Student Name	**Nitin Yadav**
Student Number	**16212304**
Programme	**MCM**
Project Title	**CA-675 Assignment 1**
Module Code	**CA-675**
Lecturer	**Dr. Alessandra Mileo**
Project Due	**21-Mar-2017**

Declaration

I declare that this material, which I now submit for assessment, is entirely my own work and has not been taken from the work of others, save and to the extent that such work has been cited and acknowledged within the text of my work. I understand that plagiarism, collusion, and copying is a grave and serious offence in the university and accept the penalties that would be imposed should I engage in plagiarism, collusion, or copying. I have read and understood the Assignment Regulations set out in the module documentation. I have identified and included the source of all facts, ideas, opinions, viewpoints of others in the assignment references. Direct quotations from books, journal articles, internet sources, module text, or any other source whatsoever are acknowledged and the source cited are identified in the assignment references.
I have not copied or paraphrased an extract of any length from any source without identifying the source and using quotation marks as appropriate. Any images, audio recordings, video or other materials have likewise been originated and produced by me or are fully acknowledged and identified.
This assignment, or any part of it, has not been previously submitted by me or any other person for assessment on this or any other course of study. I have read and understood the referencing guidelines found at http://www.library.dcu.ie/citing&refguide08.pdf and/or recommended in the assignment guidelines.
I understand that I may be required to discuss with the module lecturer/s the contents of this submission.

Date: 19 March, 2017

ASSIGNMENT 1

Acquiring Dataset

By the following queries top 250,000 posts by ViewCount were acquired from StackExchange:-

- SELECT count(*) FROM Posts;
    - >>34857917
- SELECT TOP 50000 count(*) FROM posts WHERE posts.ViewCount > 250000 ORDER BY posts.ViewCount;
    - >>4480
- SELECT count(*) FROM posts WHERE posts.ViewCount>12000;
    - >>341121
- SELECT count(*) FROM posts WHERE posts.ViewCount>12000 AND posts.ViewCount < 14000
    - >>53094
- SELECT count(*) FROM posts WHERE posts.ViewCount>14000 AND posts.ViewCount < 17000
    - >>56609
- SELECT count(*) FROM posts WHERE posts.ViewCount>17000 AND posts.ViewCount < 22000
    - >>60594
- SELECT count(*) FROM posts WHERE posts.ViewCount > 22000 AND posts.ViewCount < 33000
    - >>67323
- SELECT count(*) FROM posts WHERE posts.ViewCount>33000 AND posts.ViewCount < 77000
    - >>70768
- SELECT count(*) FROM posts WHERE posts.ViewCount > 77000 AND posts.ViewCount < 1000000
    - >>32465

These files were exported/downloaded as .csv files. Merged using google refine. The file was stored in Dropbox as pig was synced with Dropbox, so, importing the file in virtual machine was easy.

Query1

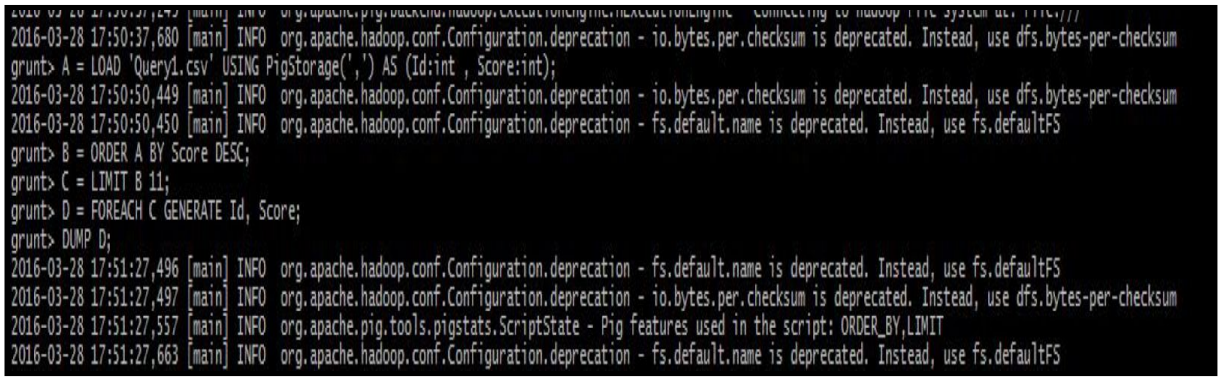
Following is the pig script for querying top ten posts by Score.

- A = LOAD 'Query1.csv' USING PigStorage(',') AS (Id:int, Score:int);

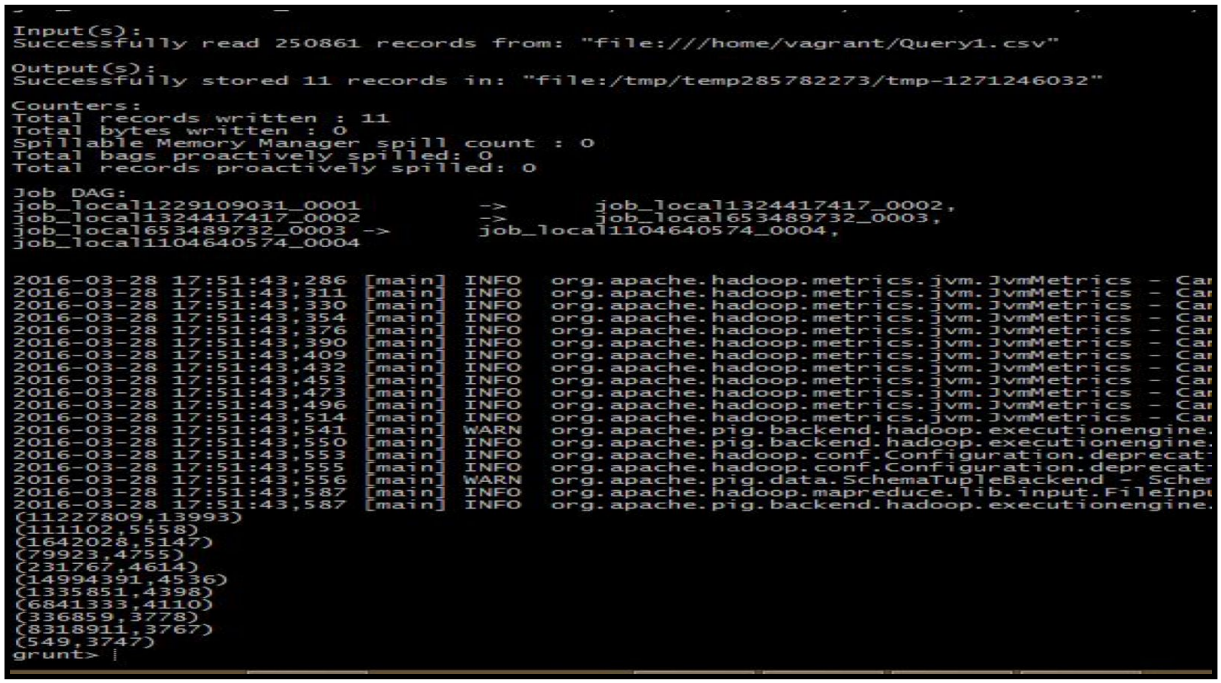
DUMP A;

- B = ORDER A BY score DESC;
- C = LIMIT B 11;
- D = FOREACH C GENERATE Id, Score;
- DUMP D;

Input :-



Output :-



Query2

Following is the pig script for querying top ten users by post score.

- A = LOAD ‘Query2.csv’ USING PigStorage(‘,’) AS (OwnerUserId:int , Score:int);
- B = GROUP A BY OwnerUserId;
- C = FOREACH B GENERATE group, SUM(A.Score) AS TOTALSCORE;
- D = ORDER C BY TOTALSCORE DESC;
- E = LIMIT D 11;
- DUMP E;

Input :-

```
grunt> A = LOAD 'Query2.csv' USING PigStorage(',') AS (OwnerUserId:int , Score:int);
2016-03-28 21:59:37,595 [main] INFO  org.apache.hadoop.conf.Configuration.deprecation - io.bytes.per.checksum is deprecated. Instead, use dfs.bytes-per-checksum
2016-03-28 21:59:37,596 [main] INFO  org.apache.hadoop.conf.Configuration.deprecation - fs.default.name is deprecated. Instead, use fs.defaultFS
grunt> B = GROUP A BY OwnerUserId;
grunt> C = FOREACH B GENERATE group, SUM(A.Score) AS TOTALSCORE;
grunt> D = ORDER C BY TOTALSCORE DESC;
grunt> E = LIMIT D 10;
grunt> DUMP E;
```

Output :-

The first column represents OwnerUserId and the second column represents corresponding aggregated Score of that User. Eliminating the 0, as they are null values changed into o. The rest shows the Top ten users by aggregated their post scores.

```
(0,184862)
(87234,20224)
(9951,14462)
(4883,13217)
(63051,8514)
(51816,8114)
(39677,8003)
(9021,7815)
(4639,7599)
(49153,7307)
grunt>
```

Query 3

Following is the pig script to get distinct users those used hadoop or hadoop related words in their posts body.

- A = LOAD 'Query3.csv' USING PigStorage(',') AS (OwnerUserId:int , Body:chararray);
- B = FOREACH A GENERATE FLATTEN(TOKENIZE((chararray)$1)) AS WORD, OwnerUserId;
- C = FILTER B BY WORD MATCHES '\\hadoop+' OR WORD MATCHES '\\Hadoop+' OR WORD MATCHES '\\ApacheHadoop+' OR WORD MATCHES '\\HADOOP+' OR WORD MATCHES '\\APACHEHADOOP+' OR WORD MATCHES '\\apache hadoop+' OR WORD MATCHES '\\Hadoop Apache+';
- D = group C by OwnerUserId;
- E = foreach D generate COUNT(C), C.WORD, group;
- DUMP E;

Input :-

```
grunt> A = LOAD 'Query3.csv' USING PigStorage(',') AS (OwnerUserId:int , Body:chararray);
2016-03-28 20:38:24,404 [main] INFO  org.apache.hadoop.conf.Configuration.deprecation - io.t
2016-03-28 20:38:24,405 [main] INFO  org.apache.hadoop.conf.Configuration.deprecation - fs.c
grunt> B = FOREACH A GENERATE FLATTEN(TOKENIZE((chararray)$1)) AS WORD , OwnerUserId;
```

Output shows distinct users post containing words such as Hadoop, Apache Hadoop, etc and it also shows the number of time the words have occurred and their order of occurrence.

Output :-

```
(2, {(HADOOP)},2064100)
(5, {(APACHEHADOOP)},5160250)
(7, {(ApacheHadoop)},7224350)
(7, {(hadoop)},7224350)
(6, {(Hadoop)},6192300)
(7, {(HADOOP)},7224350)
(1, {(APACHEHADOOP)},1032050)
(1, {(ApacheHadoop)},1032050)
(3, {(hadoop)},3096150)
(10, {(Hadoop)},10320500)
(9, {(HADOOP)},9288450)
(4, {(APACHEHADOOP)},4128200)
(1, {(ApacheHadoop)},1032050)
(10, {(hadoop)},10320500)
(5, {(Hadoop)},5160250)
(10, {(HADOOP)},10320500)
(4, {(APACHEHADOOP)},4128200)
(4, {(ApacheHadoop)},4128200)
(9, {(hadoop)},9288450)
(1, {(Hadoop)},1032050)
(9, {(HADOOP)},9288450)
(4, {(APACHEHADOOP)},4128200)
(5, {(ApacheHadoop)},5160250)
(2, {(hadoop)},2064100)
(8, {(Hadoop)},8256400)
(10, {(HADOOP)},10320500)
(1, {(APACHEHADOOP)},1032050)
(1, {(ApacheHadoop)},1032050)
(2, {(hadoop)},2064100)
(9, {(Hadoop)},9288450)
(8, {(HADOOP)},8256400)
(5, {(APACHEHADOOP)},5160250)
(10, {(ApacheHadoop)},10320500)
(10, {(hadoop)},10320500)
(8, {(Hadoop)},8256400)
(8, {(HADOOP)},8256400)
(1, {(APACHEHADOOP)},1032050)
(3, {(ApacheHadoop)},3096150)
(9, {(hadoop)},9288450)
(1, {(Hadoop)},1032050)
(5, {(HADOOP)},5160250)
(2, {(APACHEHADOOP)},2064100)
(10, {(ApacheHadoop)},10320500)
(2, {(hadoop)},2064100)
(1, {(Hadoop)},1032050)
(2, {(HADOOP)},2064100)
(6, {(APACHEHADOOP)},6192300)
(3, {(ApacheHadoop)},3096150)
(8, {(hadoop)},8256400)
(9, {(Hadoop)},9288450)
(3, {(HADOOP)},3096150)
(1, {(APACHEHADOOP)},1032050)
(5, {(ApacheHadoop)},5160250)
(2, {(hadoop)},2064100)
(10, {(Hadoop)},10320500)
(9, {(HADOOP)},9288450)
(7, {(APACHEHADOOP)},7224350)
(4, {(ApacheHadoop)},4128200)
(1, {(hadoop)},1032050)
(8, {(Hadoop)},8256400)
(6, {(HADOOP)},6192300)
(5, {(APACHEHADOOP)},5160250)
(6, {(ApacheHadoop)},6192300)
```

TF-IDF

- Tried reproducing with the above pig script by tokenizing each word without filtering any words. TF(t) was successfully found.
- By assuming Body of a post as document, was able to find count of tokenized words in the body and then grouped it by OwnerUserId.
- Top ten terms were generated using Countstar for all users.

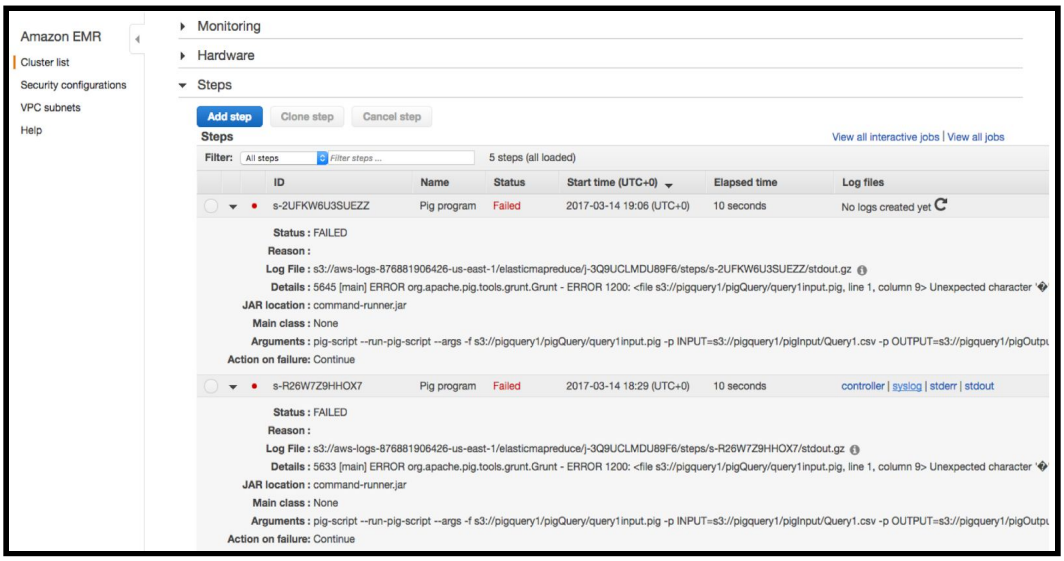
Input :-

Click here to view TF-IDF Source code

Output :-

Elastic MapReduce

- Created a S3 bucket named ‘pigquery1’.
- Created a folder called pigQuery inside which I stored my pig script.
- Created another folder named pigInput for pig input files and pigOutput folder for pig query output.
- Created an instance of EMR named Pig. This cluster was created with Hadoop debugging which had Hive and Pig pre-installed.
- Once the cluster started, it first executed the Hadoop debugging setup.
- Then, I added a step as follows:
    - Enter the below details in EMR Add step window.
        - Step type: Pig program
        - Name: Pig program
        - Script S3 location: s3://pigquery1/pigQuery/query1input.pig
        - Input S3 location: s3://pigquery1/pigInput/Query1.csv
        - Output S3 location: s3://pigquery1/pigOutput/
    - Leave the arguments blank and Action on failure Continue.
    - Click on Add.
- **Following was the outcome of this Job.**



Conclusion

- Pig is used to query the required questions for the Assignment.
- Could not get HIVE environment setted up successfully, had to do querying and transformations with Pig -x Mapreduce.

Click here to view HIVE source code for Query1
Click here to view HIVE source code for Query2
Click here to view HIVE source code for Query3

- Used pig script commands for neglecting newlines in body of posts.
- Also there were some valued missing in OwnerUserId columns which were converted to 0; As Pig was reading the next input as current value.
- TF-IDF partial output.
- Learnt pig and HIVE scripts.

APPENDIX

- Github Repository : https://github.com/NitinYadav20/Cloud-Technology
  This link contains all the source code, dataset and screenshots.