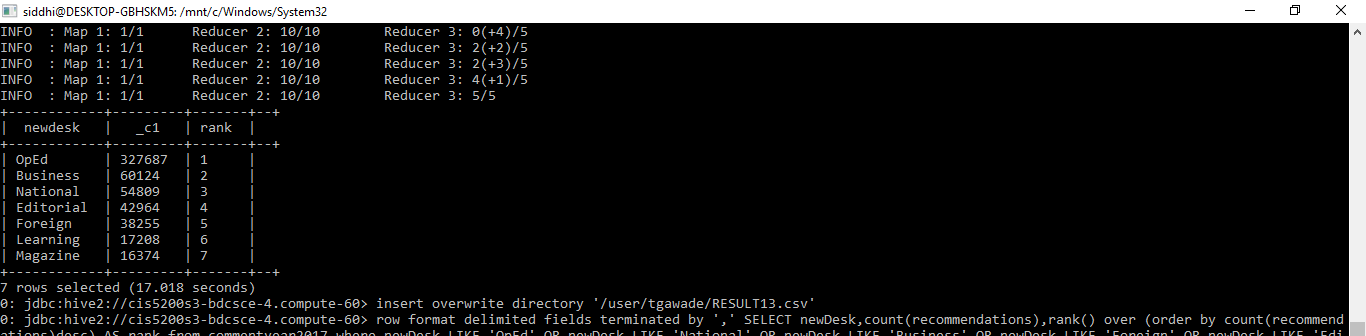
**For year 2017:**

|  |
| --- |
| SELECT newDesk,count(recommendations),rank() over (order by count(recommendations)desc) AS rank from commentyear2017 where newDesk LIKE 'OpEd' OR newDesk LIKE 'National' OR newDesk LIKE 'Business' OR newDesk LIKE 'Foreign' OR newDesk LIKE 'Editorial' OR newDesk LIKE 'Magazine' OR newDesk LIKE 'Learning' GROUP BY newDesk; |



**For year 2018:**

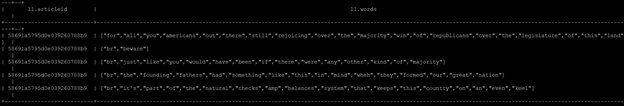
|  |
| --- |
| SELECT newDesk,count(recommendations),rank() over (order by count(recommendations)desc) AS rank from commentyear2018 where newDesk LIKE 'OpEd' OR newDesk LIKE 'National' OR newDesk LIKE 'Business' OR newDesk LIKE 'Foreign' OR newDesk LIKE 'Editorial' OR newDesk LIKE 'Magazine' OR newDesk LIKE 'Learning' GROUP BY newDesk; |



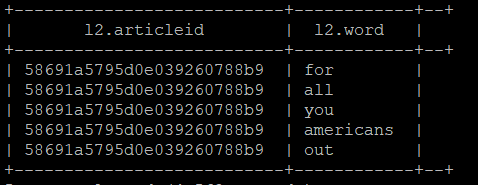
**Query 6: What is the degree of polarity by most positive headlines for the year 2017?**

Here we created a view using the function Sentences() which splits the string present in the comment body into arrays of sentences , where each sentence is an array of words. You have your data set as arrays of words which are then lateral view exploded at the first level using the function Explode().

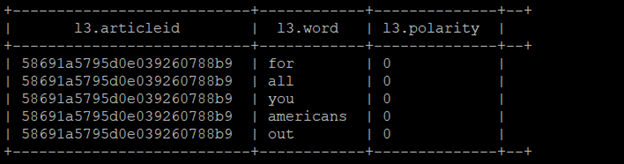
|  |
| --- |
| create view IF NOT EXISTS l1 as  select articleid,words  from commentyear2017  lateral view explode(sentences(lower(commentbody))) dummy as words; |

****

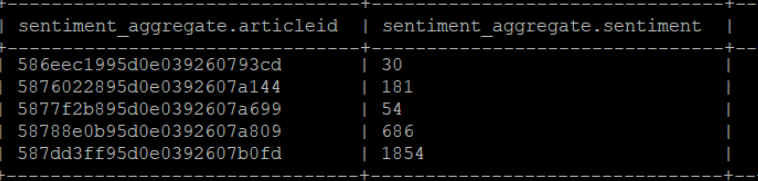
|  |
| --- |
| create view IF NOT EXISTS l2 as  select articleid, word  from l1  lateral view explode(words) dummy as word; |

****

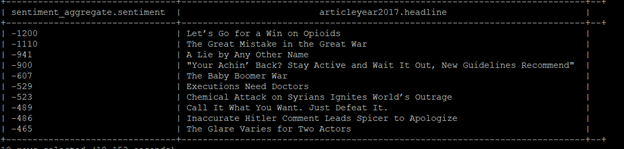
|  |
| --- |
| create view IF NOT EXISTS l3 as select  articleid,  l2.word,  case d.polarity  when 'negative' then -1  when 'positive' then 1  else 0 end as polarity  from l2 left outer join dictionary d on l2.word = d.word; |

****

|  |
| --- |
| create table IF NOT EXISTS sentiment\_aggregate  stored as orc as select  articleid,sum( polarity ) sentiment  from l3 group by articleid; |

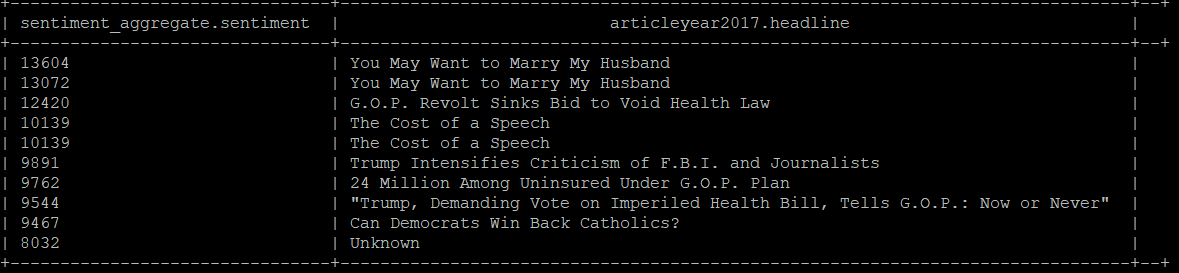
****

|  |
| --- |
| select sentiment\_aggregate.sentiment,articleyear2017.headline from articleyear2017 inner join sentiment\_aggregate on sentiment\_aggregate.articleid=articleyear2017.articleid order by sentiment asc limit 10; |

****

**Query 7: What is the degree of polarity by most negative headlines?**

|  |
| --- |
| select sentiment\_aggregate.sentiment,articleyear2017.headline from articleyear2017 inner join sentiment\_aggregate on sentiment\_aggregate.articleid=articleyear2017.articleid order by sentiment desc limit 10; |

****

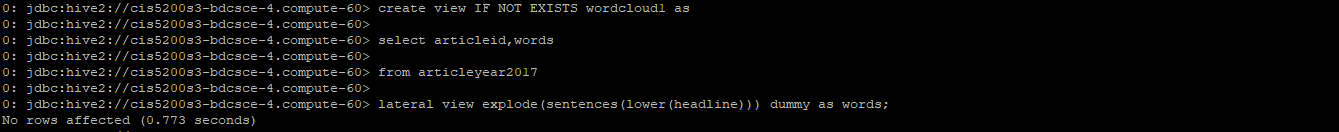
**Query 8: What are the most common words in the headlines for article year 2017?**

Here we created a view using the function Sentences() which splits the string present in the comment body into arrays of sentences , where each sentence is an array of words. You have your data set as arrays of words which are then lateral view exploded at the first level using the function Explode().

In this query, we are counting the most common words used in the headlines section.

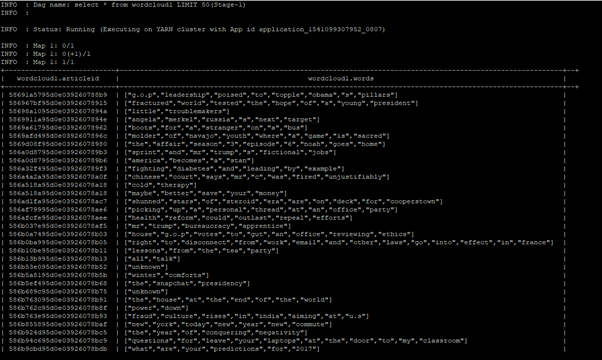
|  |
| --- |
| create view IF NOT EXISTS wordcloud1 as  select articleid,words  from articleyear2017  lateral view explode(sentences(lower(headline))) dummy as words; |

**OUTPUT:**



|  |
| --- |
| Select \* from wordcloud1 LIMIT 50; |

**OUTPUT:**



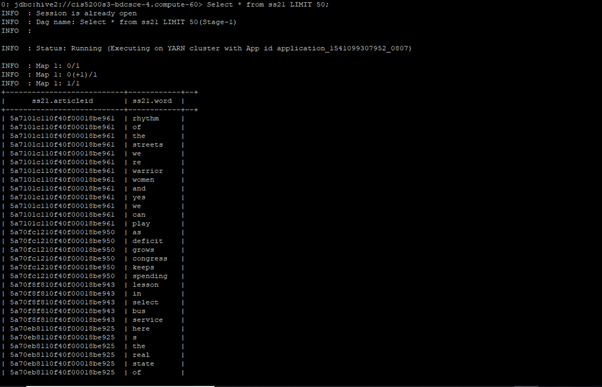
|  |
| --- |
| create view IF NOT EXISTS ss21 as  select articleid,word  from wordcloud  lateral view explode(words) dummy as word; |

**OUTPUT:**



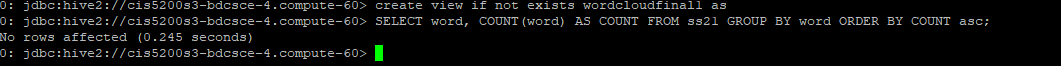
|  |
| --- |
| Select \* from ss21 LIMIT 50; |

**OUTPUT:**

****

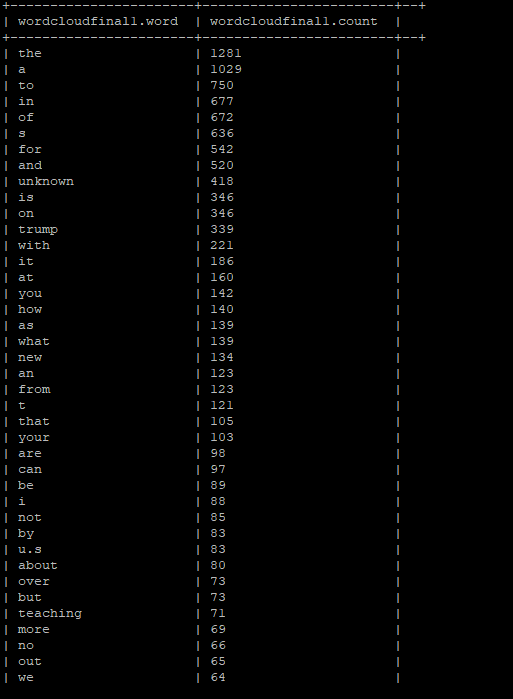
|  |
| --- |
| create view if not exists wordcloudfinal1 as  SELECT word, COUNT(word) AS COUNT FROM ss21 GROUP BY word ORDER BY COUNT asc; |

**OUTPUT**



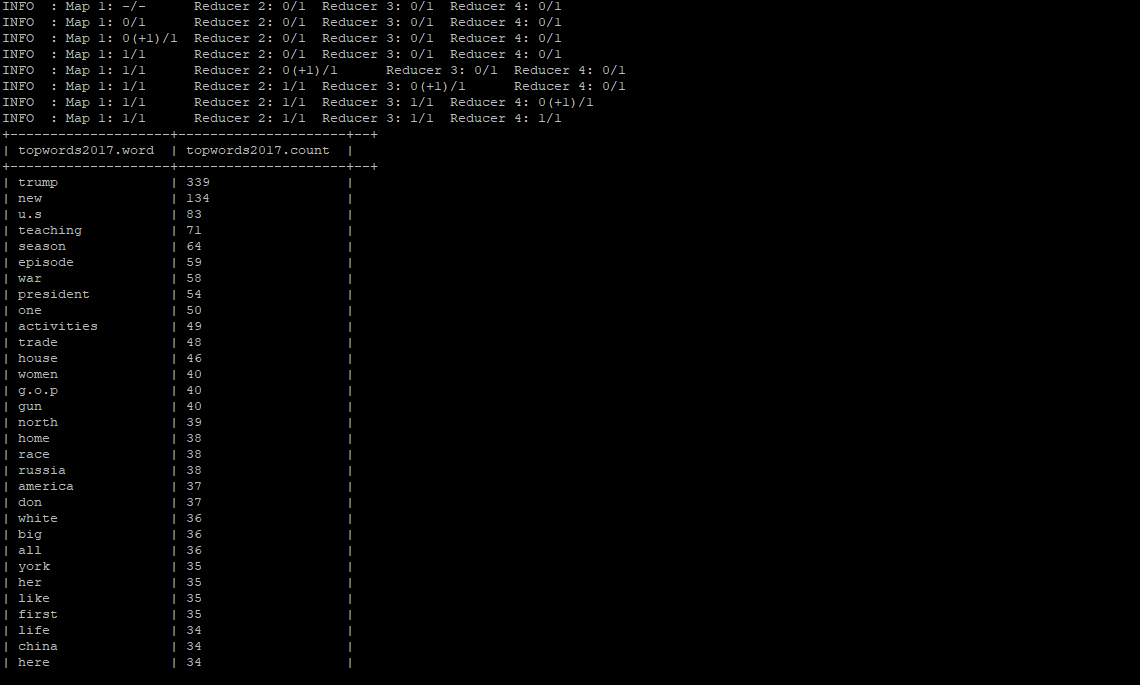
|  |
| --- |
| select \* from wordcloudfinal1 order by count desc limit 100; |

**OUTPUT**



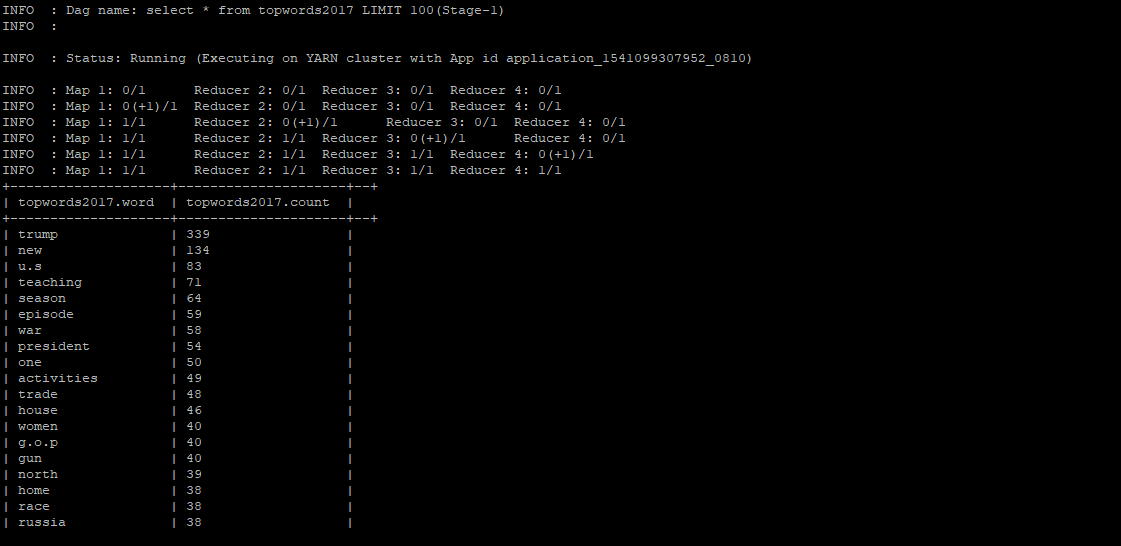
|  |
| --- |
| create view if not exists topwords2017 as select \* from wordcloudfinal1 where not word in('the','a','to','in','of','s','for','and','unknown','is','on','from','by','i','l','t','with','it','at','you','how','as','what','an','that','your','are','can','be','not','about','but','no','out','we','over','more','now','has','who','up','this','will','do','his','he','after','may','why','when','was','into','get','its','my','or','says','should','they','have','our','1') order by count desc limit 100; |

**OUTPUT**



|  |
| --- |
| Select \* from topwords2017 LIMIT 100; |

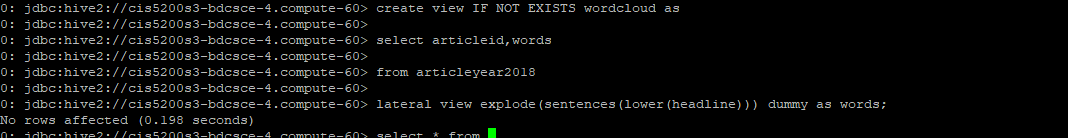
**OUTPUT**



**Query 9: What are the most common words in the headlines for article year 2018?**

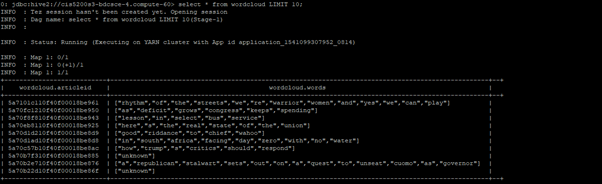
|  |
| --- |
| create view IF NOT EXISTS wordcloud as  select articleid,words  from articleyear2018  lateral view explode(sentences(lower(headline))) dummy as words; |

**OUTPUT**



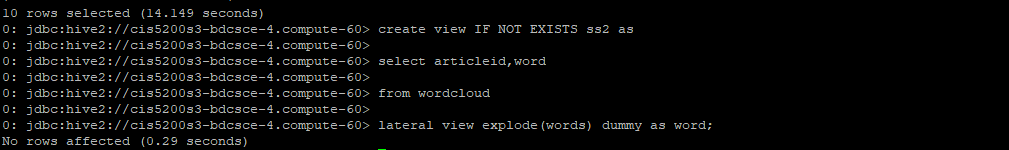
|  |
| --- |
| Select \* from wordcloud LIMIT 10; |

**OUTPUT**



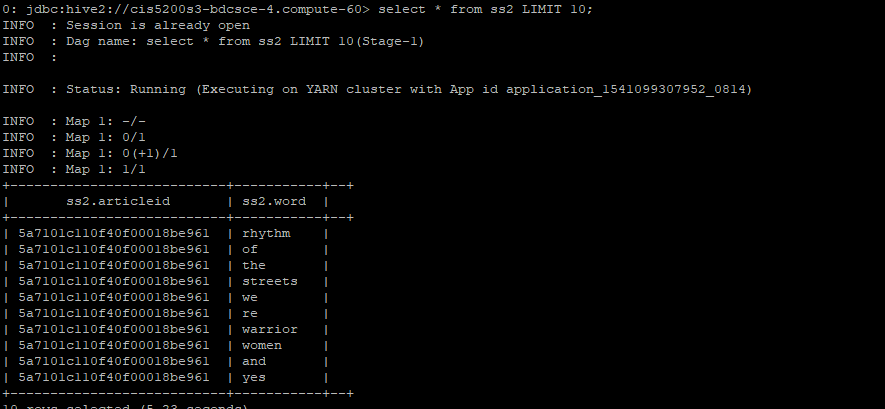
|  |
| --- |
| create view IF NOT EXISTS ss2 as  select articleid,word  from wordcloud  lateral view explode(words) dummy as word; |

**OUTPUT**



|  |
| --- |
| Select \* from ss2 LIMIT 10; |

**OUTPUT**



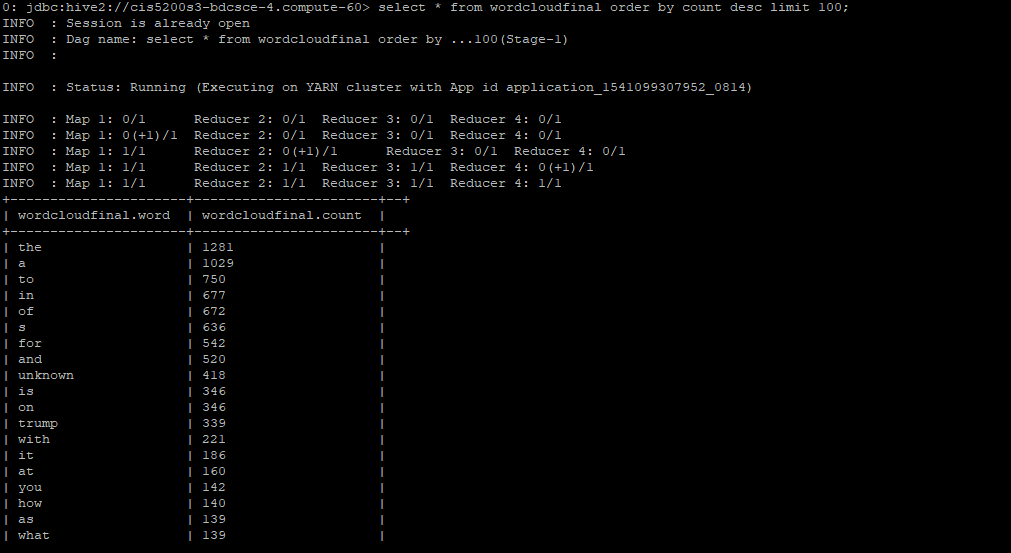
|  |
| --- |
| create view if not exists wordcloudfinal as  SELECT word, COUNT(word) AS COUNT FROM ss2 GROUP BY word ORDER BY COUNT asc; |

**OUTPUT**



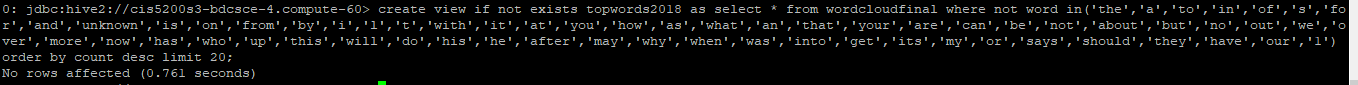
|  |
| --- |
| select \* from wordcloudfinal order by count desc limit 100; |

**OUTPUT**



|  |
| --- |
| create view if not exists topwords2018 as select \* from wordcloudfinal where not word in('the','a','to','in','of','s','for','and','unknown','is','on','from','by','i','l','t','with','it','at','you','how','as','what','an','that','your','are','can','be','not','about','but','no','out','we','over','more','now','has','who','up','this','will','do','his','he','after','may','why','when','was','into','get','its','my','or','says','should','they','have','our','1') order by count desc limit 20; |

**OUTPUT**



|  |
| --- |
| Select \* from topwords2018 LIMIT 100; |

**OUTPUT**

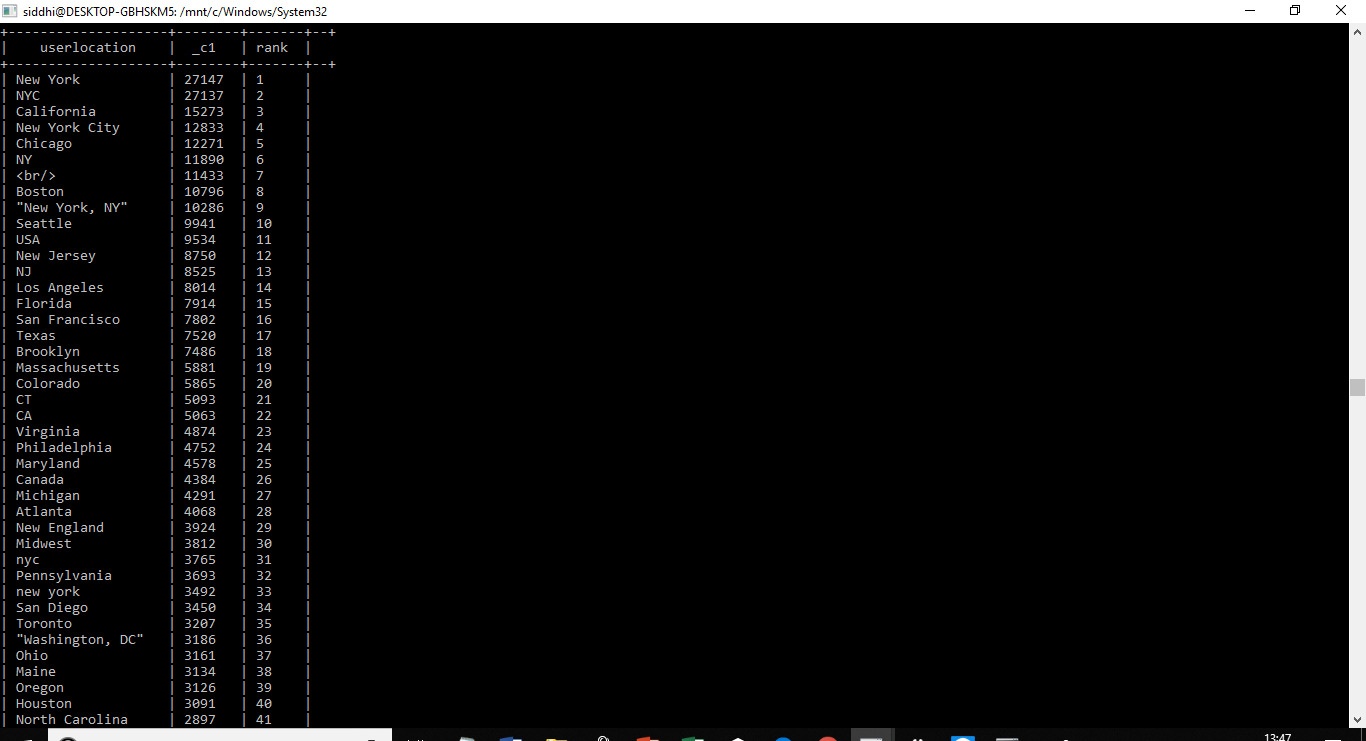


**Query 10: What are the recommendations by the user’s location for the year 2017?**

Recommendations are received from people and various users of NYT. These people and users could be present at different locations and so we have tried to get a count of recommendations that we receive from different locations across the U.S for both the years.

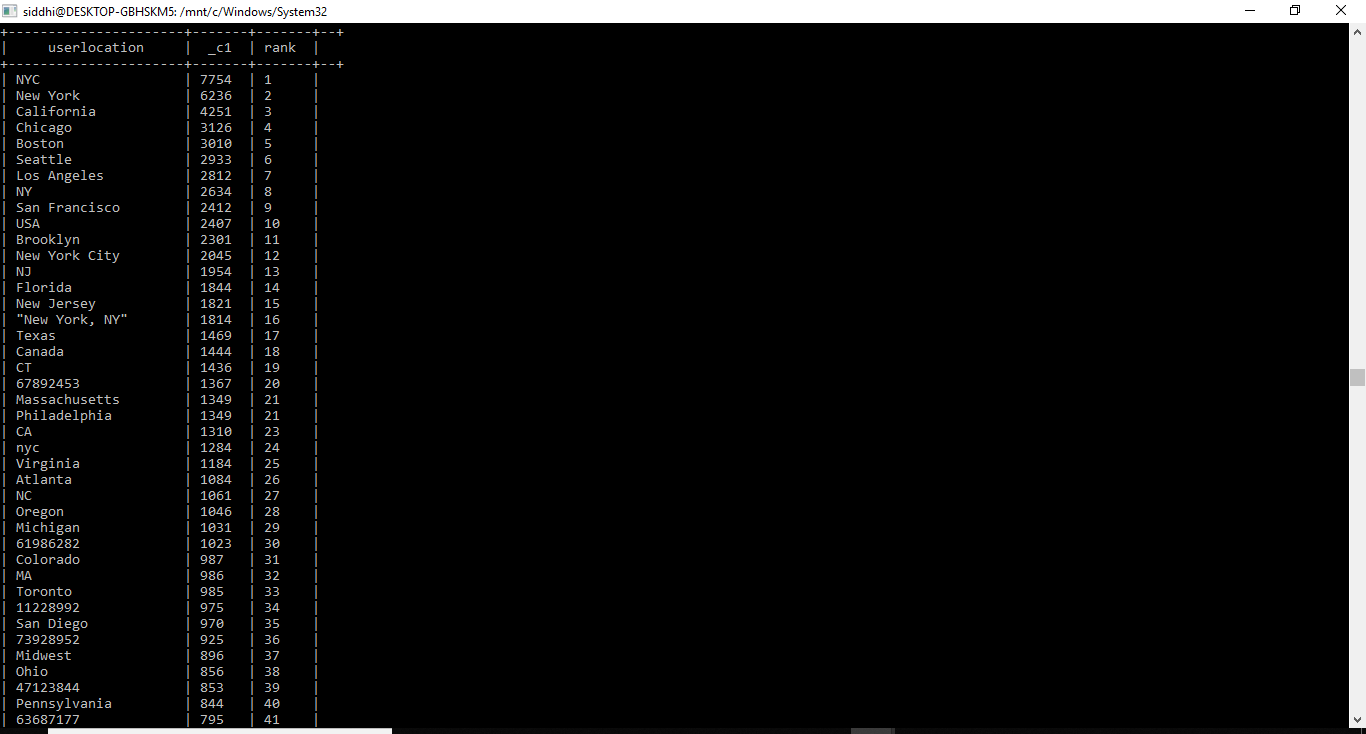
**For year 2017:**

|  |
| --- |
| select userLocation, count(recommendations), rank() over (order by count(recommendations)desc) AS  rank from commentyear2017  group by userLocation limit 100; |



**Query 11: What are the recommendations by the user’s location for the year 2018?**

|  |
| --- |
| select userLocation, count(recommendations), rank() over (order by count(recommendations)desc) AS rank from commentyear2018 group by userLocation limit 100; |



**Query 12: Most Popular Author(byline) with respect to recommendations of public for the year 2017 ?**

First, we find out the sum of recommendations as per each unique articleid with help of following query

|  |
| --- |
| create table if not exists tanvi\_byline2 as select sum(recommendations) as recommendations,articleid from commentyear2017 group by articleid; |

****

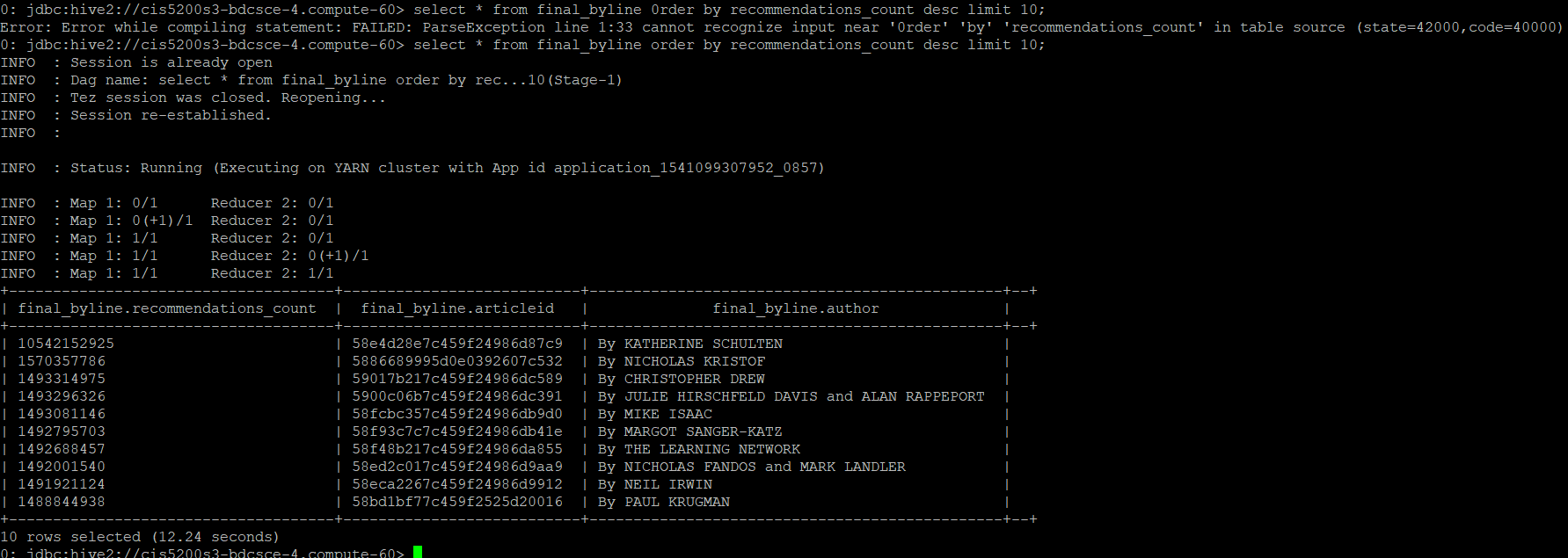
We create a new table which will store the results of the above output as well map the author with the help of inner join.

|  |
| --- |
| create table final\_byline as select tanvi\_byline2.recommendations recommendations\_count,tanvi\_byline2.articleid articleid,articleyear2017.byline author from tanvi\_byline2 inner join articleyear2017 on tanvi\_byline2.articleid = articleyear2017.articleid; |

****

Lastly, we find out the most popular author - byline with help of below query

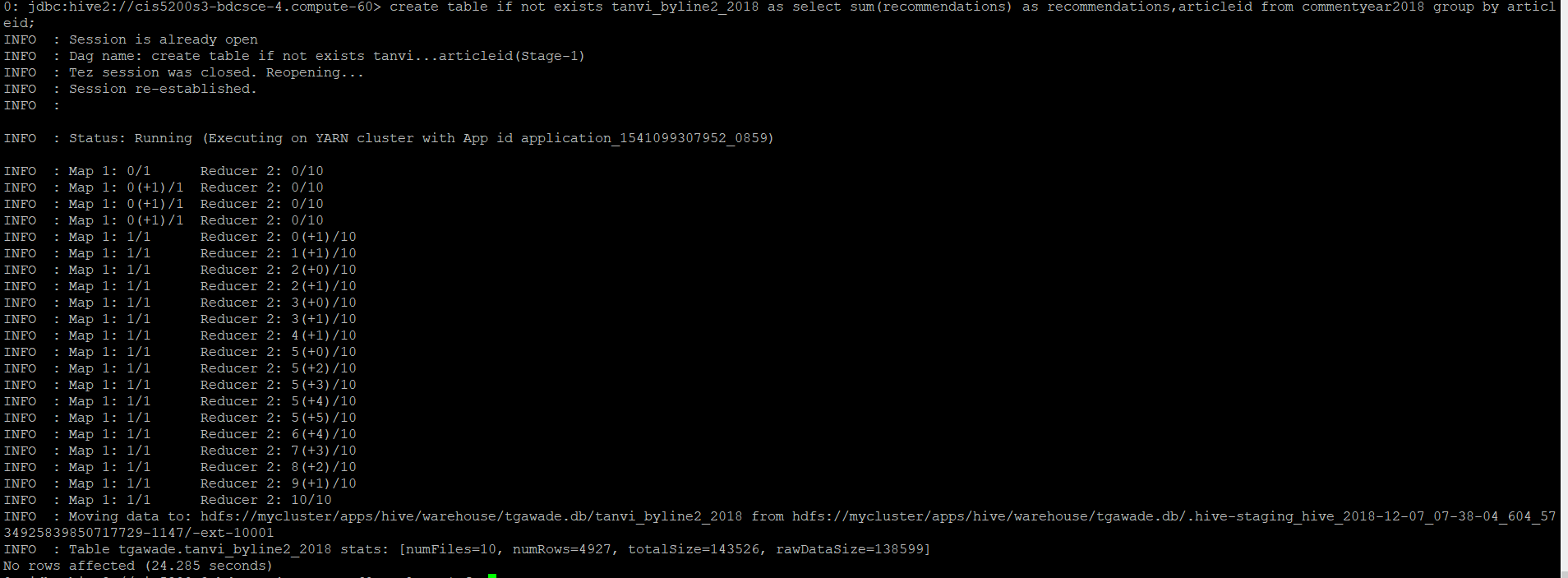
|  |
| --- |
| select \* from final\_byline order by recommendations\_count desc limit 10; |

****

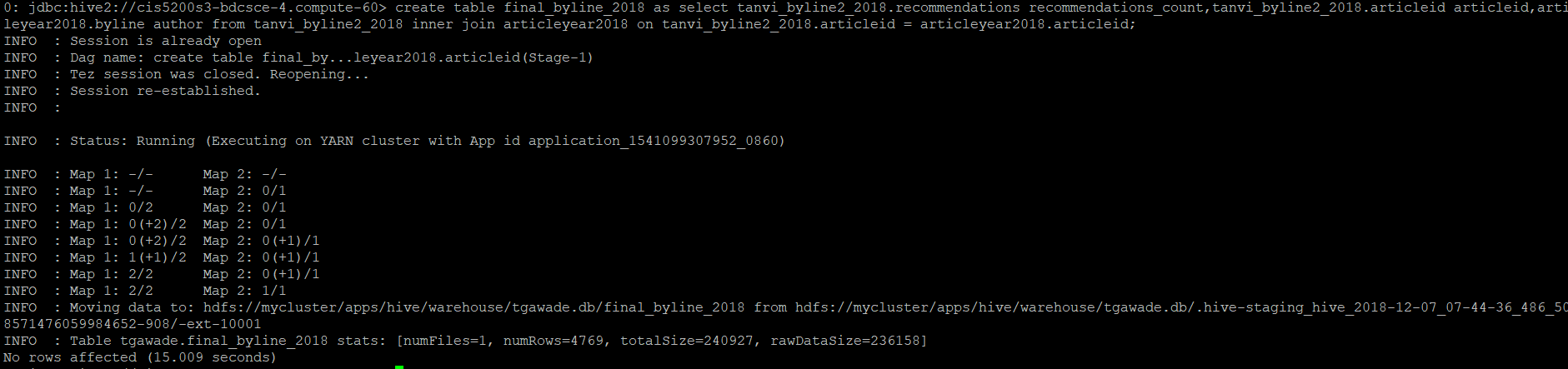
**Query 13: Most Popular Author(byline) with respect to recommendations of public for the year 2018?**

Similarly follow the above commands to find out the most popular author - byline with respect to the recommendations of public in the year 2018.

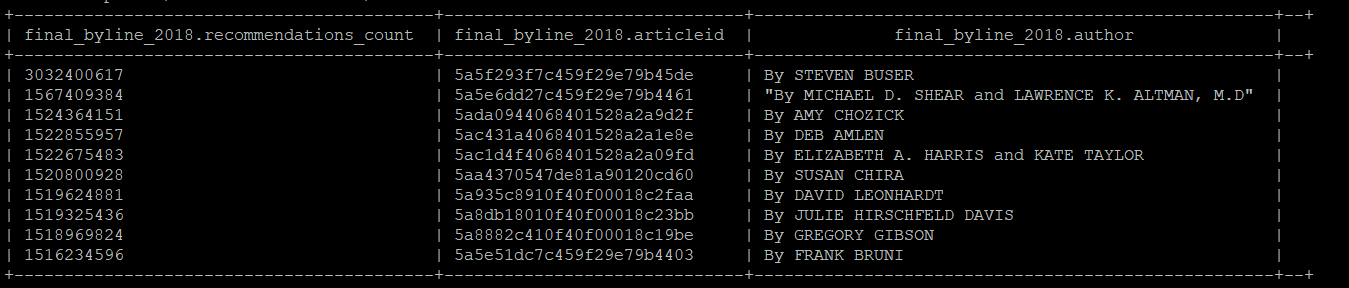
|  |
| --- |
| create table if not exists tanvi\_byline2\_2018 as select sum(recommendations) as recommendations,articleid from commentyear2018 group by articleid; |



|  |
| --- |
| create table final\_byline\_2018 as select tanvi\_byline2\_2018.recommendations recommendations\_count,tanvi\_byline2\_2018.articleid articleid,articleyear2018.byline author from tanvi\_byline2\_2018 inner join articleyear2018 on tanvi\_byline2\_2018.articleid = articleyear2018.articleid; |



|  |
| --- |
| select \* from final\_byline order by recommendations\_count desc limit 10; |



Hence the results for the most popular author - byline with respect to recommendations of public as given above for the year 2018.

**DOWNLOADING DATA (OUTPUT FILES) INTO YOUR PC**

After the Hive tables are created, we can download it to our personal PC/laptop as follows:

(The following is an example to download the output file for one query, similarly all the output files for all the queries have been downloaded in the same manner)

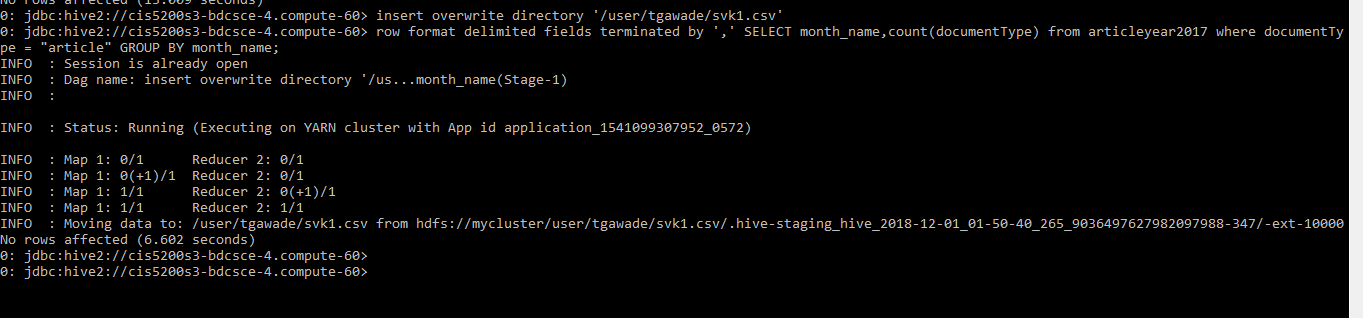
Step 1: Open another terminal Bash and connect it to Beeline which is connected to the Oracle cloud in order to download the output files and type in the following command at beeline:

|  |
| --- |
| insert overwrite directory '/user/tgawade/svk1.csv'  row format delimited fields terminated by ',' SELECT month\_name,count(documentType) from articleyear2017 where documentType = "article" GROUP BY month\_name; |

For the field marked in Green: Note: svk1.csv here is just a sample file name. you can name it anything and accordingly file with that name will be created)

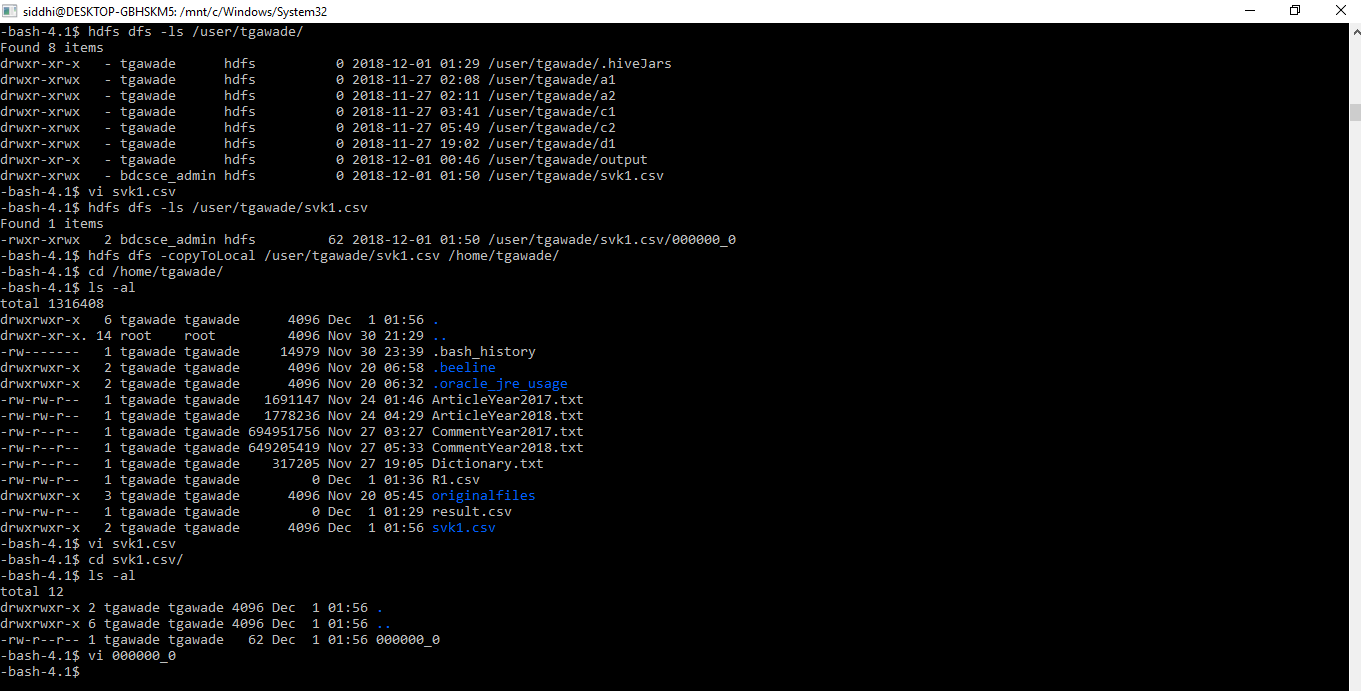
For the field marked in Red: Here, which ever query you wish to run, copy and paste it here, in the field marked red above)

The following will be displayed an output on your screen:



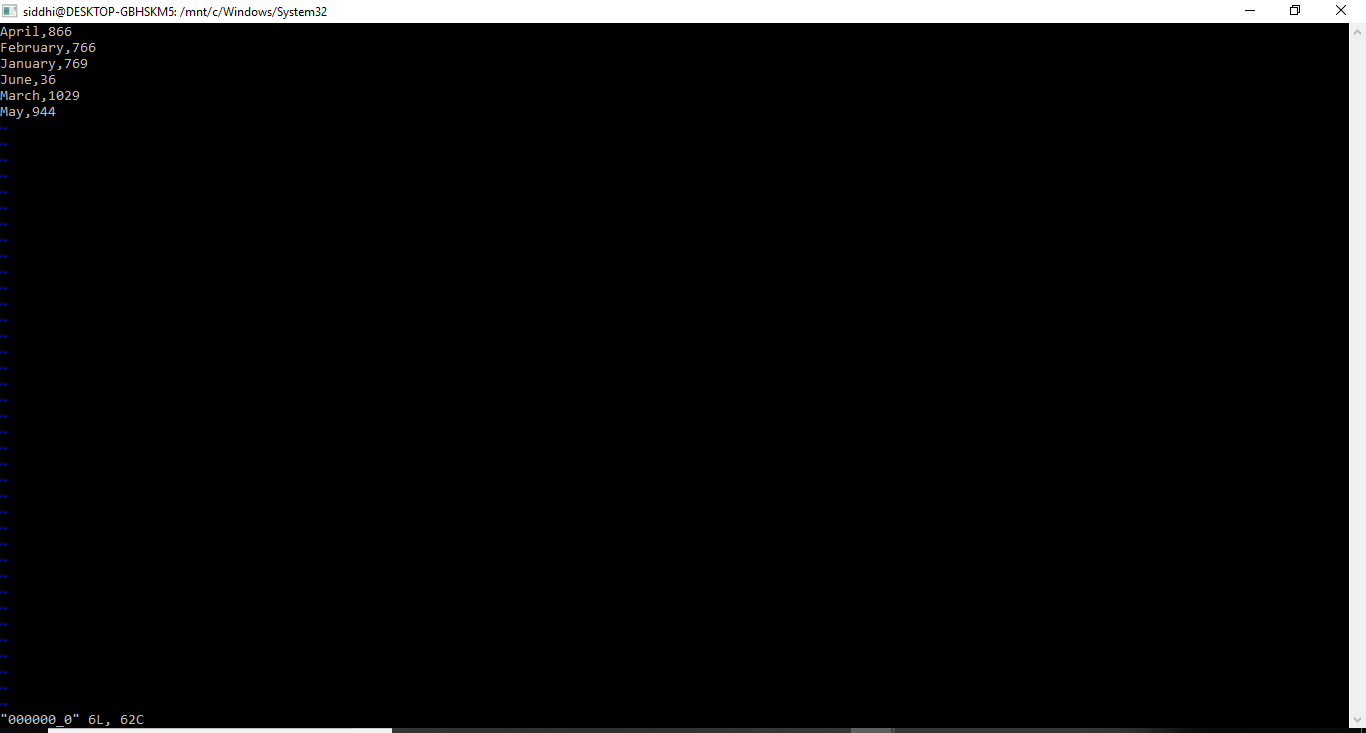
Follow the rest steps as given below:

|  |
| --- |
| Step 2 : -bash-4.1$ hdfs dfs -ls /user/tgawade/svk1.csv  : -bash-4.1$ hdfs dfs -copyToLocal /user/tgawade/svk1.csv /home/tgawade/  : -bash-4.1$ cd /home/tgawade/  : -bash-4.1$ ls -al  : -bash-4.1$ cd svk1.csv/  : -bash-4.1$ vi 000000\_0 |



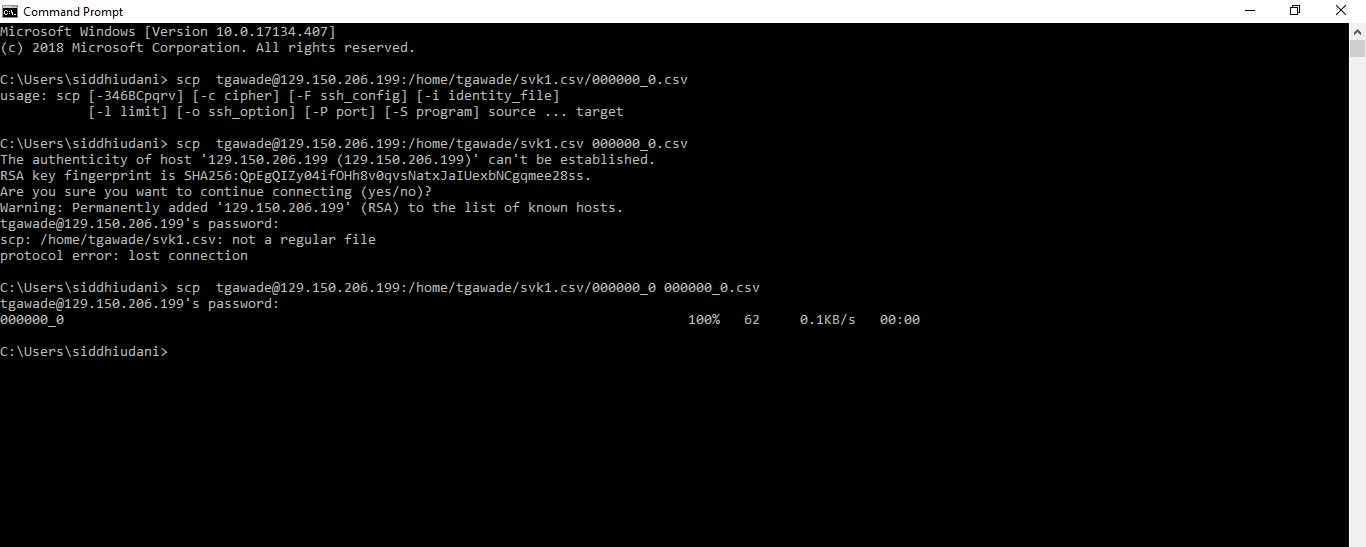
When you run the last command of the screenshot, that is, “ -bash-4.1$ vi 000000\_0” , the output should be the result of your query :

Example: For the query that I have run above, the output is as follows:



Step 3: Now open Command Prompt/Putty and run the following steps.

(We have done this using Command Prompt)



Step 4: After this go to your local machine and you will find the output file there.

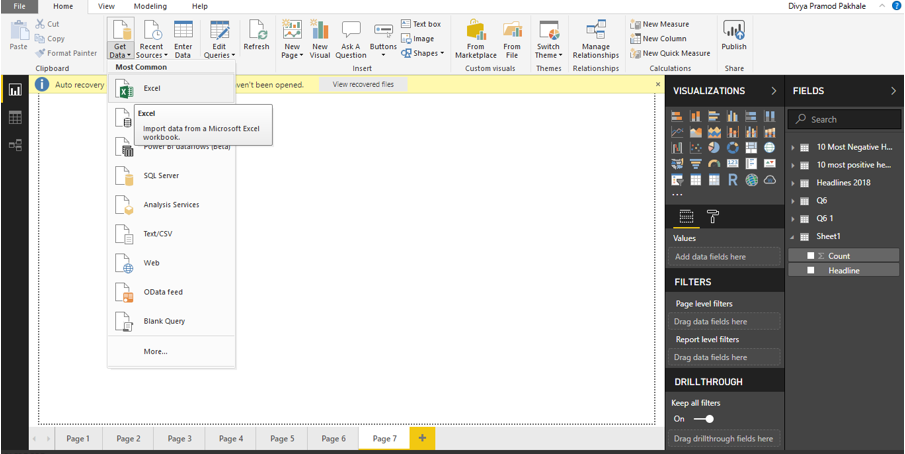
For Example, in my case it was: c drive -> users -> siddhiudani

**VISUALIZATION OF DATA**

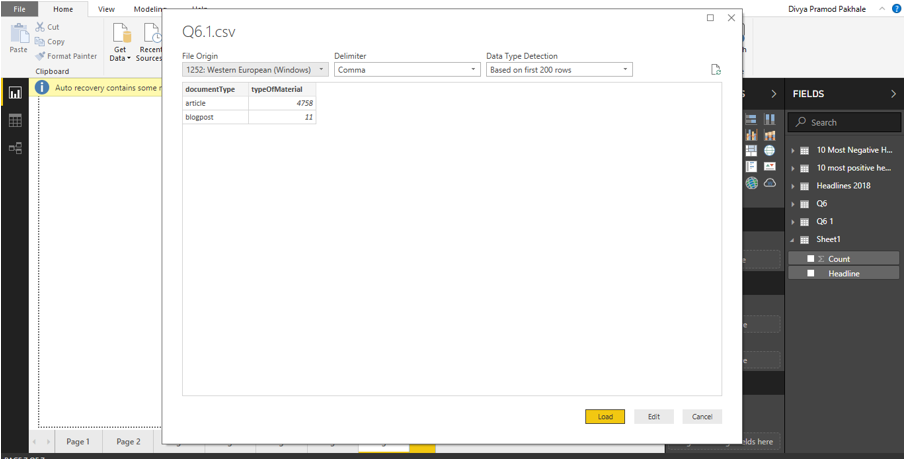
**Visualizing Data: (In order to visualize the data, we have used tableau, power BI as well as Excel 3D Maps)**

**Query 1: Count of document type by type of material for the year 2017 and 2018.**

Open the power bi tool. Click on Get data -> Excel -> Select your file (Q6.1)

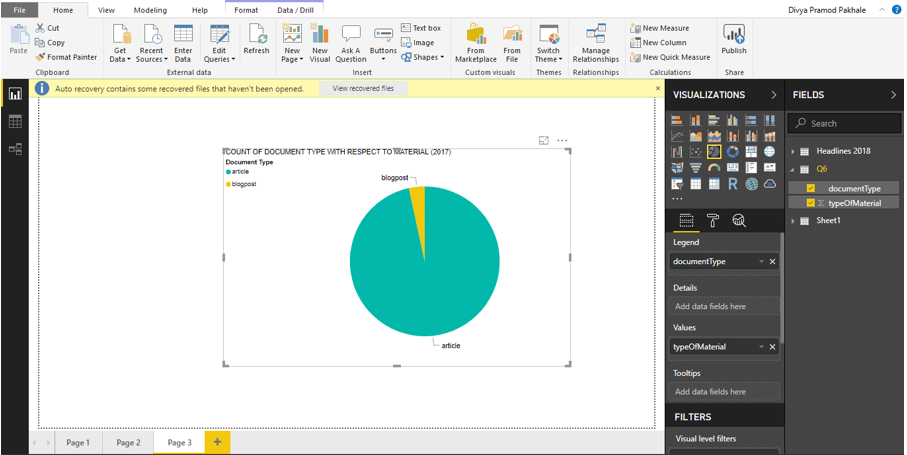


Click on Load. Once the data is loaded, go to one sheet



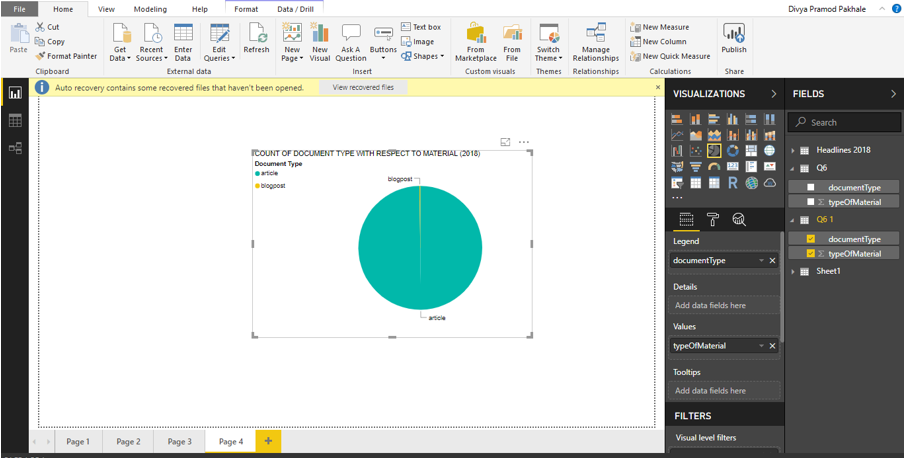
**For the Year 2017**

Drag document type and type of material as shown in the picture below and select pie chart visualization from the visualization field

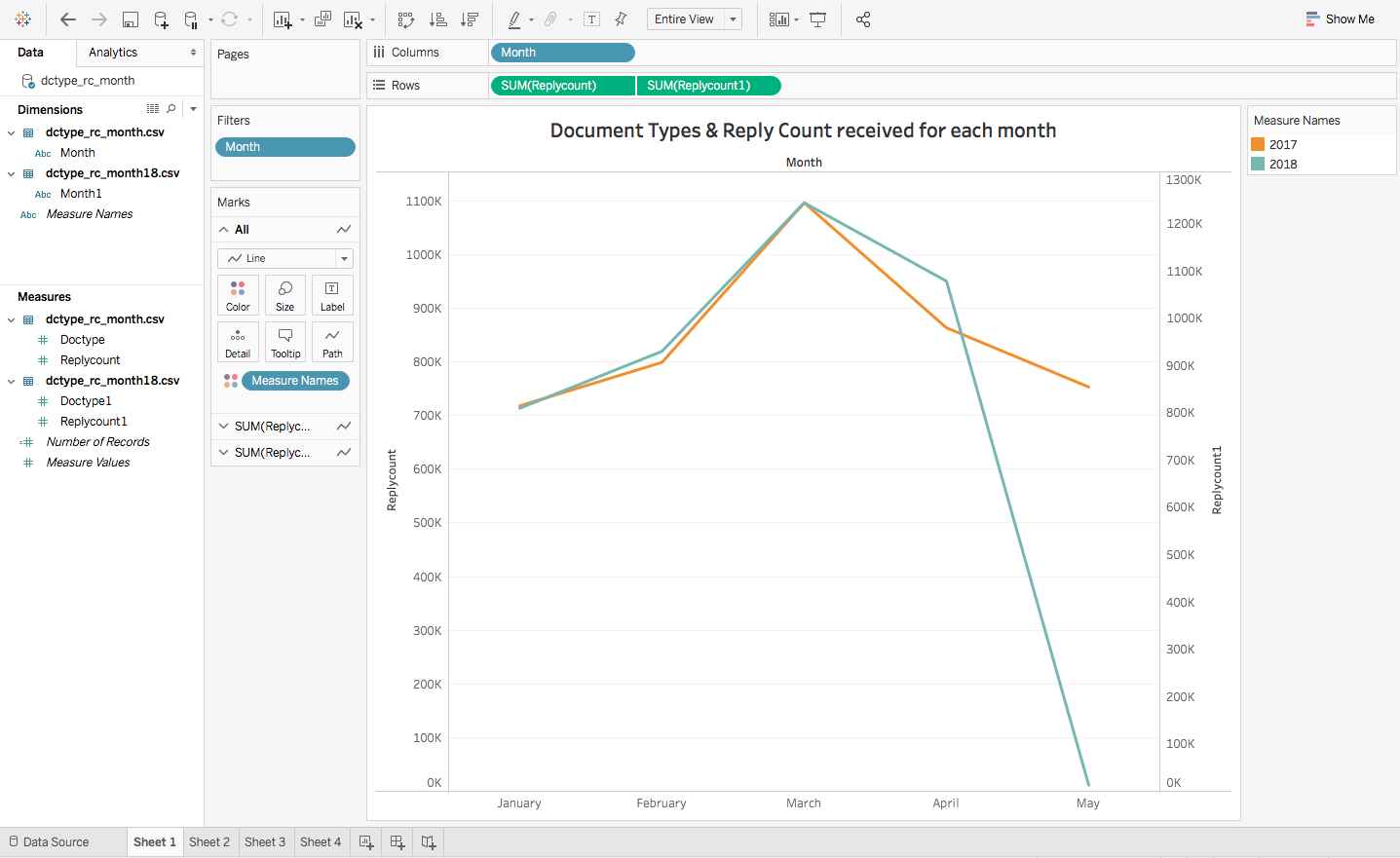


**For the year 2018**

Drag document type and type of material as shown in the picture below and select pie chart visualization from the visualization field



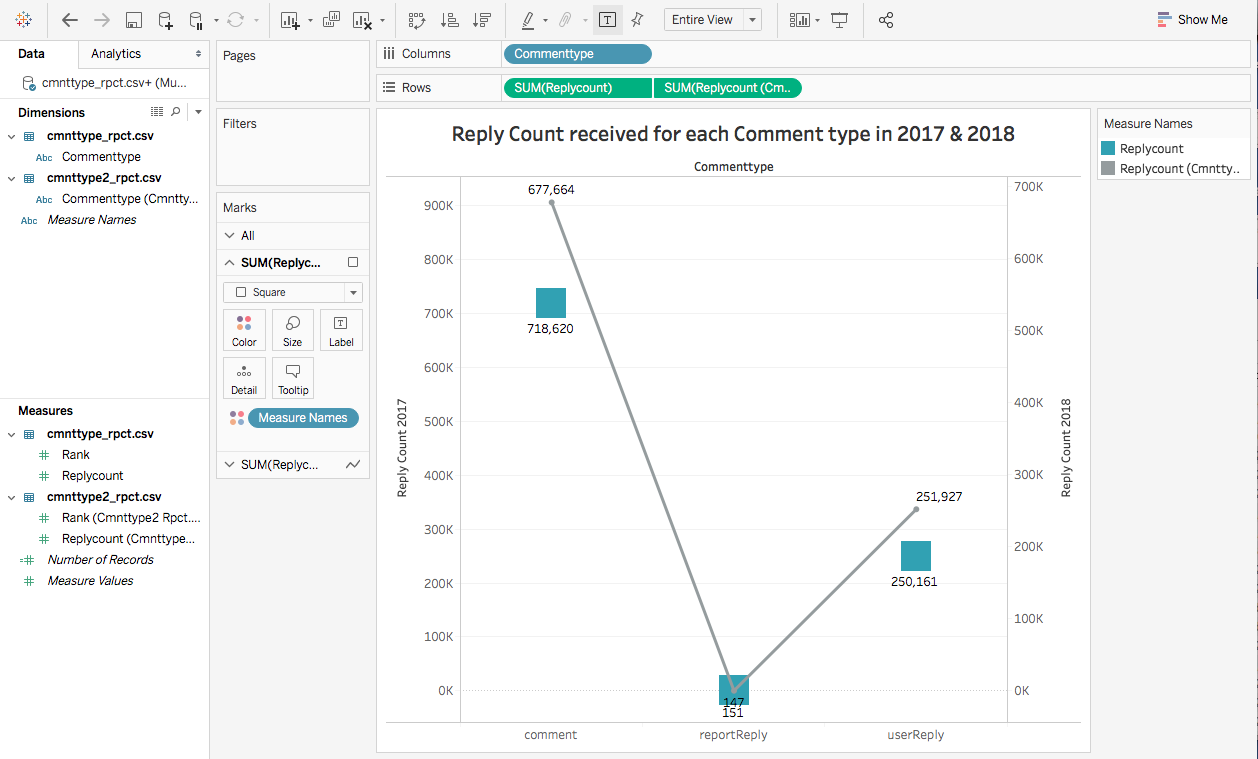
**Query 2: What is the reply count for the document type month wise for Year 2017 & 2018?**

****

Above visual shows Line Chart. Open Tableau Tool -> Upload the output file -> Drag month from dimensions to rows and reply count from dimensions to rows.

X Axis represents Month from January to May. Y Axis represents reply cunt for 2017 (Left) and 2018 (Right). Orange color represents year 2017 and blue color shows trend in year 2018

**Query 3: What is the reply count for each comment type.**

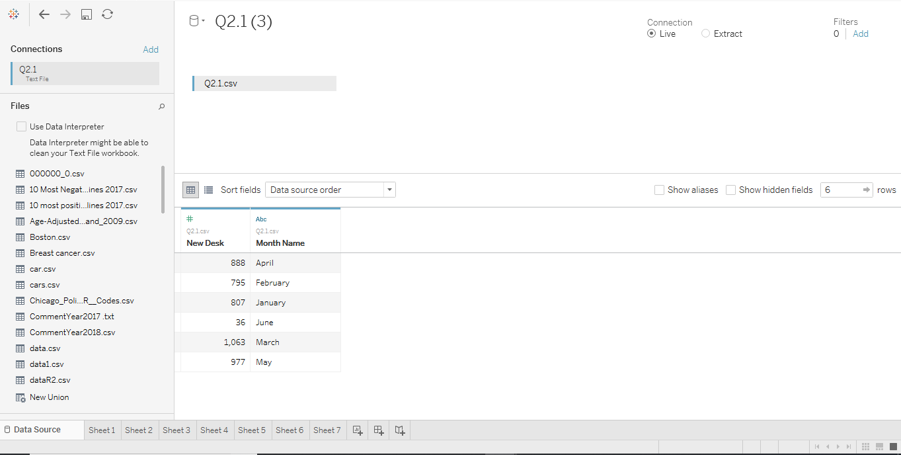
****

Above visual shows comparative analysis using dual axis. Open Tableau Tool -> Upload the output file -> Drag comment type from dimensions to rows and reply count for 2017 and 2018 from dimensions to rows.

X Axis represents comment type. Y Axis represents reply count for 2017 (Left) and 2018 (Right). Blue color represents year 2017 and grey color shows trend in year 2018.

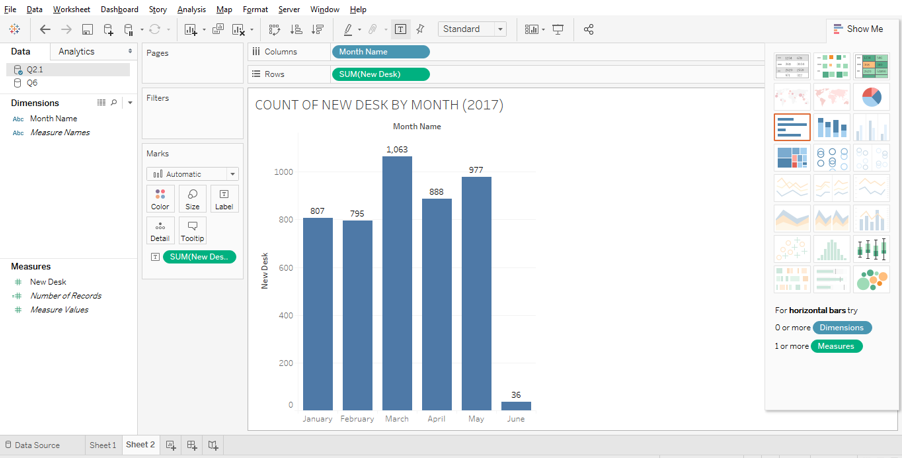
**Query 4: What is the count of new desk month wise.**

Open the output file from “q2.1” folder in Tableau. After loading the data in Tableau, we will click of sheet 1 as shown in the picture below to create our visualization.

****

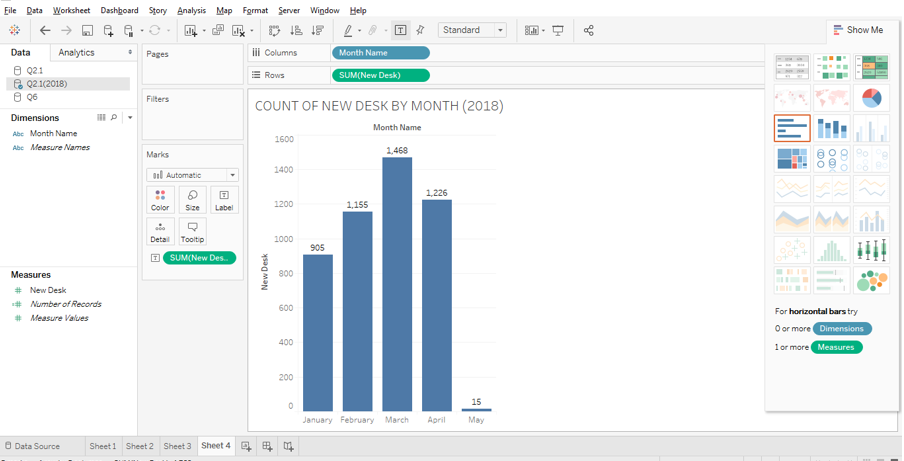
After clicking on sheet 1 worksheet, we drag the month name from dimensions to the rows field and number of desks filed from measures to the columns field.

For the year 2017



After clicking on sheet 2 worksheet, we drag the month name from dimensions to the rows field and number of desks filed from measures to the columns field.

**For the year 2018**



**Query 5: What is the count of new desk based on recommendations.**

Open the output file from “q5.1(2017)” folder in Tableau. After loading the data in Tableau, as shown in the picture below to create our visualization.

Drag New Desk from the dimensions field into rows and drag recommendations from measures into text in the marks field and then clicked on packed bubbles from the show me tab to get the visualization.

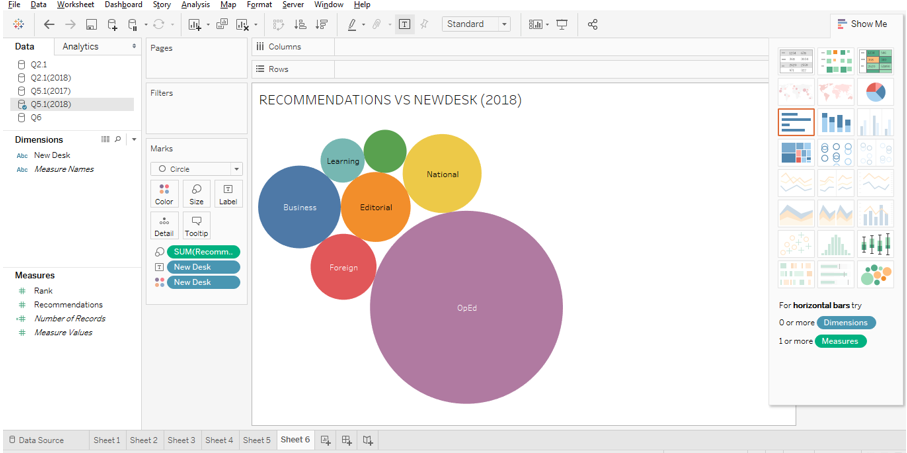
**For the year 2017**

****

**For the year 2018**

Open the output file from “q5.1(2018)” folder in Tableau. After loading the data in Tableau, as shown in the picture below to create our visualization.

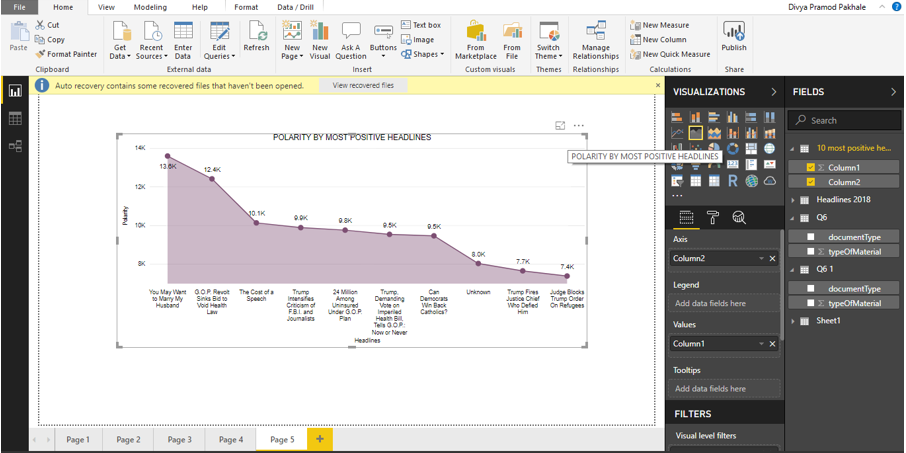
Drag New Desk from the dimensions field into rows and drag recommendations from measures into text in the marks field and then clicked on packed bubbles from the show me tab to get the visualization.

****

**Query 6: What is the degree of polarity by most positive headlines.**

Open the power bi tool. Click on Get data -> Excel -> Select your file (Ten most positive headlines)

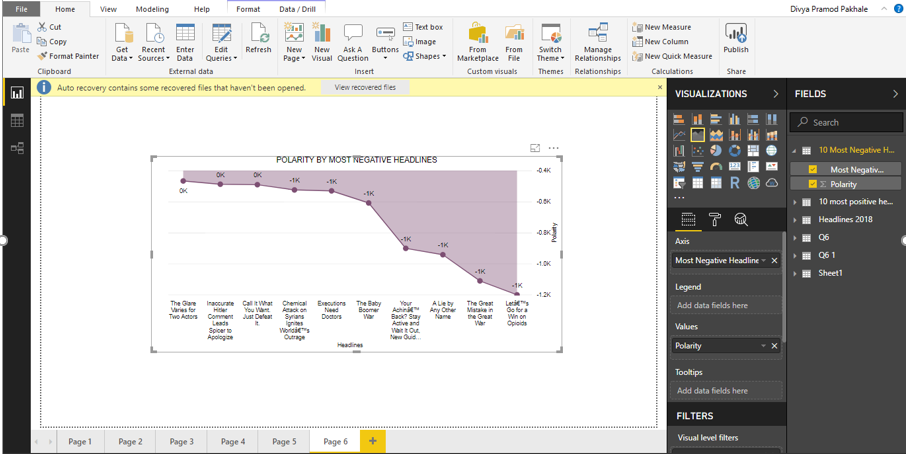
Drag column 1- Degree of Polarity and column 2 - 10 Most Positive Headlines based on the public comments as shown in the picture below and select area chart visualization from the visualization field.

****

**Query 7: What is the degree of polarity by most negative headlines.**

Open the power bi tool. Click on Get data -> Excel -> Select your file (Ten most negative headlines)

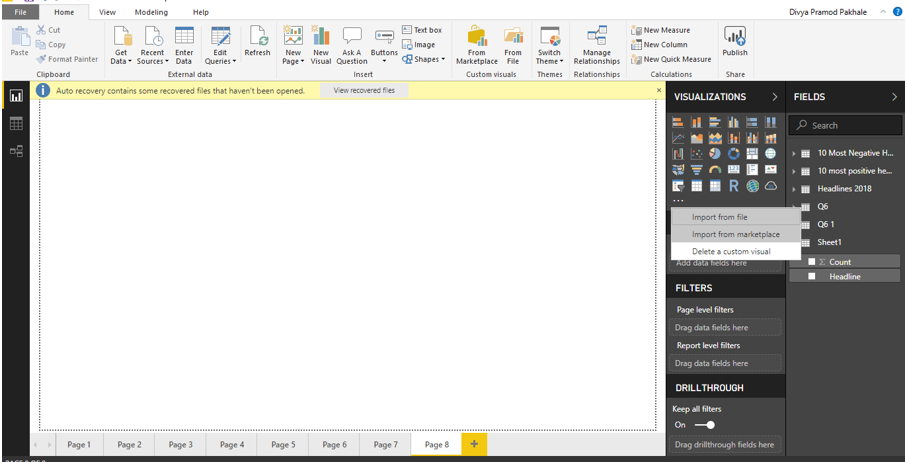
Drag most negative headlines and degree of polarity columns based on the public comments as shown in the picture below and select area chart visualization from the visualization field.

****

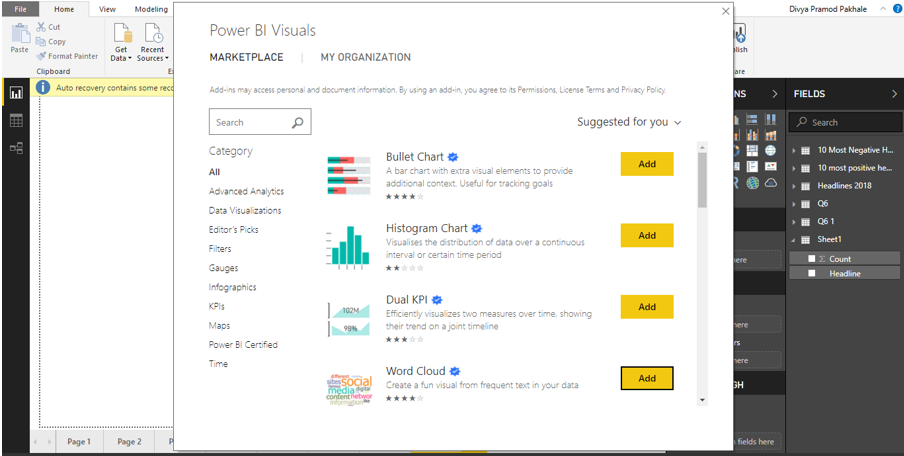
**Query 8: What is the most common words in the headlines for article year 2017.**

Open the power bi tool. Click on Get data -> Excel -> Select your file (Sheet1)

Click on … from the visualization field and select Import from marketplace.



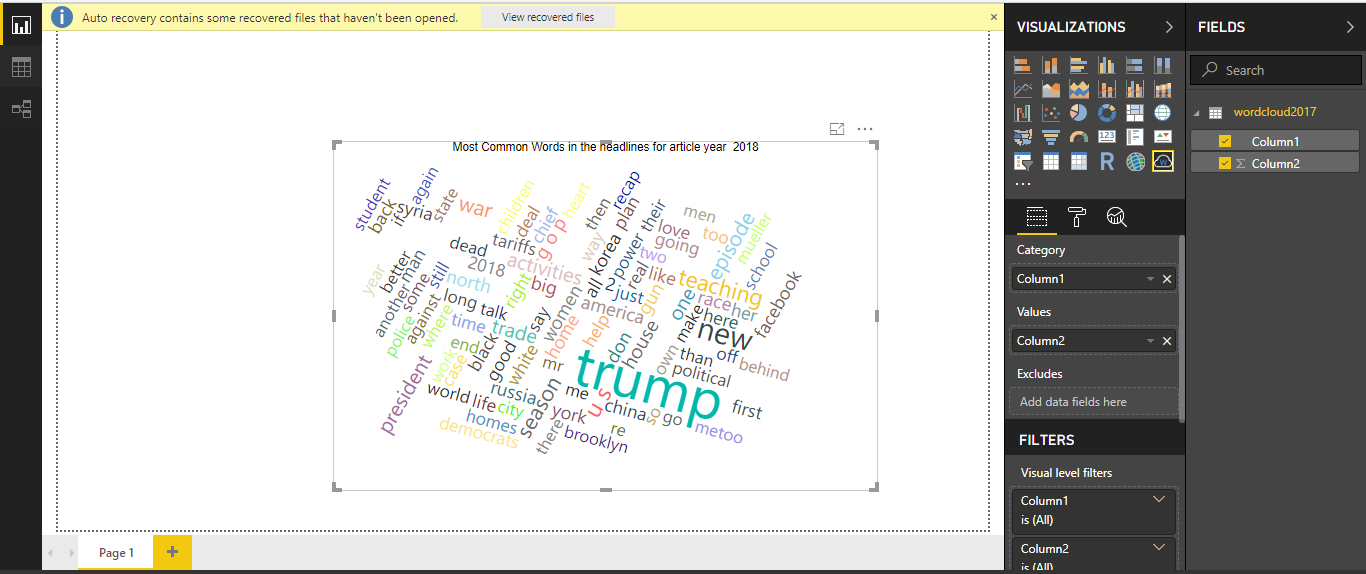
Click on ADD button for word cloud



Drag Count and headlines column and make the changes accordingly

****

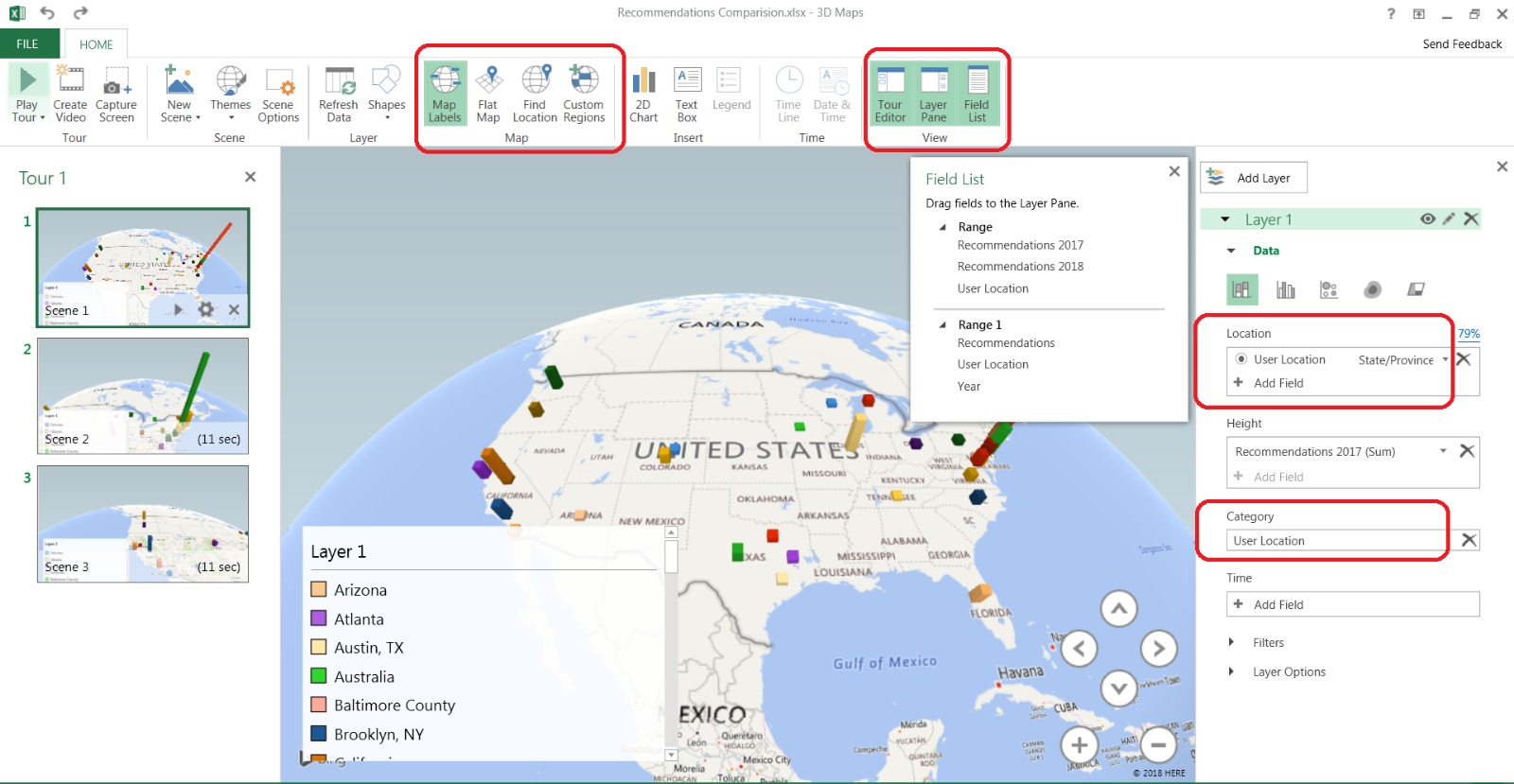
**Query 9: What are the most common words in the headlines for article year 2018.**

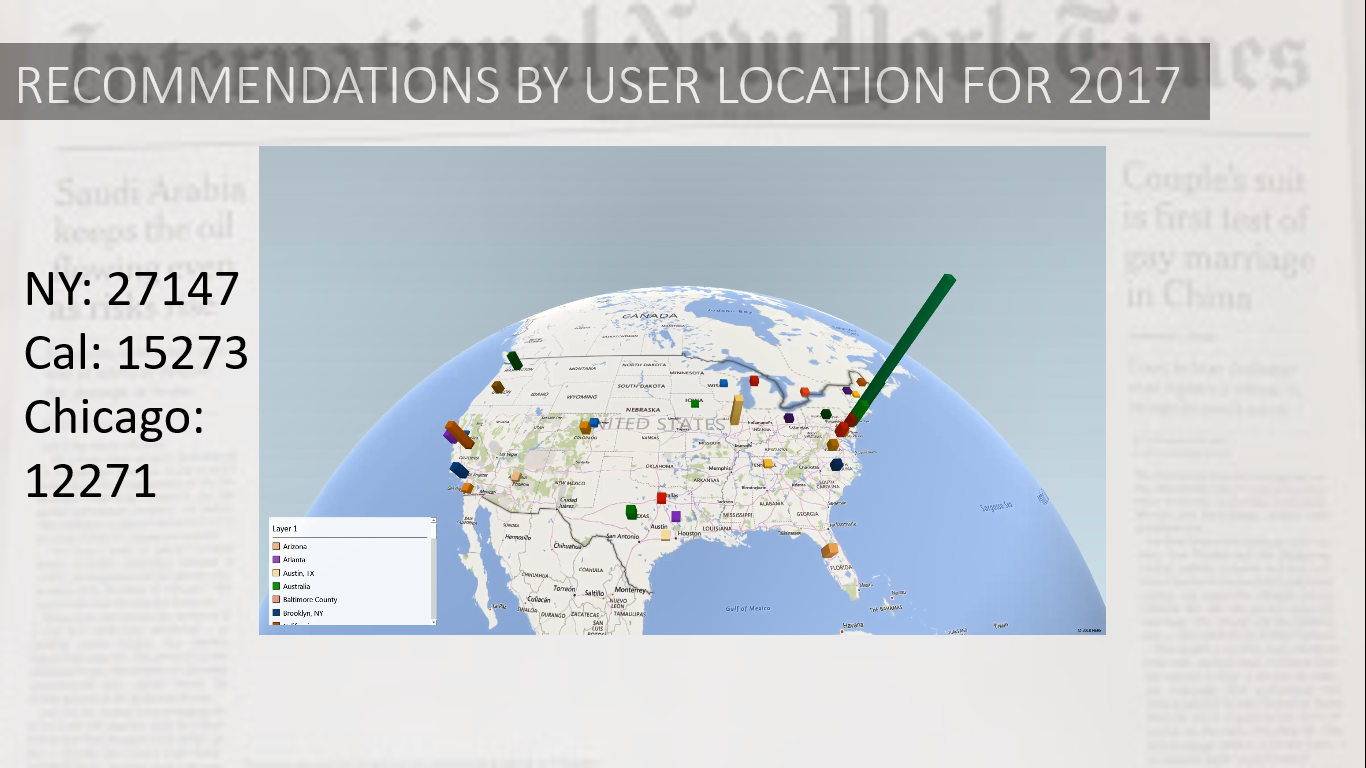


**Query 10: What are the recommendations by the user’s location for the year 2017.**

To create the 3D-Map, follow these steps:

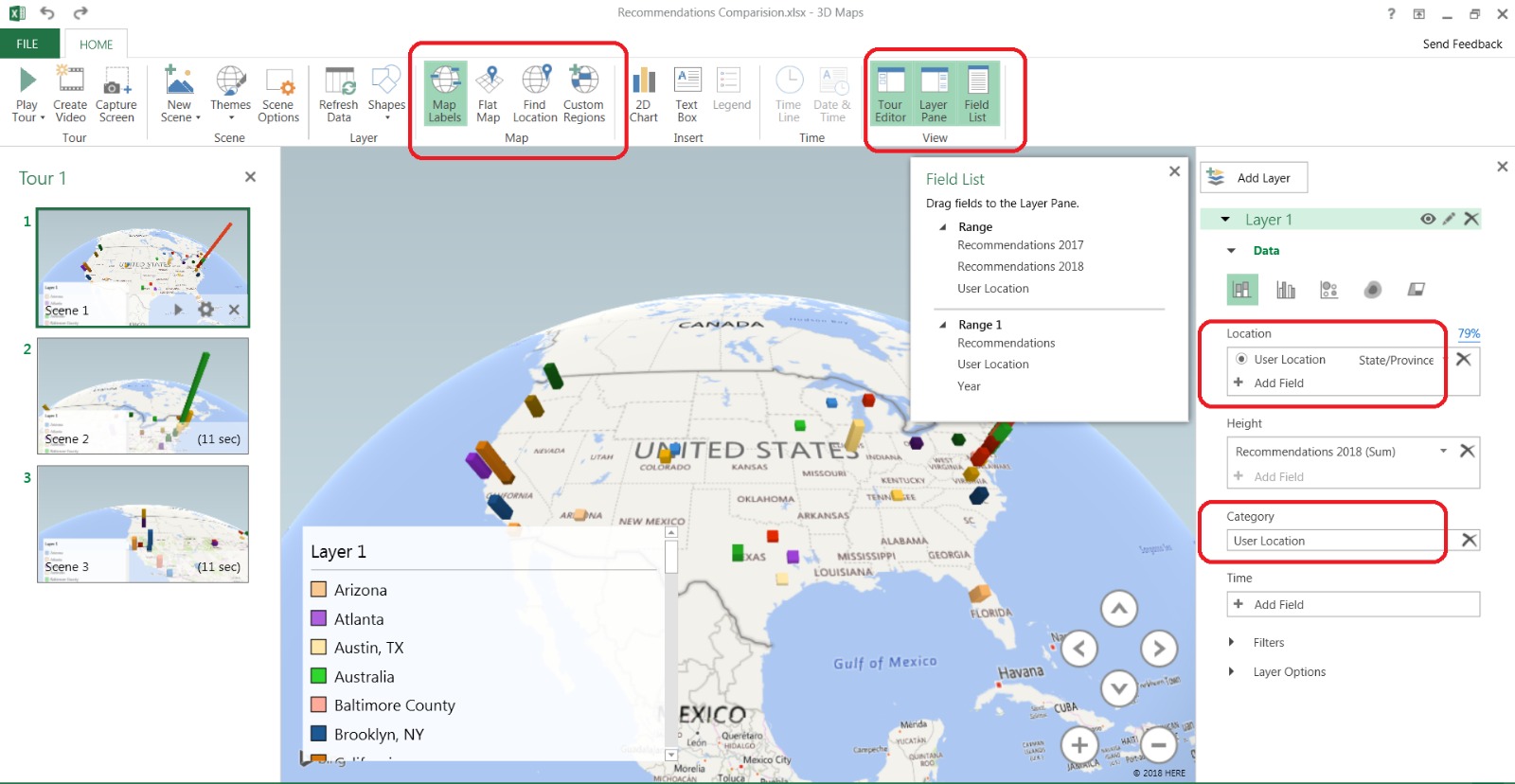
1. Load the output file for the respective query (which had been extracted from Hive) into Microsoft Excel.
2. Select the data, including the column headers in the table format.
3. Click Insert | 3D Maps | Open 3D Maps.
4. Drag fields (column header names) to the Layer panel as shown in the screenshot.
5. Reveal the 3D-Map.

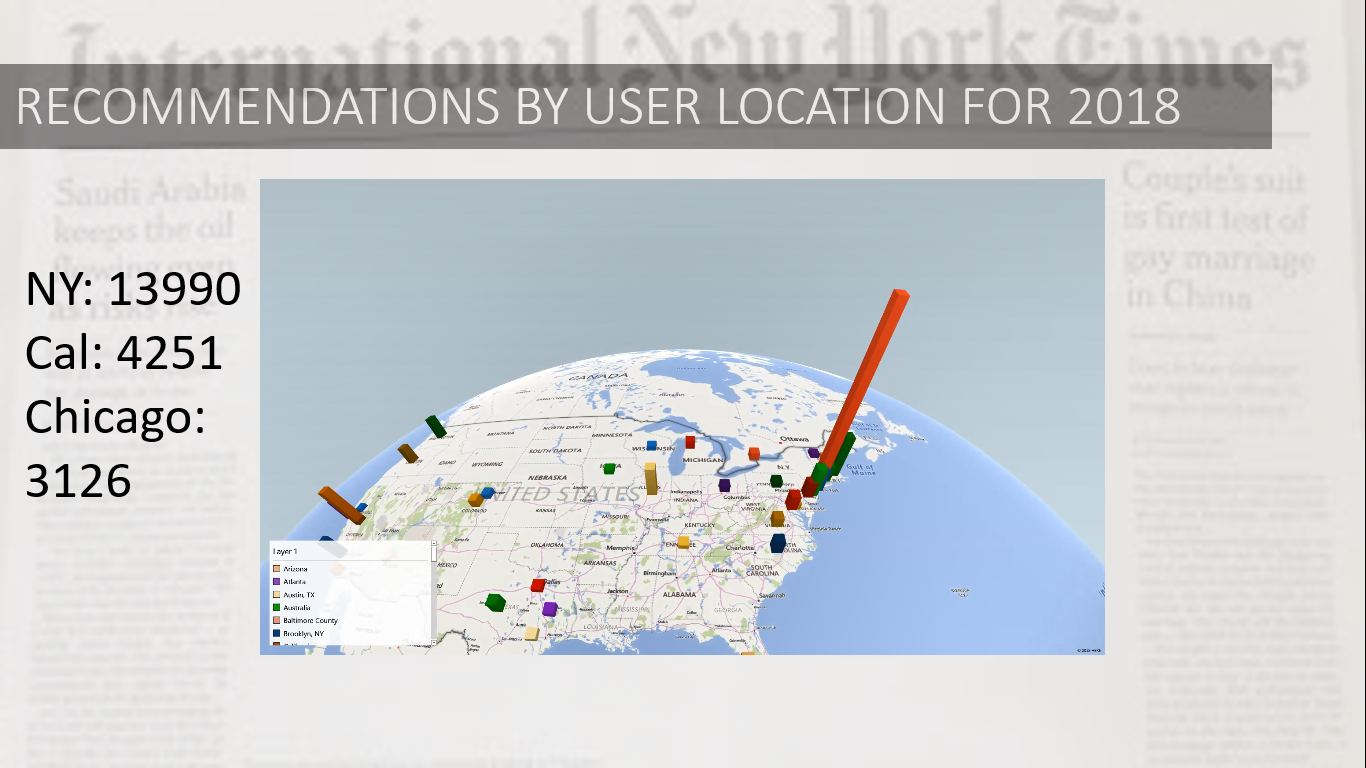




**Query 11: What are the recommendations by the user’s location for the year 2018.**

(Follow the same steps as mentioned above)

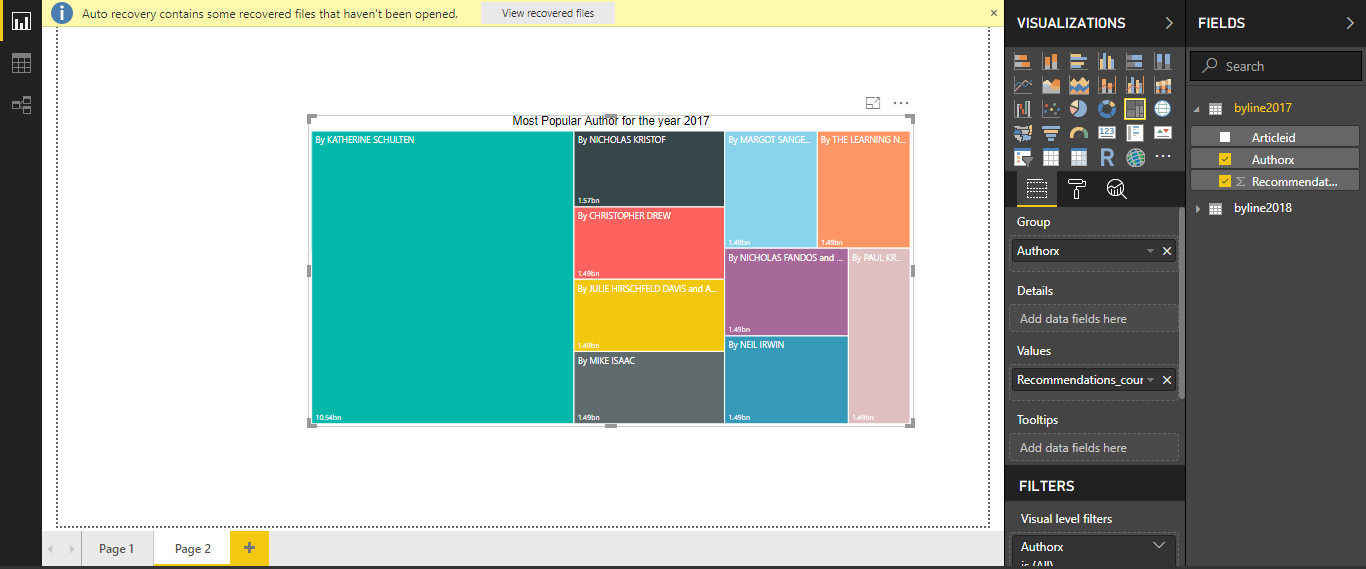




**Query 12: Most Popular Author(byline) with respect to recommendations of public for the year 2017?**

Open the power bi tool. Click on Get data -> Excel -> Select your file (byline2017.csv)

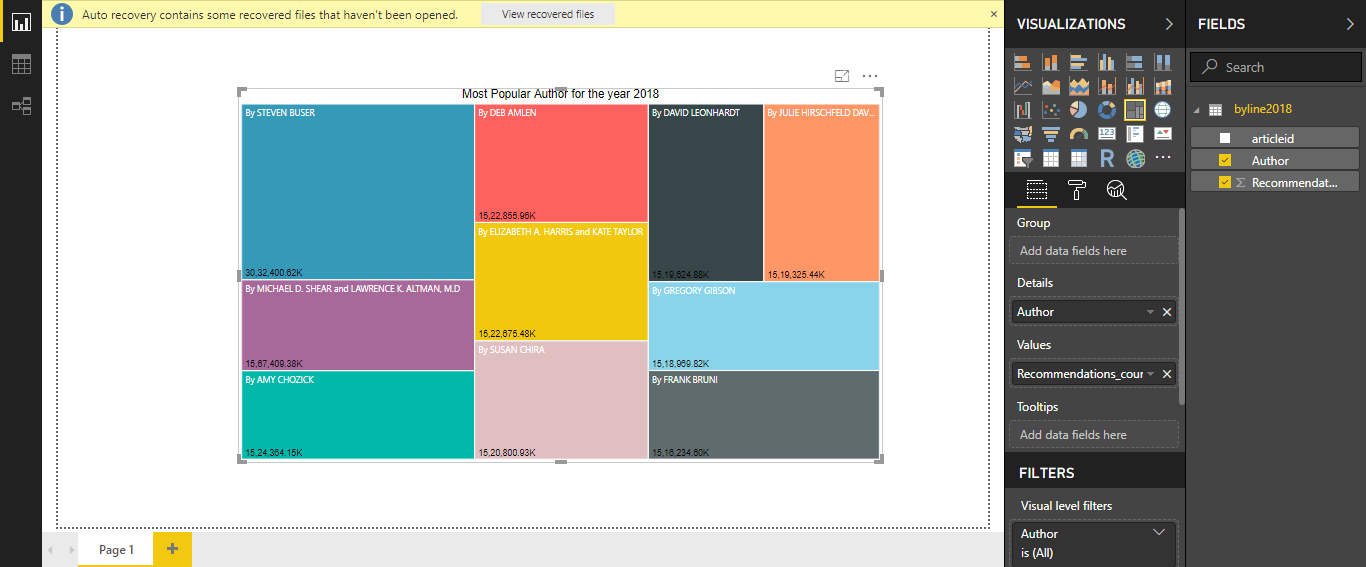
Drag Author and Recommendations column as shown in the picture below and select tree map chart visualization from the visualization field.

****

**Query 13: Most Popular Author(byline) with respect to recommendations of public for the year 2018?**

Open the power bi tool. Click on Get data -> Excel -> Select your file (byline2018.csv)

Drag Author and Recommendations column as shown in the picture below and select tree map chart visualization from the visualization field.

****

**REFERENCES AND GITHUB LINK**

**1.** <https://github.com/tanvigawade/Project5200_Group3>

**2.** [https://towardsdatascience.com/predicting-popularity-of-the-new-york-times-comments-part-1- d32f26261f6f](https://towardsdatascience.com/predicting-popularity-of-the-new-york-times-comments-part-1-%20%20d32f26261f6f)

**3.** <https://www.kaggle.com/aashita/word-clouds-of-various-shapes>

**4.** <https://www.kaggle.com/aashita/exploratory-data-analysis-of-comments-on-nyt/notebook>

**This is the end of the lab**