

**Student details**

Name – SUDNYA SHASHIKANT MALI

Student ID – 21262233

Email – [sudnya.mali2@mail.dcu.ie](mailto:sudnya.mali2@mail.dcu.ie)

|  |
| --- |
| **I. Link for the Git repository:**  <https://github.com/sudnya07/CA675-Cloud-Tech-Assignment1.git> |

|  |
| --- |
| **II. Short description of the dataset**  Stack Exchange is a website which allows users to question and answer queries on varied fields with a rewarding process. The Data set contains user queries/questions, Usernames, UserID, Scores and tags used by users on their posts. |

|  |
| --- |
| **Steps taken to acquire the dataset:**  **Task 1: Get data from Stack Exchange (Data Acquisition/Collection)**  <https://data.stackexchange.com/stackoverflow/query/new> |
| **Step 1 – Collected Data by querying the stack exchange. (**There’s a limit on the number of rows that stack exchange returns i.e 50,000, Hence 4 queries were executed.)  **Query –**   1. select top 50000 \* from posts where viewcount <=(select max (viewcount)from posts)   order by viewcount desc   1. select top 50000 \* from posts where viewcount < 127754   order by viewcount desc   1. select top 50000 \* from posts where viewcount < 74785   order by viewcount desc   1. select top 50000 \* from posts where viewcount < 53347   order by viewcount desc  **Output:** 4 different .csv files were downloaded |

**Task 2: Load data into chosen cloud technology (MapReduce/Pig/Hive)**

|  |  |
| --- | --- |
| **Step 1 – Created GCP Cluster and uploaded the file on Cloud storage.**  **GCP Cluster Creation (Loading the data in Hive)**  GCP Console  Dataproc  Create Cluster  Enable Components (Hive WebCat,Jupyter Notebook and Zookeeper )  Create Cluster.  **Output:**      **Step 2: Stored the 4 dataset files downloaded from stack exchange on cloud storage.**    **Output:**    **Step 3: Cleaned the data using Python (jupyter notebook)**   |  | | --- | | 1. import pandas as pd 2. var = pd.read\_csv("/home/sudnya\_mali2/DS1.csv")    1. var.head() 3. var['Body']=var['Body'].str.replace(r'<[^<>]\*>',"",regex=True) //Removing html tags   var['Body']=var['Body'].str.replace(r'([^\w])'," ", regex=True) // Removing Punctuations  var['Body']=var['Body'].str.replace(r'\n'," ",regex=True) -p // Removing new Line  var['Body']=var['Body'].str.replace(r'\d+'," ",regex=True) // Removing Digits   1. var.to\_csv("/home/sudnya\_mali2/DS1\_clean.csv", index=False) |     **Output:** After cleaning the files. They were stored Successfully.      **Step 3. 2 – Merge the 4 files in 1 file** (Created Assignment1Dataset.csv)  **Output:**    **Step 4 : Create Table in Hive** (Created Table in hive by the name – Sudnya 007 )  **Query:**  CREATE TABLE IF NOT EXISTS sudnya007  (Postid int,  PostScores int,  Count int,  Stuc string,  UserId int,  UserName string,  Postscript string,  Label string)  ROW FORMAT SERDE 'org.apache.hadoop.hive.serde2.OpenCSVSerde';  **Output:** |
| **Step 4.2:Create View** (Created view in hive by the name – Sudnya007 to typecast few columns)  **Query:**  CREATE view IF NOT EXISTS Sudnyaview007 as select  Cast (Postid as int) as Postid,  Cast (PostScores as int) as PostScores,  Cast(Count as int) as Count,  Stuc ,  Cast (UserId as int) as UserId,  UserName,  Postscript ,  Label from sudnya007;  **Output:**    **Step 4.3 – Loaded the data in the hive table sudnya007.**  **Query:**  LOAD DATA INPATH 'gs://dataproc-staging-us-central1-289782612369-042x4b1s/combined.csv' INTO TABLE sudnya007;  **Output:** |

**Task 3: Query data using MapReduce/Pig/Hive**

|  |
| --- |
| **Task 3.1 - The top 10 posts by score**  **Query:**  SELECT Postid, Postscript, PostScore from sudnyaview007 ORDER BY PostScoress DESC LIMIT 10;  **Output:** |
| **Task 3.2 - The top 10 users by post score**  **Query:**  SELECT Userid,UserName,sum(PostScores) as PostScores from sudnyaview007 GROUP BY Userid,UserName ORDER BY PostScores DESC LIMIT 10;  **Output:** |
| **Task 3.3 - The number of distinct users, who used the word “cloud” in one of their posts**  **Query:**  SELECT COUNT(DISTINCT UserId) as TotalDistinctUsers FROM sudnyaview007 WHERE Postscript LIKE '% cloud %' OR Stuc LIKE '% cloud %';  **Output:** |

**Task 4: Calculate TF-IDF with MapReduce/Pig/Hive**

|  |
| --- |
| **Task 4** - **Calculate the per-user TF-IDF of the top 10 terms for each of the top 10 users**  **Query:**  from pyhive import hive #Installed necessary packages and libraries  from tabulate import tabulate  import pandas as pd  #Establishing a connection with Hive  conn = hive.Connection(host='localhost',port=10000,username='sudnya\_mali2',password='2589089972390042567',database='default',auth='CUSTOM')  cur = conn.cursor()  df = pd.read\_sql('''  SELECT UserId,UserName,Postscript,Stuc  from sudnyaview007  WHERE UserId  IN  (  select UserId from(select UserId,max(UserName),sum(postscores) as scores  from sudnyaview07  group by UserId  order by scores desc limit 10)stack  )  order by UserId''', conn)  display(df)  df["text"] = df["postscript"] + df["stuc"]  top\_10\_username = list(df["username"].unique())  top\_10\_username  #Calculating TF-IDF  from sklearn.feature\_extraction.text import TfidfVectorizer  # Calculate sum() of TF-IDF and get top 10 words with highest TF-IDF and select only those columns  # Calculate sum() of TF-IDF and get top 10 words with highest TF-IDF and select only those columns  def tf\_idf(df):  vectorizer = TfidfVectorizer(stop\_words='english', lowercase=True) # Remove Stop Words  response = vectorizer.fit\_transform(df["text"]) # Use title field for TF/IDF  df\_tfidf\_sklearn = pd.DataFrame(response.toarray(),columns=vectorizer.get\_feature\_names())  total\_tf\_idf = df\_tfidf\_sklearn.sum(axis = 0) # Remove sum of TF/IDF for getting top 10 most used words  top\_10\_list = total\_tf\_idf.nlargest(10) # Get top 10 words per user  top\_10\_words = list(top\_10\_list.index) # Get list of top 10 words  df\_tfidf\_sklearn[top\_10\_words] # Select only top 10 words as column  return df\_tfidf\_sklearn[top\_10\_words]  for each\_user in top\_10\_username:  username\_id = str(each\_user)  filtered\_data = df[(s\_df['username']==username\_id)] # Filter data only for selected user  tf\_idf\_df = tf\_idf(filtered\_data) # pass on to above function  print("For Username ID TF/IDF table : "+username\_id)  tf\_idf\_df.insert(0, 'usernameid', username\_id)# attach username ID field to dataframe  display(tf\_idf\_df)  print("\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\n\n\n")  **Output:**      **Output pdf file:** |

|  |
| --- |
| **III. Technologies used for each task – describe why I chose the technology for the task**   1. Google Cloud Platform – I tried executing the assignment on AWS platform but had issues with multinode configuration, hence executed the assignment on GCP as GCP cluster is simple to configure and also the defaults are easy. 2. Python- Panda’s data frame can store data in tabular format which made it easy for me to execute the queries successfully. Tried Pig to clean the data but was unsuccessful. 3. Hive – HiveQL is quite similar to SQL and also Hive was preinstalled in GCP unlike in AWS. Also, Hive does not require Java coding and makes the task easier by SQL Query implementation. |

**References:**

<https://learnpython.com/blog/python-libraries-data-cleaning/>

<https://www.guru99.com/hive-create-alter-drop-table.html>

<https://towardsdatascience.com/tf-idf-for-document-ranking-from-scratch-in-python-on-real-world-dataset-796d339a4089>

<https://stackoverflow.com/questions/21370431/how-to-access-hive-via-python>

<https://scikit-learn.org/stable/modules/generated/sklearn.feature_extraction.text.TfidfVectorizer.html>