1.

2.

3.

4.

In Spark, to get the Top Ten Positive Stories, We ran the query:

Top Ten Positive Stories(Highest Percentage of Positive Comments about Trump)

|  |  |  |
| --- | --- | --- |
| Story Title | Story ID | Positive Comment Percentage |
| "London attack: Trump says US must get 'nasty' to fight terror, urges 'tougher' travel ban" | 709ina | 100% |
| "Former acting CIA chief Morell says Trump's intel disparagement will have \"significant effects\"" | 5msek3 | 100% |
| The 4 big Russia-Trump stories that flew under the radar this week | 70cjrk | 100% |
| Interactive Timeline: Everything We Know About Russia and President Trump | 7081jh | 100% |
| Donald Trump changes his view on Assad | 63p3vi | 95.00% |
| 'Positively Evil': Immigrant Checkpoints to Remain Open as Harvey Forces Evacuations | 6vzjne | 92.86% |
| "With Salvador decision, Trump's immigration policy veers into white nationalism" | 7p6ghj | 92.30% |
| Women’s March Organizer is a vicious Jew-hater with ties to Islamic Terror | 5pjz1b | 92.30% |
| Crowd somewhat diminished at second pro-Trump rally | 6jc56i | 92.30% |
| President Trump's campaign offers 'fake news' bumper stickers | 6pvuen | 92.30% |

To get the Top Ten Negative Stories, We ran the query:

Top Ten Negative Stories(Highest Percentage of Negative Comments about Trump)

|  |  |  |
| --- | --- | --- |
| Story Title | Story ID | Negative Comment Percentage |
| Trump hangs portrait of Andrew Jackson in Oval Office | 5q5h3j | 100% |
| Trump’s flashy executive actions could run aground | 5q7dcv | 100% |
| Kushner's Lawyer Replaced with Criminal Defense Attorney | 6ncvfe | 100% |
| Questions About Ivanka Trump’s Security Clearance Arise After 2016 Trump Tower Meeting | 6njqsp | 100% |
| "White House Math on Corporate Tax Cuts Is “Absolutely Crazy,” Experts Say" | 76yg71 | 100% |
| Alabama church compares Roy Moore to Jesus | 7jff6a | 100% |
| Biden: Trump’s attacks on Gillibrand ‘disgusting’ | 7jjf3p | 100% |
| Advocates of Puerto Rico Statehood Plan to Demand Representation | 7p97mu | 100% |
| Facebook admits to the Senate that it recommended Russian propaganda to some users | 7t0vxm | 100% |
| "Sarah Huckabee Sanders, Don’t You Dare Tell Nancy Pelosi to Smile" | 7ufadt | 100% |

Note: Before adding HAVING COUNT(\*) > 10, We would get many more 100% results as submissions that had only one positive or negative comment would always come out at 100% no matter what. Thus, we used a THRESHOLD of at least 10 responses so that there would be more chances of Percentages not being at 100.

5.

6. **Summary of our Findings:** Overall, /r/politics mostly thinks negatively on President Trump based on all the graphs that we generated. On the “President Trump Sentiment on /r/politics Over Time”, the Negative Percentage was consistently in the 90 percentile from 2017 - 2018 while Positive Percentage was consistently at 40%. On “Negative Trump Sentiment Across the US” the shades of RED were all darker than the shade of GREEN used for “Positive Trump Sentiment Across the US” - meaning that citizens felt more strongly negative than positive about Trump. On “%Positive - %Negative Trump Sentiment Across the US” we show that the darker the ‘blue’, the more the opinions on Trump differs. Through these three charts, we can clearly see that opinions on Trump do vary by state. We can see that in our dataset, the reddit users from Arkansas have the most positive percentage of trump sentiment while the reddit users from Indiana have the most negative percentage of trump sentiment. These results could prove to be inconclusive because the situation could be normal with the thresholds are set to 0.25 and 0.2. The professor also mentioned that this data was already highly biased based on majority of reddit users being young males that lean liberal. I predict that Higher Sentiment Score or Comment Score would mean a Negative Trump Sentiment since based on the users, a negative trump comment would have more people upbote it, while the opposite would be true for positive trump comments.

Question 1

Functional dependencies:

Input\_id --> labeldem

Input\_id --> labelgop

Input\_id --> labeldjt

Question 2

No, the *comments* relation does not look normalized because there are redundancies in the data. For example, the relation contains the *subreddit*, *subreddit\_id*, and *subreddit\_type* attributes. The *subreddit\_id* attribute functionally determines *subreddit* and *subreddit\_type*, and the comment’s *id* attribute functionally determines *subreddit\_id*:

id --> subreddit\_id

subreddit\_id --> subreddit

subreddit\_id --> subreddit\_type

Therefore, *subreddit* and *subreddit\_type* are redundant because we can use transitivity to determine these attributes using the comment *id*: *id* --> {*subreddit, subreddit\_type*}. There are other redundant attributes as well, such as *can\_gild*, *author\_flair\_text, author\_flair\_css\_class*, and *author\_cakeday,* which can each be functionally determined by *author* or {*author, subreddit\_id*}.

We can decompose the comments relation by taking a nontrivial functional dependency, such as *id --> subreddit\_id*, and splitting the relation into one relation containing {id, subreddit\_id} and a second relation containing *id* and every other attribute except *subreddit\_id*. We can repeat this process with other nontrivial functional dependencies until all relations are in BCNF or some other desired normal form.

We believe that the collector of the *comments* data stored the data this way for several reasons. First, having all data associated with a comment in one relation means that we do not need to compute joins across several tables, which are expensive operations. Also, the collector likely assumed that we will not have data integrity issues because we will not be adding more comment data to the relation in this project. Further, organizing all comment data into one table makes it easy for students to see what information is attached to each comment; it would be difficult for us to begin navigating this project if data were spread across many different tables.

Question 3

The join that I chose to attach explain was a join between the Resultant DataFrame of Predictions based on Comments{created\_utc, author\_flair\_text, link\_id, id, comment.score, pos, neg} and the DataFrame from Submissions{id, title, submission.score}. This join was as follows:

Between the Link ID (Pointing to the Submission the Comment Appears On) of the Resultant Comments and the Submissions ID to inner join the two tables. The output of the SQL Explain was:

Based on the output, it seems like Spark uses a BroadcastHashJoin, a very efficient join between large tables (<https://jaceklaskowski.gitbooks.io/mastering-spark-sql/spark-sql-joins-broadcast.html>). We can see the words “Inner” which I take to mean an inner join between link\_id#200 and id#97. We also see that for the Resultant DataFrame on Predictions based on Comments, Spark Projects the Attributes([created\_utc#198L, author\_flair\_text#199, link\_id#200, id#201, c\_score#202L, pos#203, neg#204]) and Makes Sure that the JOIN Key is NOT Null. This data is all gotten from the PARQUET which is fast for file scans (In memory file index!) The same procedure is made with the DataFrame from Submissions. Ultimately, a Broadcast Hash joins the two together.