**Data Analysis of Stack Exchange Website using GCP**

**Stage 1: Acquire the top 200,000 posts by view count from Stack Exchange. (Data Gathering)**

**Query 1**: select top 50000 \* from posts where posts.ViewCount > 100000 and posts.ViewCount < 500000 ORDER BY posts.ViewCount DESC;

**Query 2**: select top 50000 \* from posts where posts.ViewCount > 60000 and posts.ViewCount < 100000 ORDER BY posts.ViewCount DESC;

**Query 3**: select top 50000 \* from posts where posts.ViewCount > 44000 and posts.ViewCount < 60000 ORDER BY posts.ViewCount DESC;

**Query 4**: select top 50000 \* from posts where posts.ViewCount > 345000 and posts.ViewCount < 44000 ORDER BY posts.ViewCount DESC;

Now upload files in local home directory and move files on Hadoop using put command

aakashkhair95@cluster-a595-m: ~$ ls

Query\_Result\_1.csv Query\_Result\_2.csv Query\_Result\_3.csv Query\_Result\_4.csv

aakashkhair95@cluster-a595-m:~$ hadoop fs -put Query\_Result\_1.csv /

aakashkhair95@cluster-a595-m:~$ hadoop fs -put Query\_Result\_2.csv /

aakashkhair95@cluster-a595-m:~$ hadoop fs -put Query\_Result\_3.csv /

aakashkhair95@cluster-a595-m:~$ hadoop fs -put Query\_Result\_4.csv /

aakashkhair95@cluster-a595-m:~$ hadoop fs -ls /

aakashkhair95@cluster-a595-m:~$ cat Query\_Result\_1.csv Query\_Result\_2.csv Query\_Result\_3.csv Query\_Result\_4.csv > Final\_Query\_Result.csv

**Stage 2: Using Pig extract, transform and load the data as applicable**

* Cleaned the data like replaced the newlines with spaces

aakashkhair95@cluster-a595-m:~$sed':a;N;$!ba;s/\n//g' Final\_Query\_Result.csv > Final\_new.csv

aakashkhair95@cluster-a595-m:~$ hadoop fs -put Final\_new.csv /Final\_new

> pig – useHCatalog

> REGISTER /usr/lib/pig/piggybank.jar

> DEFINE CSVLoader org.apache.pig.piggybank.storage.CSVLoader();

* To load the data from Hadoop dfs to PIG:

>>stackdata=LOAD'/Final\_new'USING org.apache.pig.piggybank.storage.CSVExcelStorage (',','YES\_MULTILINE','NOCHANGE','SKIP\_INPUT\_HEADER') AS (id:int, posttypeid:int, acceptedanswerid:int, parentid:int, creationdate:chararray, deletiondate:chararray, score:int, viewcount:int, body:chararray, owneruserid:int, ownerdisplayname:chararray, lasteditoruserId:int, lasteditordisplayname:chararray, lasteditdate:chararray, lastactivitydate:chararray, title:chararray, tags:chararray, answercount:int, commentcount:int, favoritecount:int, closeddate:chararray, communityowneddate:chararray );

* To check the count of data. It should be 200000.

>> stackfull = GROUP stackdata ALL;

>>stackcount = FOREACH stackfull GENERATE COUNT(stackdata.id);

>>dump stackcount;

* Remove new lines form the body column and dump clean data:

>> cleaneddata FOREACH stackdata GENERATE id, score, viewcount, owneruserid, title, tags, (REPLACE(body,'[\r\n]+',' ')) AS body;

>> dump cleaneddata;

* Prerequisite step to load cleaned data from Pig to Hive we create database and table in Hive

hive> CREATE DATABASE user\_db;

hive> USE user\_db;

hive> CREATE TABLE user\_db.stackdata\_analysis (id int, score int, viewcount int, owneruserid int, title string, tags string,body string);

* Now to process the data in hive we have transferred the data to hive table from pig:

>>STORE cleaneddata INTO 'user\_db.stackdata\_analysis' USING org.apache.hive.hcatalog.pig.HCatStorer();

**Stage 3: Using Hive and/or MapReduce find the answers to following Questions.**

* Check the count to confirm the data Transferred properly:

hive> SELECT COUNT(id) FROM user\_db.stackdata\_analysis;

1)Top 10 posts by score

hive> SELECT id,score, viewcount, title FROM user\_db.stackdata\_analysis ORDER BY score DESC LIMIT 10;

2) The top 10 users by post score

hive> SELECT owneruserid AS USERID, SUM(score) AS SCORE FROM user\_db.stackdata\_analysis GROUP BY owneruserid having owneruserid is not null SORT BY score DESC LIMIT 10;

3) The number of distinct users, who used the word “Hadoop” in one of their posts

hive> SELECT COUNT(DISTINCT owneruserid) FROM user\_db.stackdata\_analysis WHERE UPPER(body) LIKE '%HADOOP%' OR UPPER(title) LIKE '%HADOOP%' OR LOWER(body) LIKE '%hadoop%' OR UPPER(tags) LIKE '%HADOOP%' OR LOWER(title) LIKE'%hadoop%' OR LOWER(tags) LIKE '%HADOOP%';

**Stage 4: Using MapReduce calculate the per-user TF-IDF (just submit the top 10 terms for each of the top 10 users from Query 3.II)**

* Loading the Data for cleaning in PIG:

>post\_data = LOAD '/Final\_new' USING org.apache.pig.piggybank.storage.CSVExcelStorage(',','YES\_MULTILINE','NOCHANGE','SKIP\_INPUT\_HEADER') AS (id:int, posttypeid:int, acceptedanswerid:int, parentid:int, creationdate:chararray, deletiondate:chararray, score:int, viewcount:int, body:chararray, owneruserid:int, ownerdisplayname:chararray, lasteditoruserId:int, lasteditordisplayname:chararray, lasteditdate:chararray, lastactivitydate:chararray, title:chararray, tags:chararray, answercount:int, commentcount:int, favoritecount:int, closeddate:chararray, communityowneddate:chararray );

* Removing all the symbols and cleaning the Body:

>post\_data = FOREACH post\_data GENERATE id, score, owneruserid, REPLACE(REPLACE(REPLACE(REPLACE(REPLACE((REPLACE(body,'[\r\n]+','')),'<[^>]\*>' , ' '),'[^a-zA-Z\\s\']+',' '),'(?=\\S\*[\'])([a-zA-Z\'-]+)',''),'(?<![\\w\\-])\\w(?![\\w\\-])',''),'[ ]{2,}',' ') AS body;

>post\_data = FILTER post\_data BY (owneruserid is not null);

* Grouping the data to perform TF IDF

>post\_data\_distinct\_users\_post = GROUP post\_data BY owneruserid;

>post\_data\_users\_by\_max\_score = FOREACH post\_data\_distinct\_users\_post GENERATE group AS userid, SUM(post\_data.score) AS maxscore;

>post\_data\_users\_by\_max\_score\_desc\_order = ORDER post\_data\_users\_by\_max\_score BY maxscore DESC;

>post\_data\_limit\_10 = LIMIT post\_data\_users\_by\_max\_score\_desc\_order 10;

>post\_data\_top\_10\_user\_id = FOREACH post\_data\_limit\_10 GENERATE userid;

>post\_data\_posts\_by\_10\_users = JOIN post\_data BY owneruserid, post\_data\_top\_10\_user\_id BY userid;

>post\_data\_posts\_by\_10\_users = FOREACH post\_data\_posts\_by\_10\_users GENERATE post\_data::owneruserid, LOWER(TRIM(REPLACE(post\_data::body,'[ ]{2,}',' '))) AS post\_data::body;

* Store the Result as a single CSV File

>STORE post\_data\_posts\_by\_10\_users INTO '/user/aakashkhair95/post\_data\_output' USING org.apache.pig.piggybank.storage.CSVExcelStorage(',','NO\_MULTILINE','NOCHANGE','SKIP\_OUTPUT\_HEADER');

aakashkhair95@cluster-57ba-m:~$ hadoop fs -ls /user/aakashkhair95/post\_data\_output

Found 2 items-rw-r--r-- 2 aakashkhair95 hadoop 0 2020-11-19 06:09 /user/aakashkhair95/post\_data\_output/\_SUCCESS-rw-r--r-- 2 aakashkhair95 hadoop 273621 2020-11-19 06:09 /user/aakashkhair95/post\_data\_output/part-r-00000

* Implementation of TFIDF in Hadoop using Python will be in four phases using three mappers and three reducers and fourth mapper is used to generate a single file with 10 users word list and its

TF-IDF value

hadoop jar /usr/lib/hadoop-mapreduce/hadoop-streaming.jar -file /home/aakashkhair95/mapper1.py /home/aakashkhair95/reducer1.py -mapper "python mapper1.py" -reducer "python reducer1.py" -input /user/aakashkhair95/post\_data\_output/part-r-00000 -output /user/aakashkhair95/output1

hadoop jar /usr/lib/hadoop-mapreduce/hadoop-streaming.jar -file /home/aakashkhair95/mapper2.py /home/aakashkhair95/reducer2.py -mapper "python mapper2.py" -reducer "python reducer2.py" -input /user/aakashkhair95/output1/part-0000\* -output /user/aakashkhair95/output2

hadoop jar /usr/lib/hadoop-mapreduce/hadoop-streaming.jar -file /home/aakashkhair95/mapper3.py /home/aakashkhair95/reducer3.py -mapper "python mapper3.py" -reducer "python reducer3.py" -input /user/aakashkhair95/output2/part-0000\* -output /user/aakashkhair95/output3

hadoop jar /usr/lib/hadoop-mapreduce/hadoop-streaming.jar -file /home/aakashkhair95/mapper4.py -mapper "python mapper4.py" -input /user/aakashkhair95/output3/part-0000\* -output /user/aakashkhair95/output4

* To find out the top 10 words by each user I loaded final output to - HIVE:

>hive CREATE TABLE stack\_post (word string, userId int, tfidf double) row format delimited fields terminated by '\t' stored as textfile;

>hive load data inpath '/user/aakashkhair95/output4/part-0000\*' into table stack\_post;

> hive SELECT \* FROM (SELECT userId,tfidf,word, rank() over (PARTITION BY userId ORDER BY tfidf DESC) as rank FROM stack\_post DISTRIBUTE BY userId SORT BY userId desc) a WHERE rank < 10 ORDER BY userId, rank;