Amazon Data Analysis using HIVE and Power BI

GROUP 2, SECTION 2 CIS 5200

SAHAVAT TANGCHITNOB (STANGCH2@CALSTATELA.EDU)

MARIA ESPINOZA (MESPINO6@CALSTATELA.EDU)

JULIO ROJAS (JROJAS2@CALSTATELA.EDU)

SWARNIM JAMBHULE (SJAMBHU@CALSTATELA.EDU)

Objectives

- Download, unzip and upload data to HDFS
- Create Hive tables to query Amazon data
- Create Hive queries to analyze the data
- Create Hive queries to analyze sentiment of data using dictionary
- Download Data into your PC
- Analyze Data using Power BI

Data Size and Source

- Data source URL:

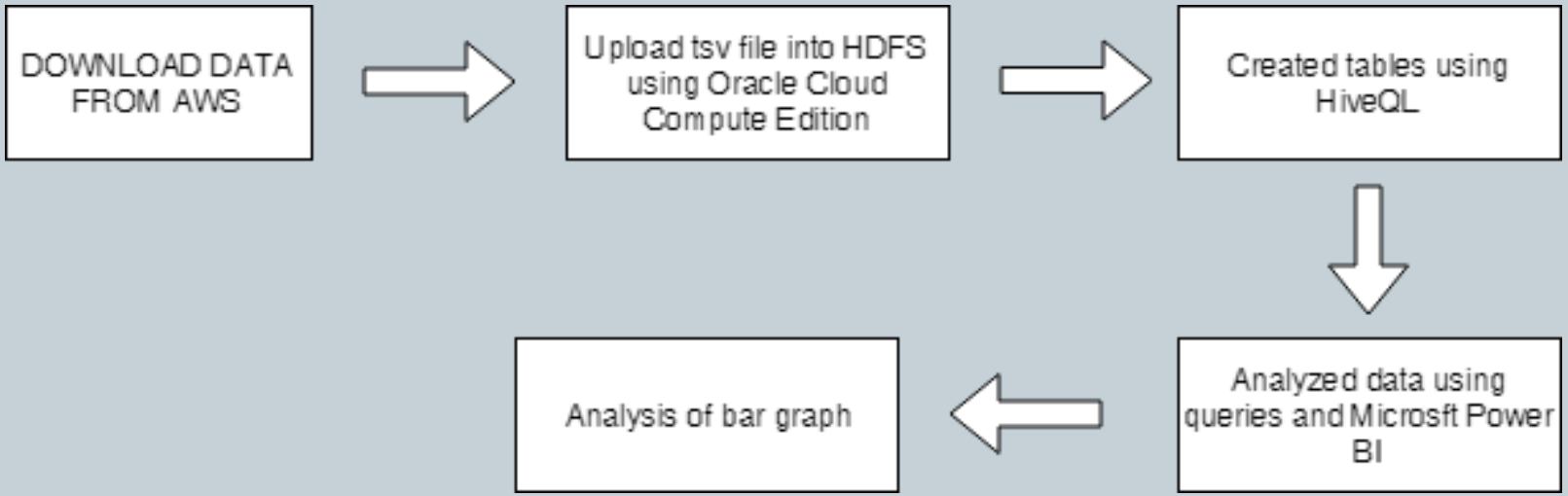
https://s3.amazonaws.com/amazon-reviews-pds/tsv/amazon_reviews_us_Books_v1_02.tsv.gz

- Data Size (unzipped): 3.24 GB

Platform Specification

- Platform Spec
- Oracle Big Data Compute Edition: 5 nodes
- CPU Speed: 2.2 GHz
- OCPUs: 10
- Memory: 150 GB
- Storage: 678 GB
- HDFS Capacity: 147 GB

Workflow


DOWNLOAD DATA FROM AWS
Upload tsv file into HDFS using Oracle Cloud Compute Edition
Created tables using HiveQL
Analyzed data using queries and Microsft Power BI
Analysis of bar graph

Step 1: Connect to Oracle Cloud

- You must have an ip address to connect to Oracle Cloud
- Your CalStateLA username should be a username/pwd for Oracle account
- 1. SSH to connect it, for example, the instructor mespino6:
  - ssh mespino6@129.150.128.177

```
Sahavats-Air:~ sahavattangchitnob$ ssh mespino6@129.150.128.177
mespino6@129.150.128.177's password:
-bash-4.1$
```

Step 1: continued

- And, you may run the following HDFS commands to test if hdfs works well at your Oracle account:
  - hdfs dfs -ls
  - hdfs dfs -mkdir test
  - hdfs dfs -ls

```
-bash-4.1$ hdfs dfs -mkdir test;
-bash-4.1$ hdfs dfs -ls;
Found 7 items
drwxr-xrwx   - mespino6 hdfs          0 2018-10-03 01:46 .hiveJars
drwx------   - mespino6 hdfs          0 2018-11-28 03:50 .staging
drwxr-xrwx   - mespino6 hdfs          0 2018-09-26 03:12 SensorFiles
drwxr-xr-x   - mespino6 hdfs          0 2018-11-28 02:33 dualcore
drwxr-xr-x   - mespino6 hdfs          0 2018-11-08 22:25 output
drwxr-xr-x   - mespino6 hdfs          0 2018-12-03 04:05 test
drwxr-xrwx   - mespino6 hdfs          0 2018-12-01 01:22 tmp
```

Step 1: Create directories

- 2. Now you have the following 3 commands. The first is to create a directory named “data”. The second is to create a directory named "tables" inside tmp/data/.The third is to list the files and folders in /user/mespino6/tmp/data/tables

  - hdfs dfs -mkdir tmp/data
  - hdfs dfs -mkdir tmp/data/tables
  - hdfs dfs -ls tmp/data/tables

Step 1: Create directories

- 3. Run the following HDFS command to make your beeline command work:
  - hdfs dfs -chmod -R o+w tmp/

```
-bash-4.1$ hdfs dfs -chmod -R o+w tmp/
-bash-4.1$ hdfs dfs -ls tmp/
Found 1 items
drwxr-xrwx   - mespino6 hdfs          0 2018-12-01 01:22 tmp/data
```

Step 2: Downloading Data into your Oracle Big Data

- After the Hive tables are created, you can download it to your lab (or personal PC/Laptop) as follows:
- 1. Open another terminal with git bash, minty, or putty, which is to connect the Oracle cloud to download the output file and unzip the contents to - at the HDFS path "/user/mespino6/tmp/data/tables ":
  - wget -O amazon_reviews_us_Books_v1_02.tsv.gz https://s3.amazonaws.com/amazon-reviews-pds/tsv/amazon_reviews_us_Books_v1_02.tsv.gz

Step 2: continued

- 2. Unzip and put tsv.gz into hdfs
  - gunzip –c amazon_reviews_us_Books_v1_02.tsv.gz | hadoop fs -put - /user/mespino6/tmp/data/tables;

Step 3: Creating Hive Tables and Queries to Analyze Data

- 1. 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 is for multiple users' access to Hive Server of a Hadoop cluster. You have to copy and paste "!connect …" command given by the instructor at the lab page of Canvas to beeline and press enter without any password when it asks for password.
  - -bash-4.1$ beeline

Step 3: continued

- Now you have to create your database with your username to separate your tables with other users. For example, the user TtoTH should run the following:
  - Create database TtoTH;
  - use TtoTH;

```
0: jdbc:hive2://cis5200-bdcsce-4.compute-6082> use ttoth;
No rows affected (0.168 seconds)
```

Step 3: continued

- 3. In the beeline shell CLI, you need to copy and paste the following HiveQL code to create an external table “amazon_reviews_traditional” and populate it with data you downloaded:

Step 3: continued

- CREATE EXTERNAL TABLE amazon_reviews_traditional (marketplace string, customer_id string, review_id int, product_id int, product_parent string, product_title string, product_category string,star_rating int, helpful_votes int, total_votes int, vine string, verified_purchase string, review_headline string, review_body string, review_date bigint, year int)
- ROW FORMAT DELIMITED FIELDS TERMINATED BY '\t' LOCATION '/user/mespino6/tmp/data/tables';

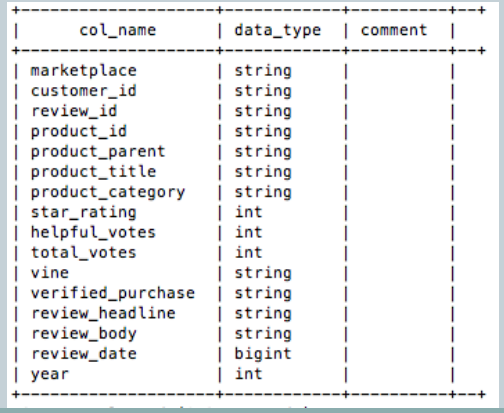
Step 3: continued

- Now you can query the content of the amazon_reviews_traditional table:
  - SELECT * FROM amazon_reviews_traditional limit 10;

```
| marketplace                    | customer_id                    | review_id                    | product_id
        | product_parent                    | product_title                    | product_category
          | NULL                    | NULL                    | NULL                    | vine
               | verified_purchase                    | review_headline                    | review_body




                    | NULL                    | NULL                    |
| US                    | 12076615                    | RQ58W7SMO911M                    | 0385730586
        | 122662979                    | Sisterhood of the Traveling Pants (Book 1)                    | Books
          | 4                    | 2                    | 3                    | N
               | N                    | this book was a great learning novel!                    | this boook was a great one that
you could learn from. it not only teaches the imponrtance of family and their values but it also deals with basic issues that teens and some kids even deal
with.  this book is about 4 best friends who are for the first time in their lives spending their summer apart. one day they are all in one of the girls roo
ms and finds a pair of pants that were tucked away in her closet.  once all four  of them try them on they realize that there is really something special ab
out these pants.  seeming as how all 4 girls are differnt shapes and sizes and somehow the pants fit all of them,  they realize that these pants were the st
art of something special.  immediatley following they decided to make up certian rules abut the pants such as you must write the best thing u did while wear
ing the pants over your summer on the right leg and also some silly things such as to \\"never pick yuor nose while wearing the pants.\\"  this book follows
 the girls throuh their summers in differnt places of the world and through all of the different obstacles that life takes them through. it can really teach
 you alot not only about what is going on around you but most imporntantly about yuorself.  i would give this book 4 stars and would reccommend it to anyone
 who seems the slihgtest bit interested.


                    | NULL                    | NULL                    |
| US                    | 12703090                    | RF6IUKMGL8SF                    | 0811828964
        | 56191234                    | The Bad Girl's Guide to Getting What You Want                    | Books
          | 3                    | 5                    | 5                    | N
               | N                    | Fun Fluff                    | If you are looking for something
 to stimulate your brain, this isn't it.  However, if you are just looking for a good laugh, you'll enjoy The Bad Girl's Guide.  It's funny and light, and d
efinitely a good way to pass a little bit of time.
```

Step 3: continued

- . Count the records in the ratings table to ensure that all 3105521 records are available:
  - SELECT COUNT(*) FROM amazon_reviews_traditional;
- You can see the structure of the table as well:
  - DESCRIBE amazon_reviews_traditional;



col_name	data_type	comment
marketplace	string	
customer_id	string	
review_id	string	
product_id	string	
product_parent	string	
product_title	string	
product_category	string	
star_rating	int	
helpful_votes	int	
total_votes	int	
vine	string	
verified_purchase	string	
review_headline	string	
review_body	string	
review_date	bigint	
year	int	

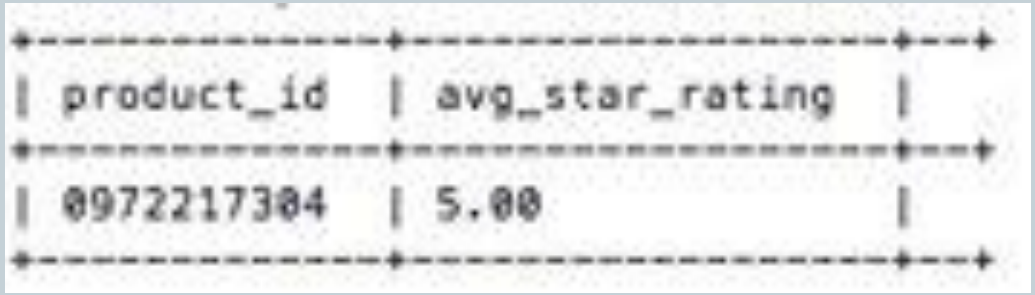
Step 3: continued

- We want to find the product that customers like most, but must guard against being misled by products that have few ratings assigned. Run the following query to find the product with the highest average using DESC among all those with at least 50 ratings, which should show the following result:

Step 3: continued

```
SELECT product_id,
FORMAT_NUMBER(avg_star_rating,2)
AS avg_star_rating
FROM (SELECT product_id, AVG(star_rating)
AS avg_star_rating, COUNT(*) AS num
FROM amazon_reviews_traditional GROUP
by product_id) amazon_reviews_traditional WH
ERE num >=50 ORDER
BY avg_star_rating DESC  LIMIT 1;
```

Step 3: continued



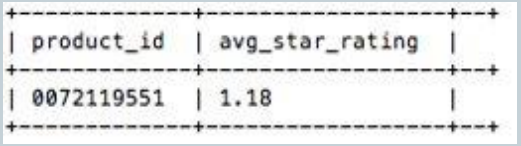
product_id	avg_star_rating
0972217304	5.00

Step 3: continued

- Rewrite, and then execute, the query above to find the product with the lowest average using ASC among products with at least 50 ratings (num >= 50). You should see that the result is product ID 007119551 with an average rating of 1.18, which should show the following result:

Step 3: continued

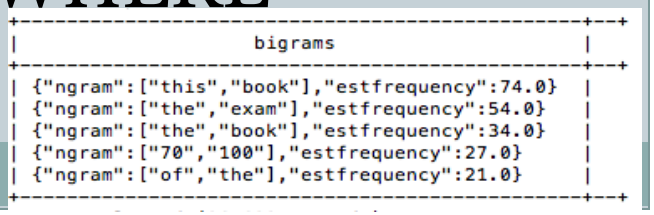
```
SELECT product_id,
FORMAT_NUMBER(avg_star_rating,2)
AS avg_star_rating
FROM (SELECT product_id, AVG(star_rating)
AS avg_star_rating,
COUNT(*)
AS num  FROM amazon_reviews_traditional
GROUP by product_id) amazon_reviews_traditio
nal  WHERE num >=50  ORDER
BY avg_star_rating ASC  LIMIT 1;
```



product_id	avg_star_rating
0072119551	1.18

Step 3: continued

- 7. The following query normalizes all comments on that product to lowercase, breaks them into individual words using the SENTENCES function, and passes those to the NGRAMS function to find the five most common bigrams (two-word combinations). Run the query in Hive:
  - SELECT EXPLODE(NGRAMS(SENTENCES(LOWER(review_body)), 2, 5)) AS bigrams FROM amazon_reviews_traditional WHERE product_id = 0072119551;



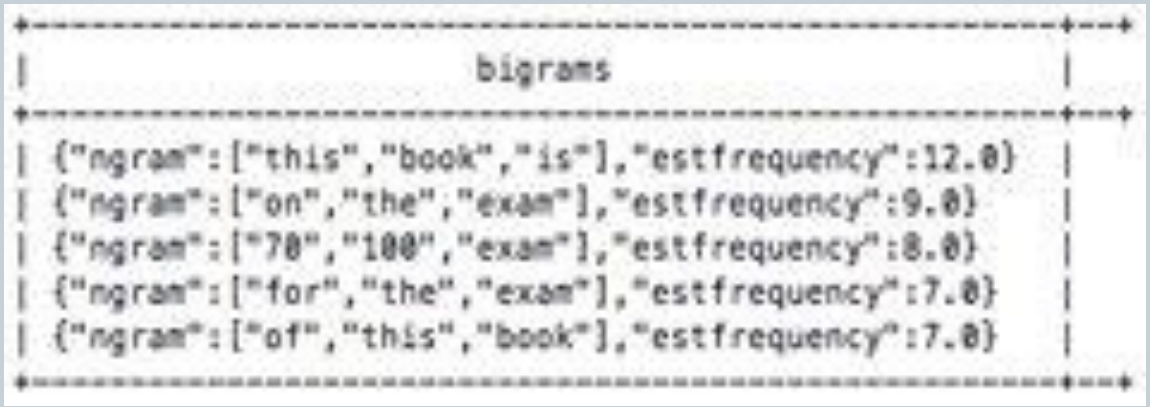
```
+-----------------------------------------------+--+
|                    bigrams                    |
+-----------------------------------------------+--+
| {"ngram":["this","book"],"estfrequency":74.0} |
| {"ngram":["the","exam"],"estfrequency":54.0}  |
| {"ngram":["the","book"],"estfrequency":34.0}  |
| {"ngram":["70","100"],"estfrequency":27.0}    |
| {"ngram":["of","the"],"estfrequency":21.0}    |
+-----------------------------------------------+--+
```

Step 3: continued

- Most of these words are too common to provide much insight. Modify the previous query to find the five most common trigrams (three-word combinations), and then run that query in Hive, which shows the following result:
  - SELECT EXPLODE(NGRAMS(SENTENCES(LOWER(review_body)), 3, 5)) AS bigrams FROM amazon_reviews_traditional WHERE product_id = 0072119551;

Step 3: continued

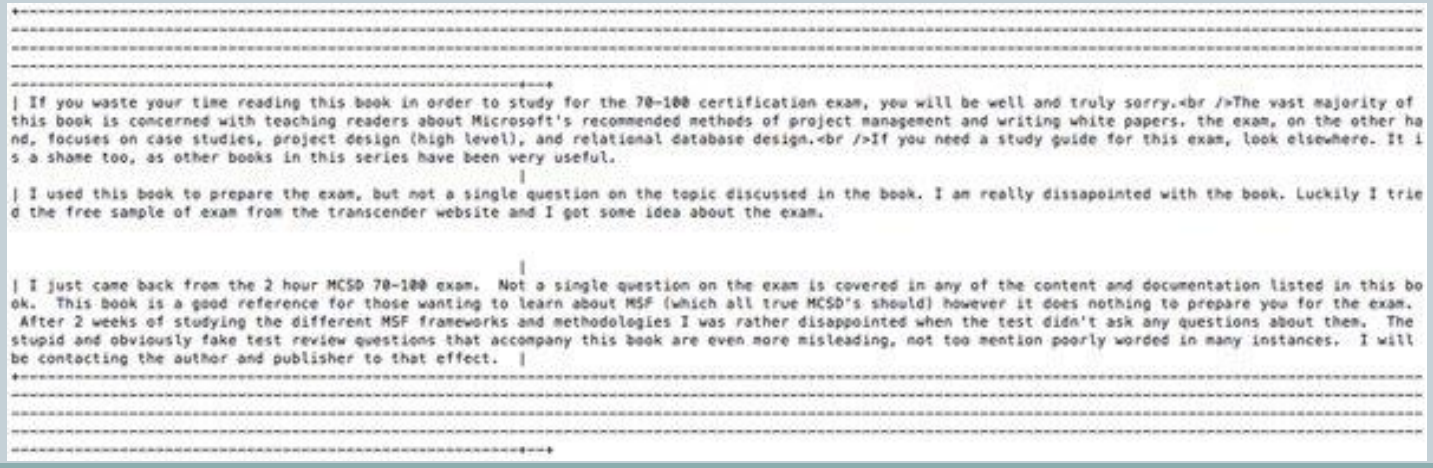
- Among the patterns you see in the result is the phrase “the exam.” This might be related to the complaints that the book does not help students study for an exam. Now that you’ve identified a specific phrase, look at a few comments that contain it by running this query:



Step 3: continued

- SELECT review_body FROM amazon_reviews_traditional WHERE product_id = 0072119551 AND review_body LIKE '%the exam%' LIMIT 3;

- You should see three comments that talk about the 70-100 certification exam and how the book does not adequately prepare the customer for the exam

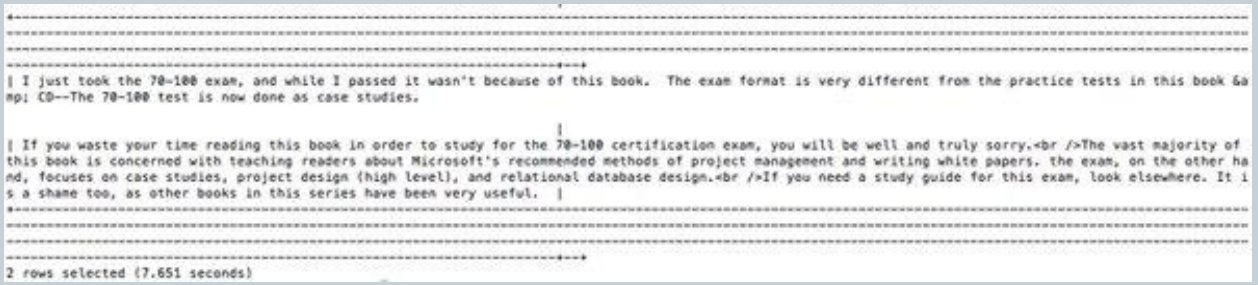


Step 3: continued

- We can infer that customers are complaining about how the book does not have relevant content for those taking the exam in question, but the comment alone doesn't provide enough detail. One of the words ("70") in that comment was also found in the list of trigrams from the earlier query. Run the following query that will find all distinct comments containing the word "70" that are associated with product ID 0072119551, which shows the result below:

Step 3: continued

- SELECT review_body FROM amazon_reviews_traditional WHERE product_id = 0072119551 AND review_body LIKE '%70%' LIMIT 2;



```
| I just took the 70-100 exam, and while I passed it wasn't because of this book.  The exam format is very different from the practice tests in this book &a
mp; CD--The 70-100 test is now done as case studies.

| If you waste your time reading this book in order to study for the 70-100 certification exam, you will be well and truly sorry.<br />The vast majority of
this book is concerned with teaching readers about Microsoft's recommended methods of project management and writing white papers. the exam, on the other ha
nd, focuses on case studies, project design (high level), and relational database design.<br />If you need a study guide for this exam, look elsewhere. It i
s a shame too, as other books in this series have been very useful.  |
2 rows selected (7.651 seconds)
```

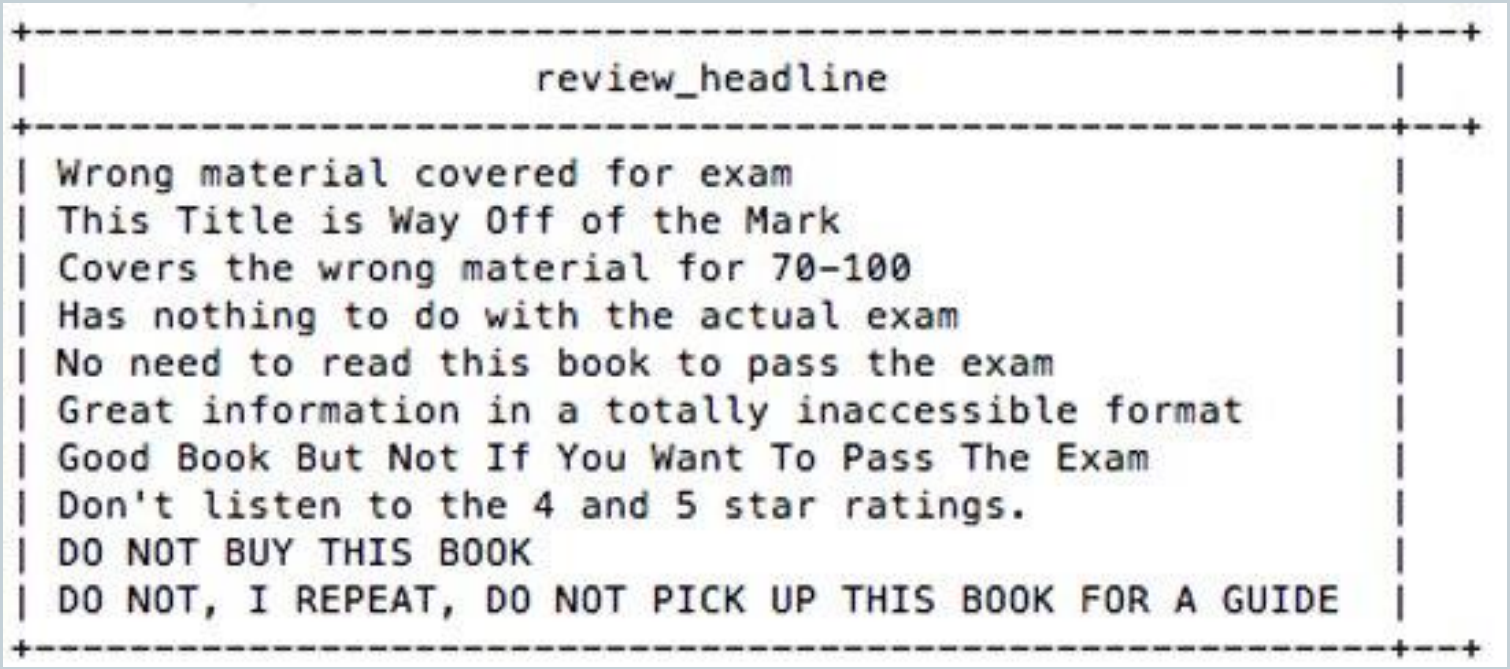
Step 3: continued

- The previous step should have displayed two comments:
  - *I just took the 70-100 exam, and while I passed it wasn't because of this book. The exam format is very different from the practice tests in this book & CD--The 70-100 test is now done as case studies.*
  - *If you waste your time reading this book in order to study for the 70-100 certification exam, you will be well and truly sorry.The vast majority of this book is concerned with teaching readers about Microsoft's recommended methods of project management and writing white papers. the exam, on the other hand, focuses on case studies, project design (high level), and relational database design. If you need a study guide for this exam, look elsewhere. It is a shame too, as other books in this series have been very useful.*

Step 3: continued

- The second comment states that the book's content is irrelevant to the 70-100 exam, unlike similar books in its series. Write and run a query that will display 10 review headlines for product ID 0072119551 in the amazon_traditional_reviews table.
  - SELECT review_headline FROM amazon_reviews_traditional WHERE product_id=0072119551 LIMIT 10;

Step 3: continued



review_headline
Wrong material covered for exam
This Title is Way Off of the Mark
Covers the wrong material for 70-100
Has nothing to do with the actual exam
No need to read this book to pass the exam
Great information in a totally inaccessible format
Good Book But Not If You Want To Pass The Exam
Don't listen to the 4 and 5 star ratings.
DO NOT BUY THIS BOOK
DO NOT, I REPEAT, DO NOT PICK UP THIS BOOK FOR A GUIDE

Step 3: Conclusion

- The query results show that the book's content is good, but the title is misleading. Customers who want a study guide for the exam are purchasing this book based on the title, but the content is not geared towards the exam.

- Based on the review_body and review_headline columns, it appears that doing text processing has helped this author uncover a title error.

Step 4: Create Hive Queries to Analyze the Sentiment of Data Using Dictionary and Download Data into your PC

- Copy the dictionary table, which has **polarity** to show each word's meaning implied as positive or negative, from the main database into the TtoTh Database:
  - CREATE TABLE TtoTH.dictionary AS select * from dictionary

```
INFO  : Map 1: 0/1
INFO  : Map 1: 0(+1)/1
INFO  : Map 1: 1/1
INFO  : Moving data to: hdfs://mycluster/apps/hive/warehouse/ttoth.db/dictionary
 from hdfs://mycluster/apps/hive/warehouse/.hive-staging_hive_2018-12-04_02-21-1
1_579_2477824766358443135-1888/-ext-10001
INFO  : Table ttoth.dictionary stats: [numFiles=1, numRows=8221, totalSize=30892
2, rawDataSize=300701]
No rows affected (4.921 seconds)
```

Step 4: continued

- We need to use the TtoTH database to query data:
  - Use TtoTH;

```
0: jdbc:hive2://cis5200-bdcsce-4.compute-6082> use ttoth;
No rows affected (0.168 seconds)
```

Step 4: continued

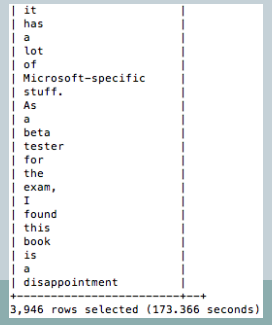
- Make sure that the dictionary table has been created using command:
  - SHOW tables;

```
0: jdbc:hive2://cis5200-bdcsce-4.compute-6082> show tables;
+-----------------------------+--+
|          tab_name           |
+-----------------------------+--+
| amazon_reviews              |
| amazon_reviews_traditional  |
| dictionary                  |
+-----------------------------+--+
3 rows selected (0.178 seconds)
0: jdbc:hive2://cis5200-bdcsce-4.compute-6082>
```

Step 4: continued

- Using EXPLODE, list all words in review body for product_id=0072119551, MCSD Analyzing Requirements: Exam 70-100 (MCSD Study Guides), which should produce **3,946** results:

  - SELECT EXPLODE(SPLIT(review_body, ' ')) AS word
    FROM amazon_reviews_traditional WHERE pro duct_id = 0072119551;

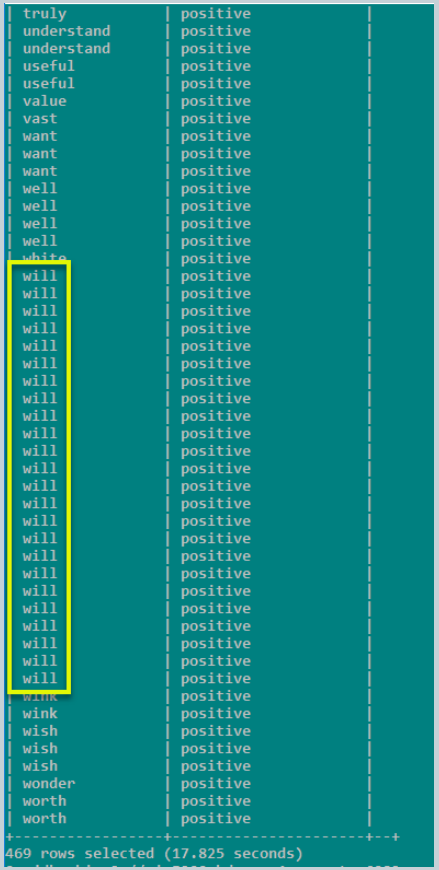


```
| it                         |
| has                        |
| a                          |
| lot                        |
| of                         |
| Microsoft-specific         |
| stuff.                     |
| As                         |
| a                          |
| beta                       |
| tester                     |
| for                        |
| the                        |
| exam,                      |
| I                          |
| found                      |
| this                       |
| book                       |
| is                         |
| a                          |
| disappointment             |
+----------------------------+---+
3,946 rows selected (173.366 seconds)
```

Step 4: continued

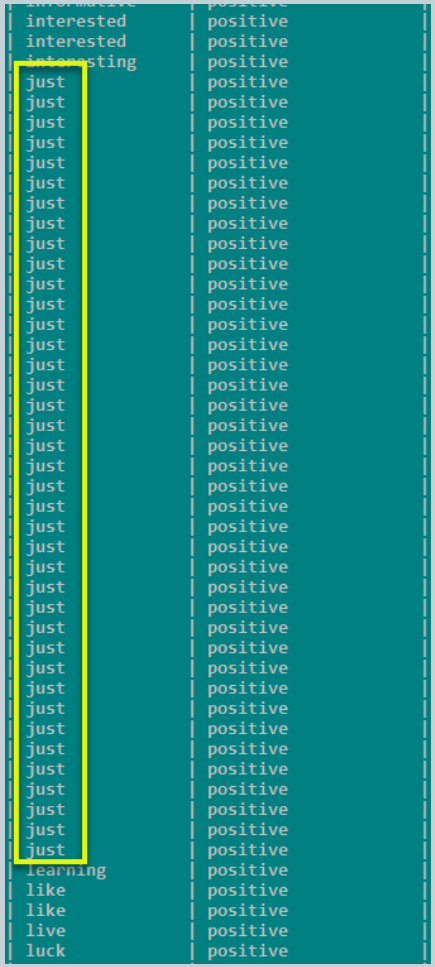
- Display words which are accounted for in the Dictionary table and order by polarity, which should display **469** Results:
  - SELECT words, dictionary.polarity FROM (SELECT explode(split(review_body, ' ')) AS words FROM amazon_reviews_traditional WHERE product_id = 0072119551) a join Dictionary on words=word ORDER by dictionary.polarity, words;

Step 4: continued



truly	positive
understand	positive
understand	positive
useful	positive
useful	positive
value	positive
vast	positive
want	positive
want	positive
want	positive
well	positive
well	positive
well	positive
well	positive
white	positive
will	positive
will	positive
will	positive
will	positive
will	positive
will	positive
will	positive
will	positive
will	positive
will	positive
will	positive
will	positive
will	positive
will	positive
will	positive
will	positive
will	positive
will	positive
will	positive
will	positive
will	positive
will	positive
will	positive
will	positive
will	positive
wink	positive
wink	positive
wish	positive
wish	positive
wish	positive
wonder	positive
worth	positive
worth	positive

469 rows selected (17.825 seconds)



interested	positive
interested	positive
interesting	positive
just	positive
just	positive
just	positive
just	positive
just	positive
just	positive
just	positive
just	positive
just	positive
just	positive
just	positive
just	positive
just	positive
just	positive
just	positive
just	positive
just	positive
just	positive
just	positive
just	positive
just	positive
just	positive
just	positive
just	positive
just	positive
just	positive
just	positive
just	positive
just	positive
just	positive
just	positive
just	positive
just	positive
just	positive
just	positive
just	positive
just	positive
just	positive
just	positive
just	positive
learning	positive
like	positive
like	positive
live	positive
luck	positive

Step 4: continued

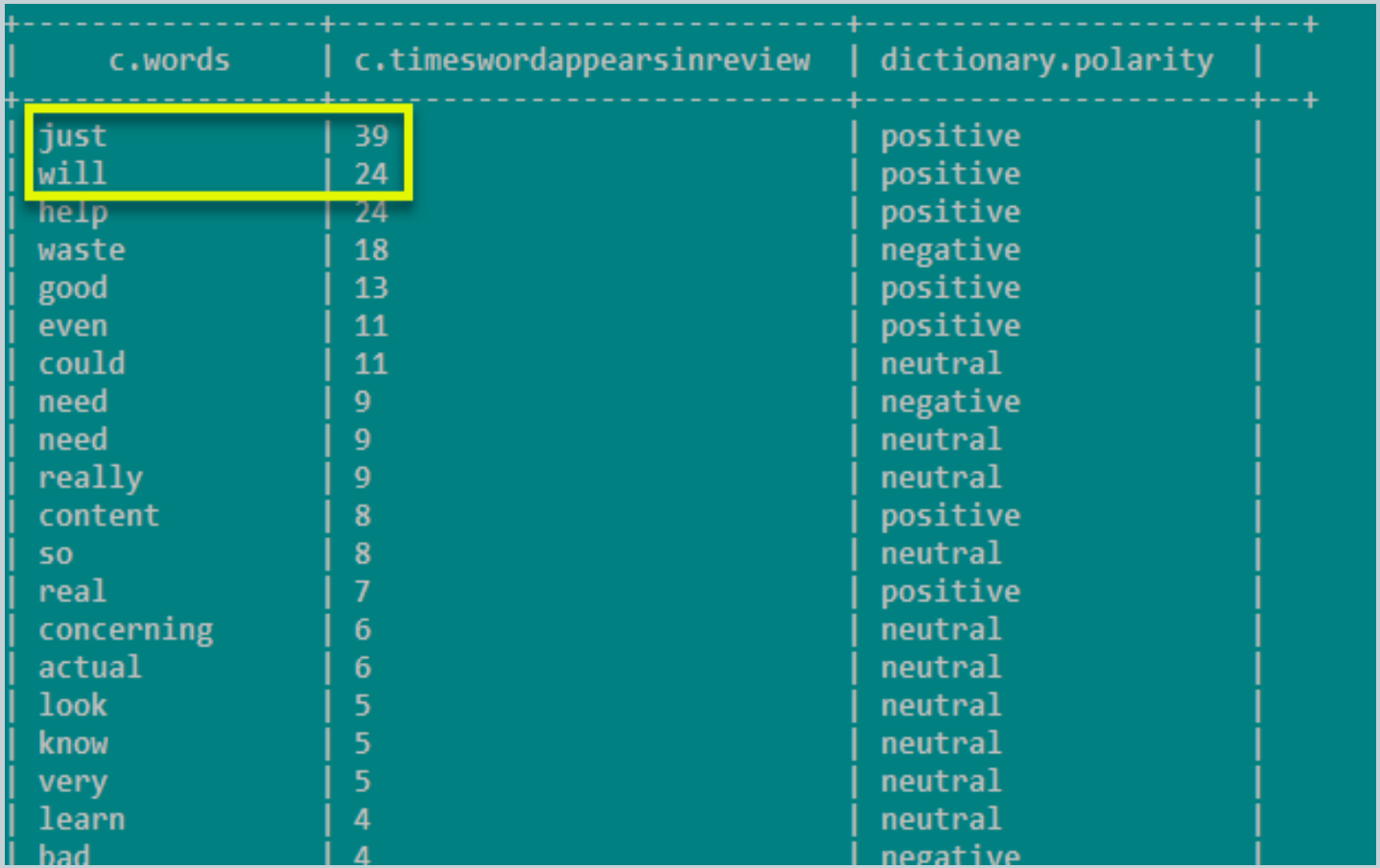
- Show totals by polarity to determine if the reviews are mainly positive or negative:
  - SELECT polarity, sum(TimesWordAppearsInReview) FROM(SELECT DISTINCT words, TimesWordAppearsInReview, polarity FROM (SELECT words, COUNT(words) as TimesWordAppearsInReview FROM(SELECT words FROM(SELECT explode(split(review_body, ' ')) AS words FROM amazon_reviews_traditional WHERE product_id = 0072119551) a join Dictionary on words=word ORDER BY words) b GROUP BY words) c JOIN Dictionary on words=word ORDER BY words, polarity) d GROUP by polarity;

```
+-----------+------+--+
| polarity  | _c1  |
+-----------+------+--+
| positive  | 223  |
| negative  | 114  |
| neutral   | 145  |
+-----------+------+--+
3 rows selected (17.919 seconds)
```

Step 4: continued

- Even though this book has the lowest average rating, it's "positive" polarity is such a high number. This may be an error, so run the following hive command to determine what positive words may be skewing the results:
  - SELECT DISTINCT c.words, c.TimesWordAppearsInReview, Dictionary.polarity
    FROM (SELECT words, COUNT(words) as TimesWordAppearsInReview
    FROM(SELECT words FROM(SELECT explode(split(review_body, ' ')) as words
    FROM amazon_reviews_traditional
    WHERE product_id = 0072119551) a JOIN Dictionary on words=word
    ORDER BY words) b GROUP by words) c JOIN Dictionary on words=word
    ORDER by TimesWordAppearsInReview DESC;

Step 4: continued



c.words	c.timeswordappearsinreview	dictionary.polarity
just	39	positive
will	24	positive
help	24	positive
waste	18	negative
good	13	positive
even	11	positive
could	11	neutral
need	9	negative
need	9	neutral
really	9	neutral
content	8	positive
so	8	neutral
real	7	positive
concerning	6	neutral
actual	6	neutral
look	5	neutral
know	5	neutral
very	5	neutral
learn	4	neutral
bad	4	negative

Step 4: continued

- The words “just” and “will” appear 39 and 24 times respectively. These may or may not be positive in most contexts. We can look at a few comments that contain these words to see if the comment is positive or negative overall:
  - SELECT substr(review_body,0,100)
    FROM amazon_reviews_traditional
    WHERE review_body like '%just%' and product_id = 0072119551;

```
+----------------------------------------------------------------------------------------------------+--+
|                                                _c0                                                 |
+----------------------------------------------------------------------------------------------------+--+
| I just took the 70-100 exam, and while I passed it wasn't because of this book.  The exam format is  |
| I just came back from the 2 hour MCSD 70-100 exam.  Not a single question on the exam is covered in  |
| This review mostly just for spite to lower rating.  Look for my copy of this book on auctions!  MSF  |
| I give it negative 5 stars. What a fool I was not to have read these reviews earlier. I bought the b |
| It could be argued that this book contains good reference material for Analyzing Requirements.  Argu |
| I've just passed the exam. This book is not useful for passing exam 70-100. If you buy it for the ex |
| Syngress should be shot, along with the folks who reviewed and approved the content for logo use.  I |
| Having just taken Exam 70-100, I can say that this book does NOT prepare you for it.<br />The book d |
| Having read the book and taken the test, I must say that this book does a good job covering the MSF, |
| This book mainly covers MSF. If you are interested in MSF then it might be worth buying. I just took |
| This is a great book if you want to learn about the Microsoft Solutions Framework (MSF). I just wrot |
| I just did the exam.<br />This is a good book, and after you ignore the first 5 chapters it DOES fol |
| Having just flunked the exam miserably after studying all 3 books currently available, only one thre |
+----------------------------------------------------------------------------------------------------+--+
13 rows selected (16.154 seconds)
0: jdbc:hive2://cis5200-bdcsce-4.compute-6082>
```

Step 4: continued

- We can see from the comments that they are not positive overall. The words “just” and “will” are skewing our results. We need to clean up the data by omitting these words by using the following HIVE command:
  - CREATE TABLE TtoTH.dictionary_adj AS select * from dictionary where **word not in ('just','will')**;

```
INFO  : Map 1: 0/1
INFO  : Map 1: 0(+1)/1
INFO  : Map 1: 1/1
INFO  : Moving data to: hdfs://mycluster/apps/hive/warehouse/ttoth.db/dictionary
_adj from hdfs://mycluster/apps/hive/warehouse/ttoth.db/.hive-staging_hive_2018-
12-04_02-34-40_631_711012909501973834-1888/-ext-10001
INFO  : Table ttoth.dictionary_adj stats: [numFiles=1, numRows=8216, totalSize=3
08756, rawDataSize=300540]
No rows affected (4.49 seconds)
```

Step 4: continued

- Show totals by polarity using the new dictionary without words “just” and “will”
  - SELECT polarity, SUM(TimesWordAppearsInReview) FROM (SELECT DISTINCT words, TimesWordAppearsInReview, polarity FROM (SELECT words, COUNT(words) as TimesWordAppearsInReview FROM (SELECT words FROM (SELECT explode(split(review_body, ' ')) AS words FROM amazon_reviews_traditional WHERE product_id = 0072119551) a JOIN dictionary_adj on words=word ORDER BY words) b GROUP BY words) c JOIN dictionary_adj on words=word ORDER BY words, polarity) d GROUP BY polarity;

```
INFO  : Map 1: 5/5      Map 6: 1/1      Map 7: 1/1      Reducer 2: 1/1  Reducer
3: 8/8  Reducer 4: 1/1  Reducer 5: 3(+1)/4
INFO  : Map 1: 5/5      Map 6: 1/1      Map 7: 1/1      Reducer 2: 1/1  Reducer
3: 8/8  Reducer 4: 1/1  Reducer 5: 4/4
+-----------+------+--+
| polarity  | _c1  |
+-----------+------+--+
| positive  | 160  |
| negative  | 114  |
| neutral   | 145  |
+-----------+------+--+
3 rows selected (17.855 seconds)
```

Step 4: continued

- Now that we've cleaned our data we can get around to computing the sentiment. Use the following 3 Hive commands to create 3 views that will allow us to do that:
  - CREATE view IF NOT EXISTS temp_One AS
    SELECT product_id, words
    FROM amazon_reviews_traditional lateral view
    EXPLODE(SENTENCES(LOWER(review_body))) dummy AS
    words;
  - CREATE view IF NOT EXISTS temp_Two AS
    SELECT product_id, word
    FROM temp_One
    lateral view explode(words) dummy AS word;

Step 4: continued

- CREATE view IF NOT EXISTS temp_Three AS
- SELECT product_id,
  temp_two.word,
  case d.polarity
  when 'negative' then -1
  when 'positive' then 1
  else 0 end as polarity
  from temp_two left outer join dictionary_adj d on temp_two.word = d.word;

```
0: jdbc:hive2://cis5200-bdcsce-4.compute-6082> CREATE view IF NOT EXISTS temp_On
e AS SELECT product_id, words FROM amazon_reviews_traditional lateral view EXPLO
DE(SENTENCES(LOWER(review_body))) dummy AS words; CREATE view IF NOT EXISTS temp
_Two AS SELECT product_id, word FROM temp_One lateral view explode( words ) dumm
y AS word; CREATE view IF NOT EXISTS temp_Three AS SELECT product_id, temp_two.w
ord, case d.polarity when 'negative' then -1 when 'positive' then 1 else 0 end a
s polarity from temp_two left outer join dictionary_adj d on temp_two.word = d.w
ord;
No rows affected (0.22 seconds)
No rows affected (0.211 seconds)
No rows affected (0.242 seconds)
```

temp_three.product_id	temp_three.word	temp_three.polarity
product_id	review_body	0
0385730586	this	0
0385730586	boook	0
0385730586	was	0
0385730586	a	0
0385730586	great	1
0385730586	one	0
0385730586	that	0
0385730586	you	0
0385730586	could	0

Step 4: continued

- We can determine overall sentiment of each product id using average sentiment:
  - CREATE View IF NOT EXISTS productsreview_sentiment AS
    SELECT product_id,
    CASE
    when sum(polarity) > 0 then 'positive'
    when sum(polarity) < 0 then 'negative'
    ELSE 'neutral' end as sentiment, sum(polarity) as sentiment_rating, AVG(polarity) AS sentiment_average FROM temp_three GROUP by product_id;

```
0: jdbc:hive2://cis5200-bdcsce-4.compute-6082> CREATE View IF NOT EXISTS product
sreview_sentiment AS SELECT product_id, CASE when sum( polarity ) > 0 then 'posi
tive' when sum( polarity ) < 0 then 'negative' ELSE 'neutral' end as sentiment,
sum( polarity ) as sentiment_rating, AVG(polarity) AS sentiment_average FROM tem
p_three GROUP by product_id;
No rows affected (0.24 seconds)
```

Step 4: continued

- . You have to query data from productreview_sentiment to see if it has the correct data and values:
  - SELECT * from productsreview_sentiment LIMIT 10;

product_id	sentiment	sentiment_rating	sentiment_average
0001527355	positive	21	0.11229946524064172
0002200155	positive	10	0.1388888888888889
0002151928	positive	1	0.01694915254237288
0002250381	positive	3	0.1
0001983679	positive	8	0.0761904761904762
0002161621	positive	10	0.046296296296296294
0001006002	positive	6	0.09230769230769231
0001857029	positive	11	0.07857142857142857
0002000172	positive	3	0.036585365853658534
000215725X	positive	38	0.031198686371100164

Step 4: Create Star Rating view

- CREATE View IF NOT EXISTS productsreview_avg_star_rating AS SELECT product_id, FORMAT_NUMBER(avg_star_rating,2) AS avg_star_rating FROM (SELECT product_id, AVG(star_rating) AS avg_star_rating, COUNT(*) AS num FROM amazon_reviews_traditional GROUP by product_id) amazon_reviews_traditional WHERE num >=100;

```
0: jdbc:hive2://cis5200-bdcsce-4.compute-6082> CREATE View IF NOT EXISTS product
sreview_avg_star_rating AS SELECT product_id, FORMAT_NUMBER(avg_star_rating,2) A
S avg_star_rating FROM (SELECT product_id, AVG(star_rating) AS avg_star_rating,
COUNT(*) AS num FROM amazon_reviews_traditional GROUP by product_id) amazon_revi
ews_traditional WHERE num >=100;
No rows affected (0.225 seconds)
```

Step 4: Consolidate the Star Ratings and Sentiment Information

- CREATE View IF NOT EXISTS consolidates_sentiment_starrating AS SELECT b.product_id, b.sentiment, FORMAT_NUMBER(b.sentiment_average,4) as Sentiment_Range, a.avg_star_rating FROM productsreview_avg_star_rating a LEFT OUTER JOIN productsreview_sentiment b on a.product_id = b.product_id;

```
0: jdbc:hive2://cis5200-bdcsce-4.compute-6082> CREATE View IF NOT EXISTS consoli
dates_sentiment_starrating AS SELECT b.product_id, b.sentiment, FORMAT_NUMBER(b.
sentiment_average,4) as Sentiment_Range, a.avg_star_rating FROM productsreview_a
vg_star_rating a LEFT OUTER JOIN productsreview_sentiment b on a.product_id = b.
product_id;
No rows affected (0.276 seconds)
```

consolidates_sentiment_starrating.product_id	consolidates_sentiment_starrating.sentiment	consolidates_sentiment_starrating.sentiment_range	consolidates_sentiment_starrating.avg_star_rating
0060275103	positive	0.0640	4.81
0061015725	positive	0.0125	3.36
0312187459	positive	0.0478	4.46
0345335511	positive	0.0438	4.12
0345378482	positive	0.0076	4.14
0380814676	positive	0.0299	4.45
0385335482	positive	0.0354	4.35
038549081X	positive	0.0210	4.16
0385493622	positive	0.0369	4.68
0385501560	positive	0.0160	4.07

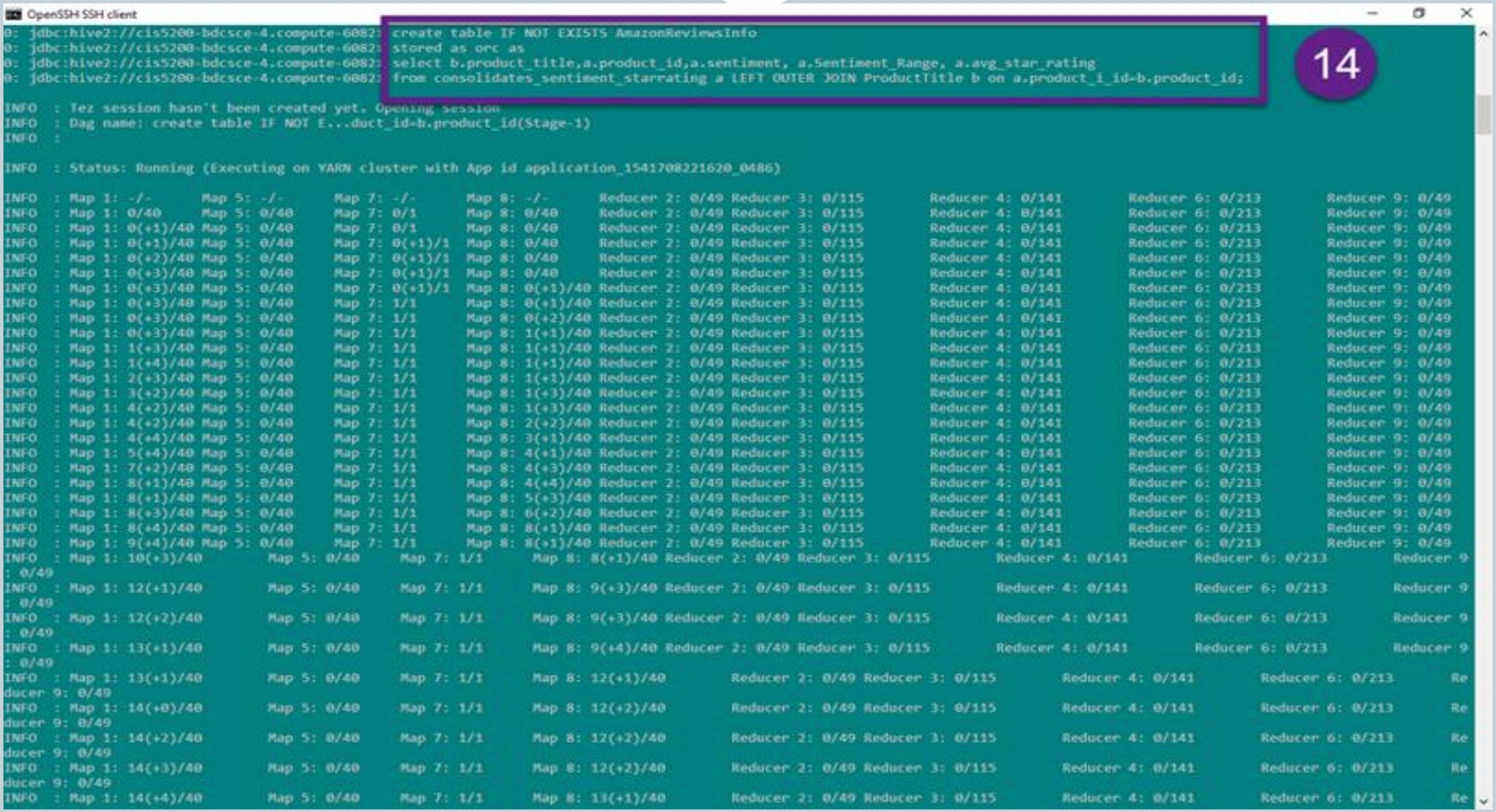
Step 4: Create Product Name and Product Table View

- CREATE View IF NOT EXISTS ProductTitle AS SELECT DISTINCT product_title, product_id FROM amazon_reviews_traditional;

```
0: jdbc:hive2://cis5200-bdcsce-4.compute-6082> CREATE View IF NOT EXISTS Product
Title AS SELECT DISTINCT product_title, product_id FROM amazon_reviews_tradition
al;
No rows affected (0.226 seconds)
```

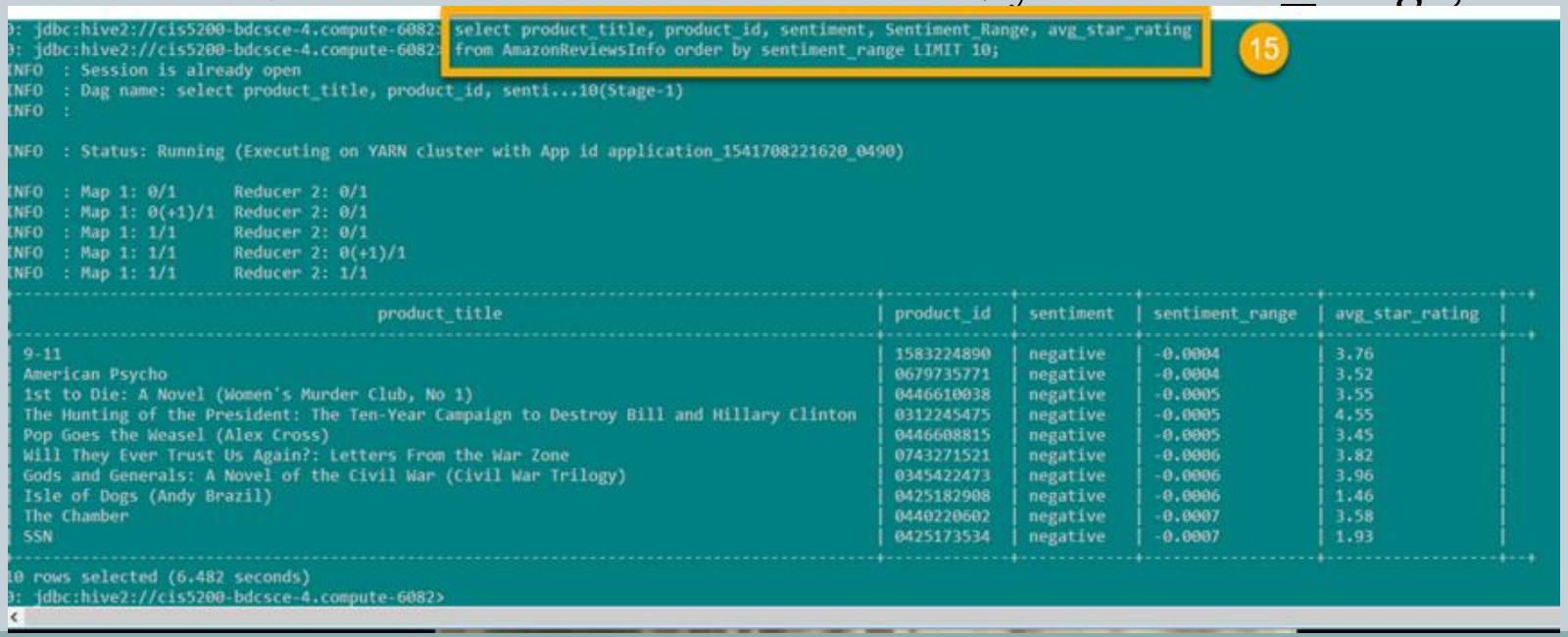
- Create Table with all of our sentiment information:
  - CREATE table IF NOT EXISTS AmazonReviewsInfo STORED AS orc AS SELECT b.product_title,a.product_id,a.sentiment, a.Sentiment_Range, a.avg_star_rating FROM consolidates_sentiment_starrating a LEFT OUTER JOIN ProductTitle b on a.product_id=b.product_id;

Step 4: continued



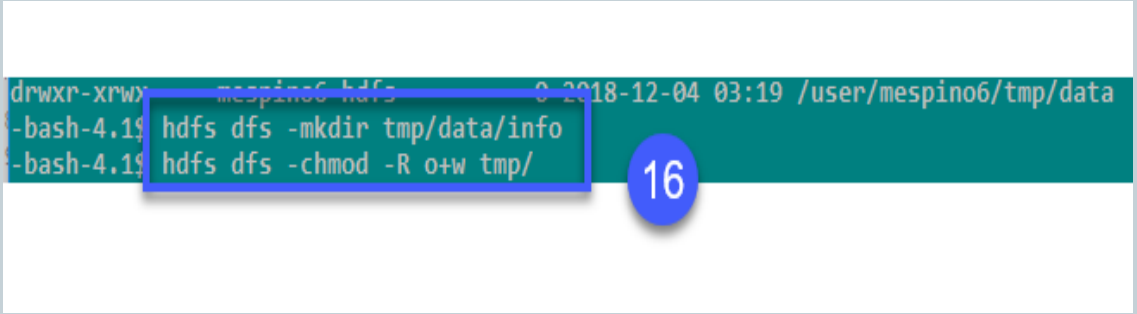
Step 4: continued

- You have to query data from the table to see if it has the correct data and values:
  - SELECT product_id, sentiment, Sentiment_Range, avg_star_rating
    FROM AmazonReviewsInfo order by sentiment_range;



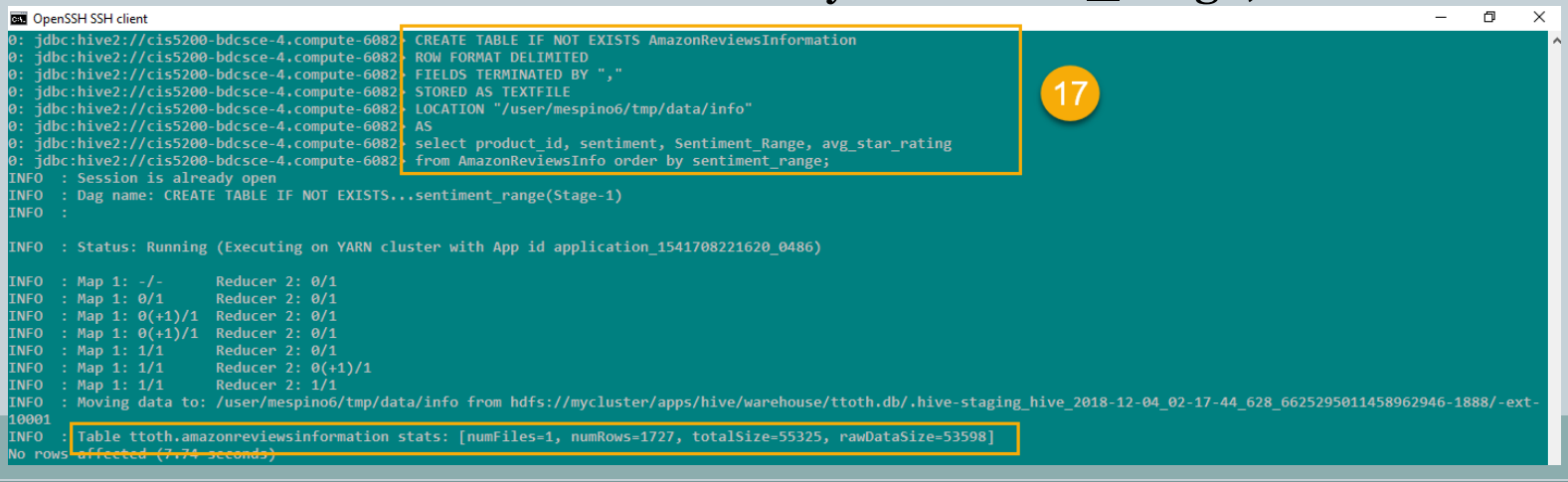
Step 4: continued

- Run the following shell commands to make sure that the directory tmp/data/info is there and that beeline command will work:
  - hdfs dfs -mkdir tmp/data/info
  - hdfs dfs -chmod -R o+w tmp/



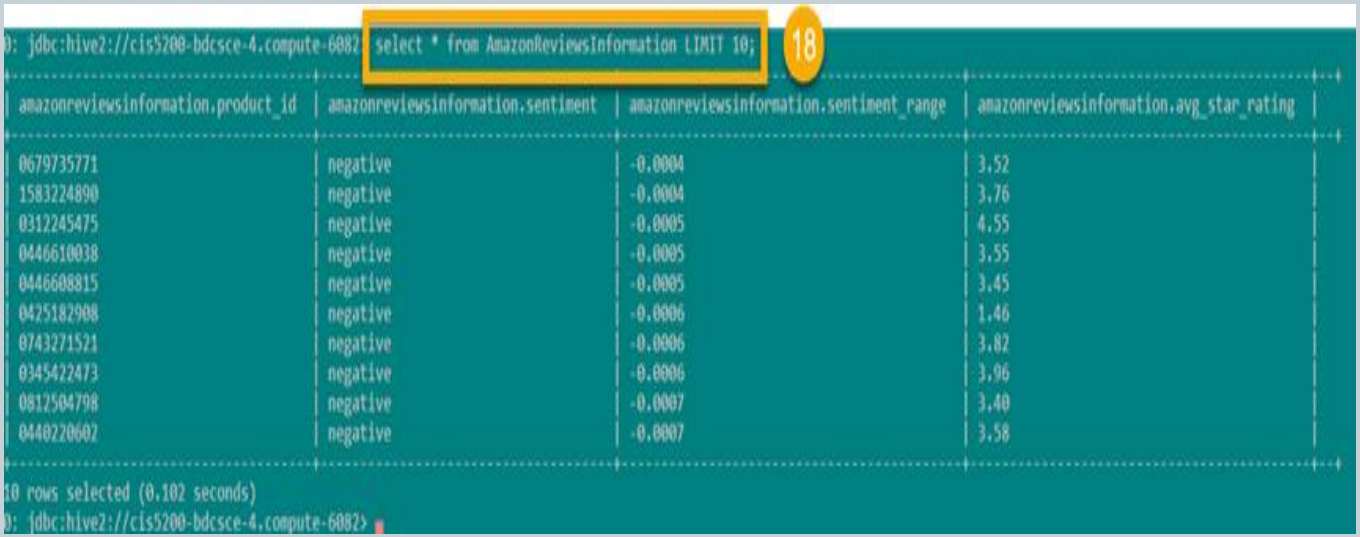
Step 4: Download file to HDFS

- Download the file to HDFS path"/user/mespino6/tmp/data/info":
  - CREATE TABLE IF NOT EXISTS AmazonReviewsInformation
    ROW FORMAT DELIMITED
    FIELDS TERMINATED BY ","
    STORED AS TEXTFILE
    LOCATION "**/user/mespino6/tmp/data/info**"
    AS select product_id, sentiment, Sentiment_Range, avg_star_rating from AmazonReviewsInfo order by sentiment_range;



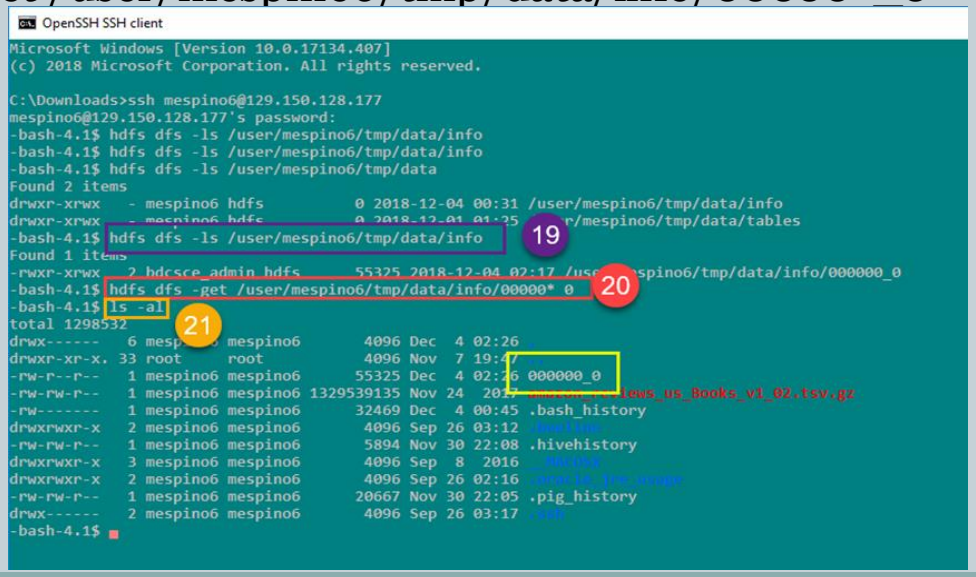
Step 4: Check downloaded data

- You have to query data from the table to see if it has the correct data and values:
  - SELECT * from AmazonReviewsInfo LIMIT 10;



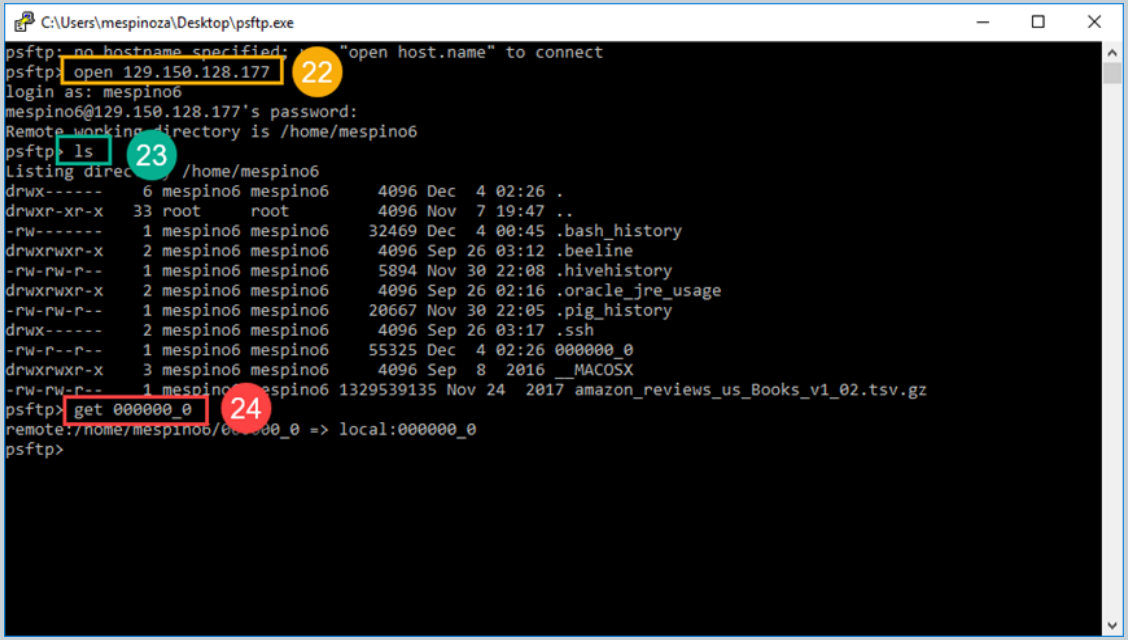
Step 4: Locate output file

- Open another terminal with git bash, minty, or putty, which is to connect the Oracle Cloud to download the output file oooooo_o at the HDFS path "/user/mespino6/tmp/data/info":
  - hdfs dfs -ls /user/mespino6/tmp/data/info
  - hdfs dfs -get /user/mespino6/tmp/data/info/ooooo*_o



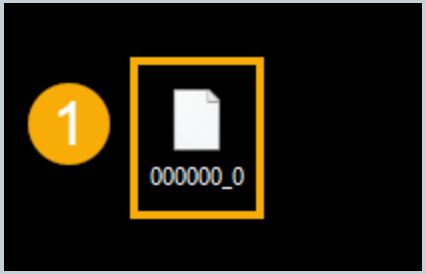
Step 4: Download output file

- For Windows user, you may use psftp to download the file. You need to download it at http://the.earth.li/~sgtatham/putty/latest/w64/psftp.exe
- In order to download 000000_0, you have to run psftp as follows:
  - open 129.150.128.177
  - Ls
  - get 000000_0

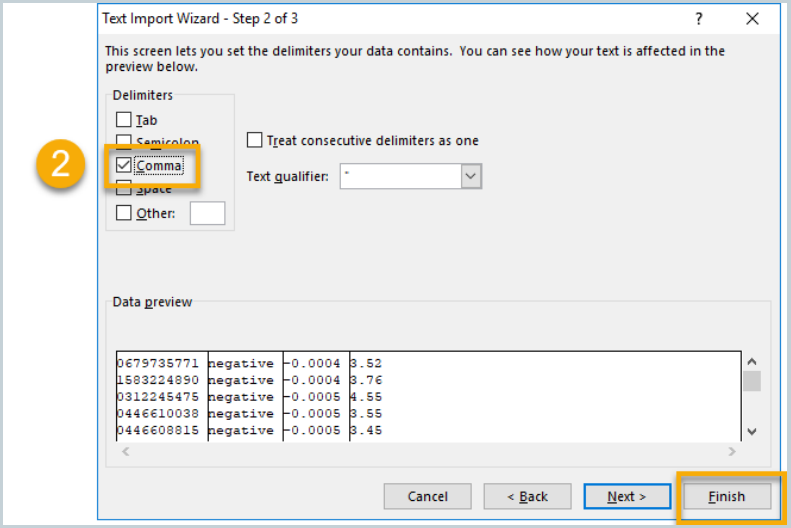


STEP 5: Loading Data Into Power BI

- You have to open the following file in Excel using the Text Import Wizard.

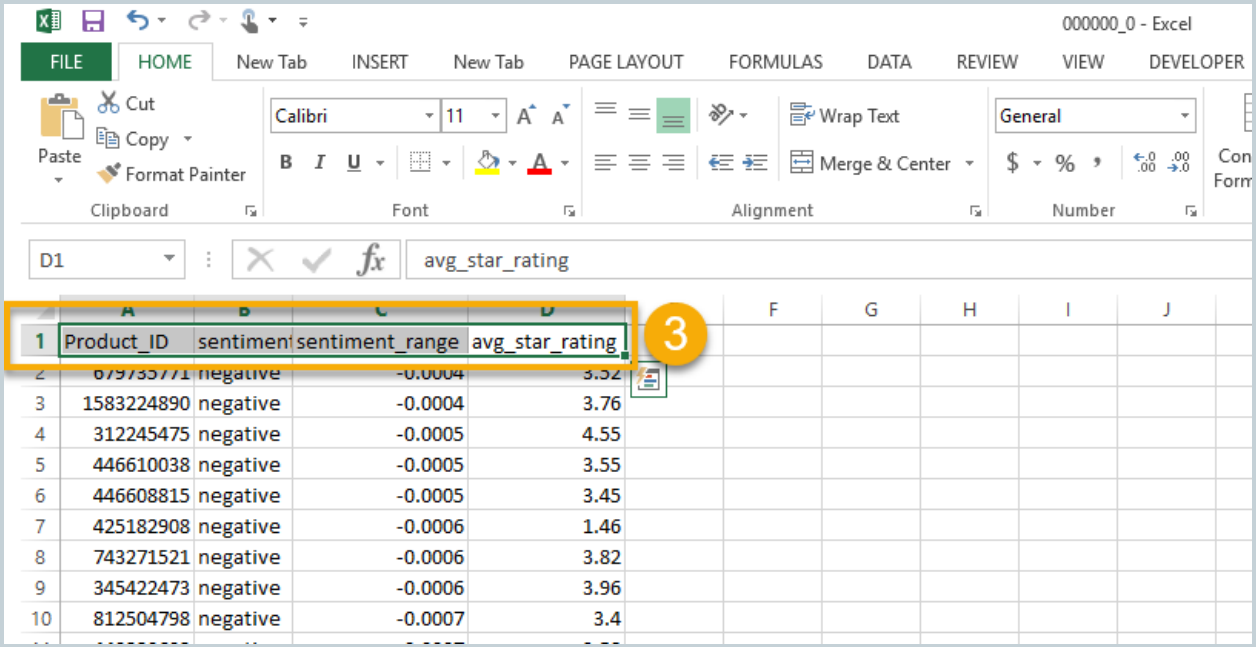


- You have to specify Comma as a Delimiter.



Step 5: Insert headers

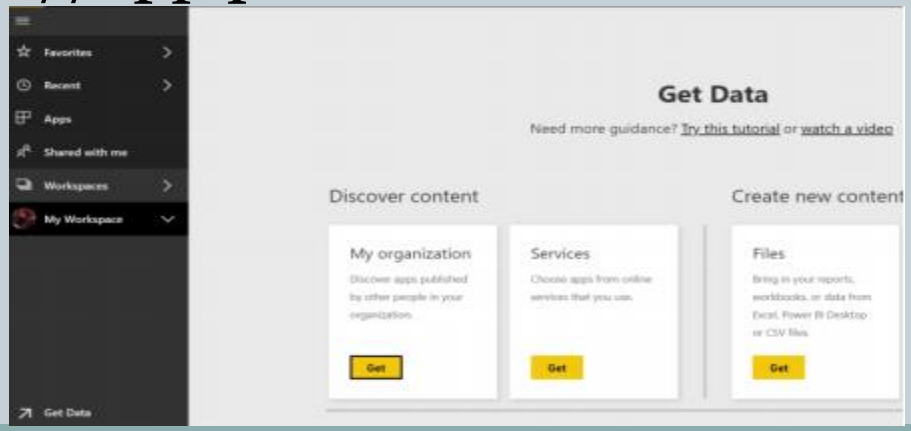
- For the first row of the file, you need to insert the header to each column as follows: Product_ID sentiment sentiment _range avg_star_rating



	A	B	C	D
1	Product_ID	sentiment	sentiment_range	avg_star_rating
2	679735771	negative	-0.0004	3.52
3	1583224890	negative	-0.0004	3.76
4	312245475	negative	-0.0005	4.55
5	446610038	negative	-0.0005	3.55
6	446608815	negative	-0.0005	3.45
7	425182908	negative	-0.0006	1.46
8	743271521	negative	-0.0006	3.82
9	345422473	negative	-0.0006	3.96
10	812504798	negative	-0.0007	3.4

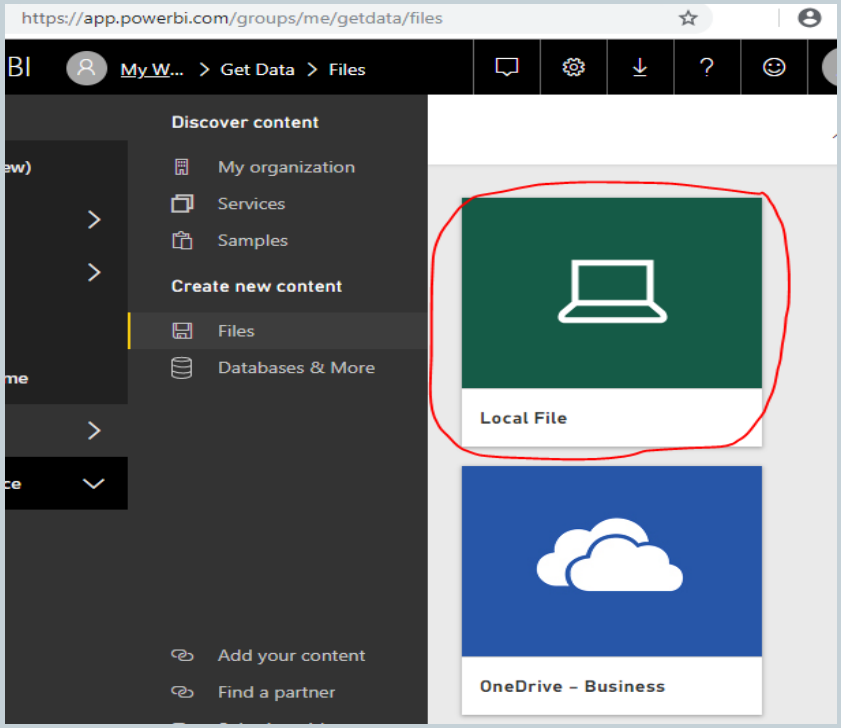
Step 5: Load Power BI

- Then you have to save the file as comma separated value format, that is, as 000000_0.csv
- You need to sign in Power BI website using your school account to import the result data into Power BI to visually explore the data.
- Open a web browser and go to sign in with your school account at: htts://app.powerbi.com

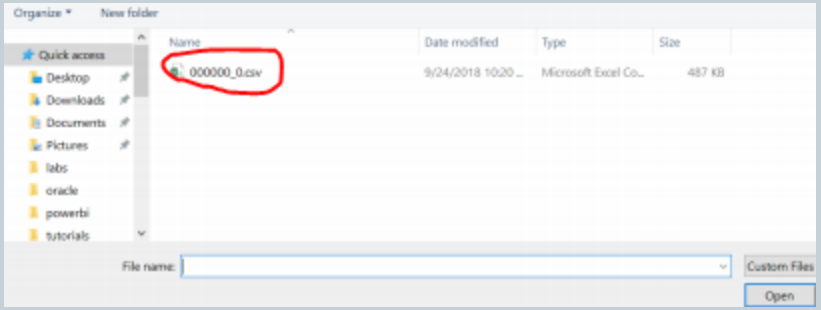


Step 5: load csv file

- Once you sign in, you will see the following pages. Select Local File to upload your output file "000000_0.csv"

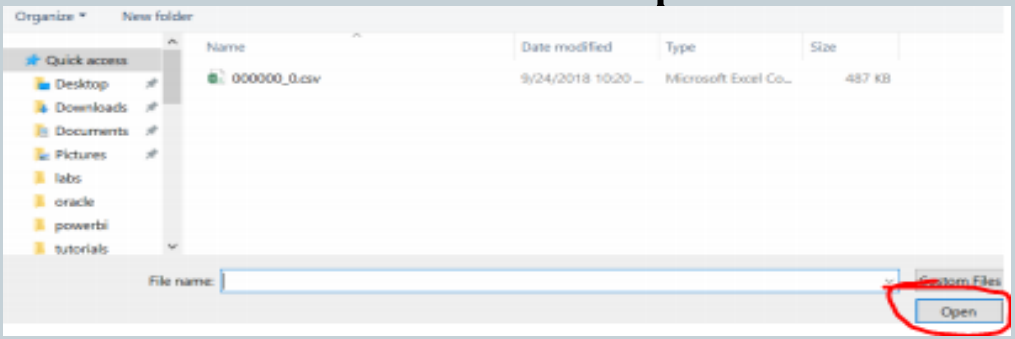


You will see the following window popped up:

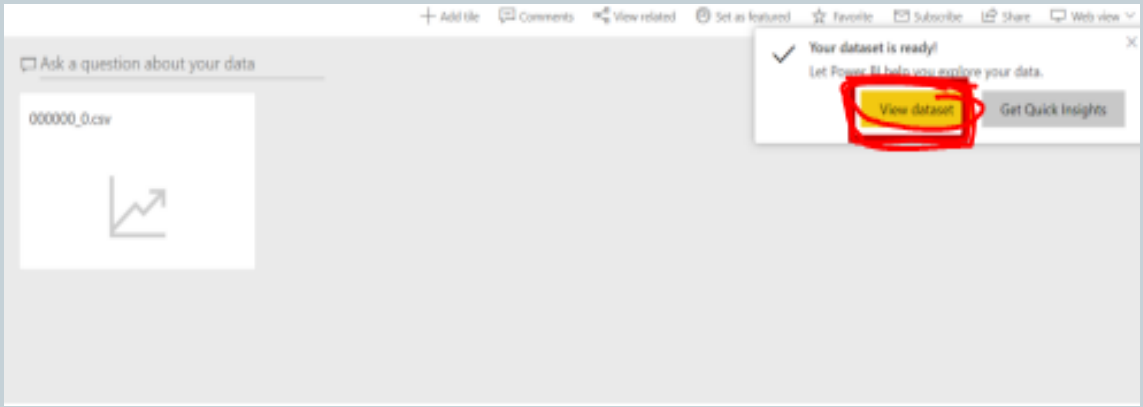


Step 5: View dataset

- Select the file and open it:



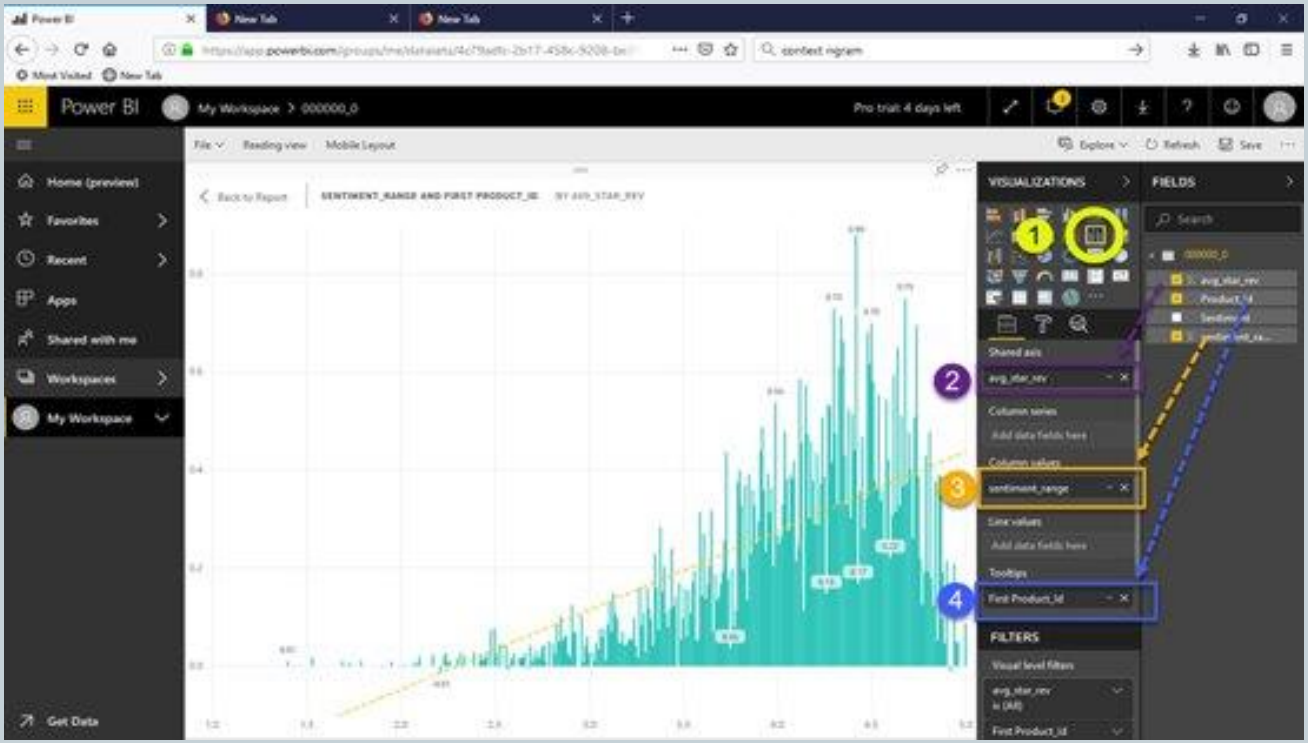
Now you will see the following page at your Power BI.
Select “View dataset”:



STEP 6: Visualizing Data In Power BI

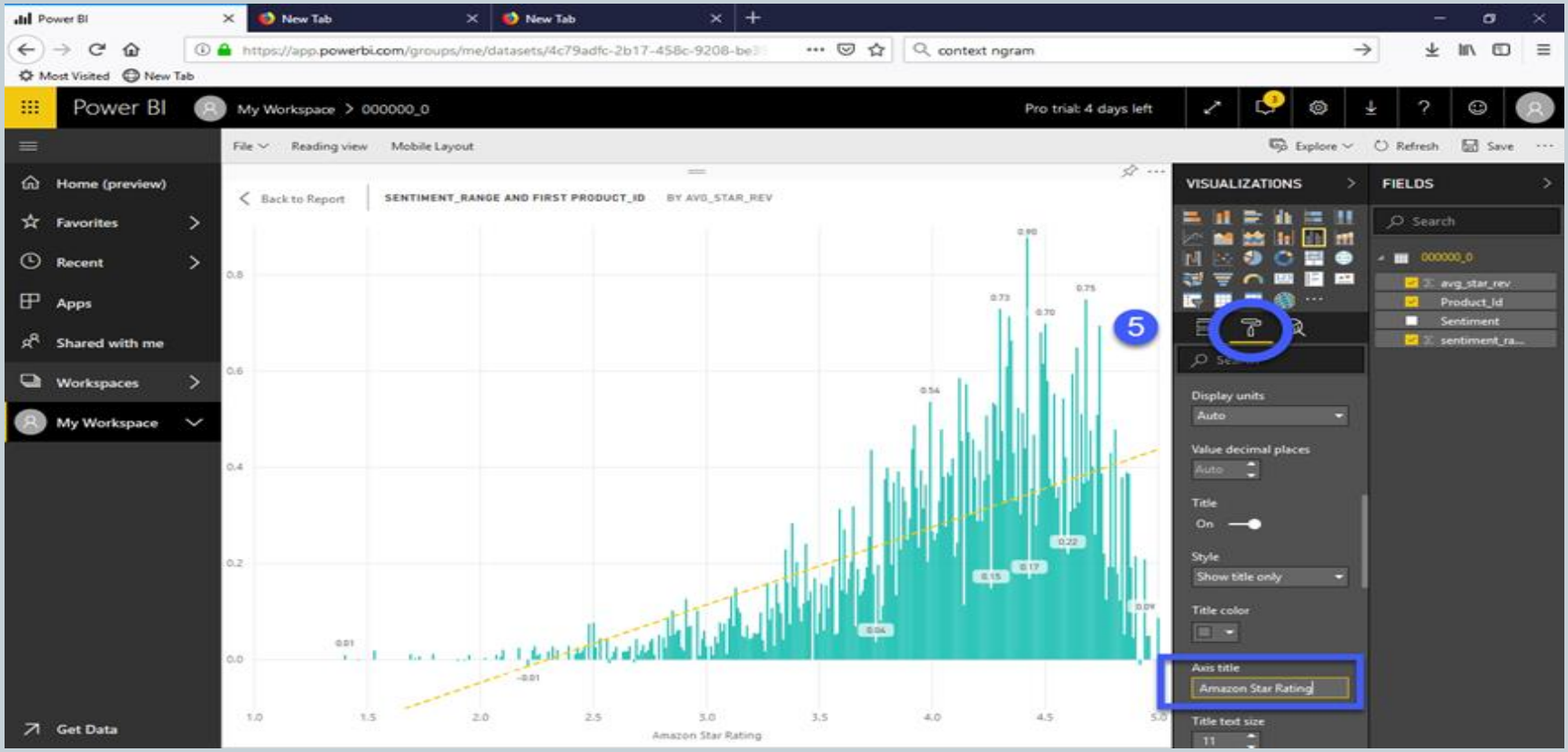
1. Select graph
2. Drag avg_star_rating under shared axis
3. Drag sentiment_range under Column values
4. Drag Product_Id under Tooltips

Step 6: continued

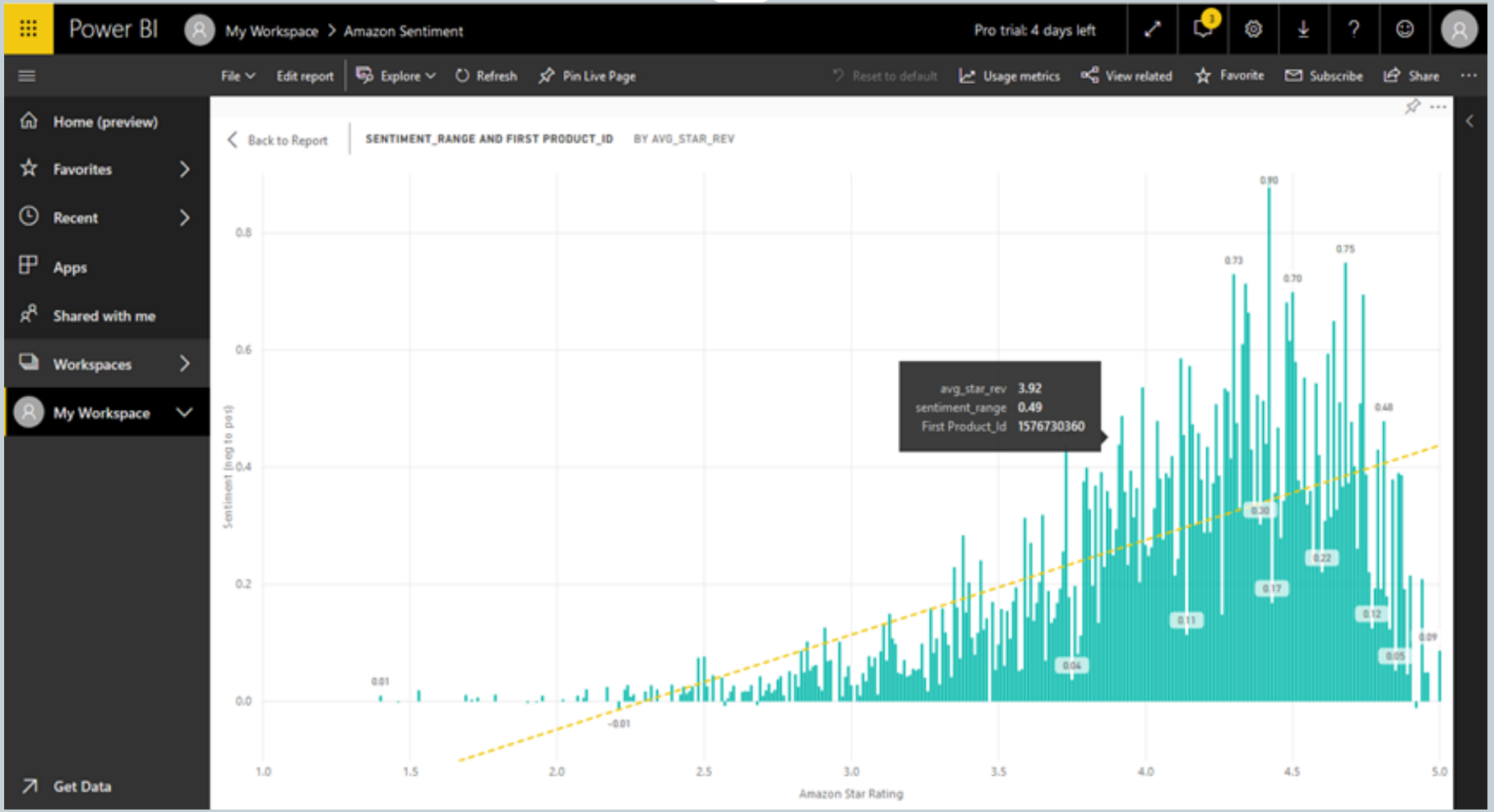


Step 6: continued

- Go to design tab and change Axis title to Amazon Star Rating



Step 6: Finale



References

- Hive Twitter Sentiment Analysis using Oracle Cloud Tutorial
- HOW TO ANALYZE MACHINE AND SENSOR DATA, http://hortonworks.com/hadooptutorial/how-to-analyze-machine-and-sensor-data/
- https://azure.microsoft.com/en-us/documentation/articles/hdinsight-use-hive/

Github Link

- Github link:
- https://github.com/ttothcalstate/Final/

Conclusion

- In this tutorial you learned how to use Oracle Cloud and Power BI to analyze Amazon data using Apache Hive. Going through this workflow, you understand how the raw data is first uploaded to Oracle Cloud, unzipped, and then loaded to Hive tables to perform queries. Finally, you finish by learning how to import the results of your Hive queries into Power BI.

- Questions, comments, concerns, grievances?