CA 675 Cloud Technologies Assignment 1

Conall Butler

Student Number:21269599

conall.butler36@mail.dcu.ie

Git repository: https://github.com/ConallButler/CA675-Assignment-1

Task 1; The Dataset

The acquired dataset used in subsequent tasks consists for 5 CSV files of total 200,000 entries excluding headers. Schema provided by stack exchange[1].

The dataset was acquired using the Stack Exchange Data Explorer[2]. See queries and number of entries yielded are show below. Each row corresponds to a post on the Stack Exchange forum, with each cell containing some piece of metadata, the Title, or the Body of the post. The body of posts appear as HTML in the dataset.

```
select top 50000 * from posts where posts.ViewCount >140000 ORDER BY posts.ViewCount;

select top 50000 * from posts where posts.ViewCount<140001 and posts.ViewCount >80000 ORDER BY posts.ViewCount;

select top 50000 * from posts where posts.ViewCount<80001 and posts.ViewCount >58000 ORDER BY posts.ViewCount;

select top 50000 * from posts where posts.ViewCount<58001 and posts.ViewCount >45000 ORDER BY posts.ViewCount;

select top 18724 * from posts where posts.ViewCount<45001 and posts.ViewCount >41000 ORDER BY posts.ViewCount DESC;
```

Query	Range	Posts
1	>14000	43628
2	>80000, <140001	47772
3	>58000, <80001	43732
4	>45000, <58001	46144
5	>41000, <45001	18724

ViewCount ranges in intervals approaching 50000 posts were determined using the method laid out in the assignment document Data Acquisition section. 18724 posts in descending order were retrieved in query 5 as this was the remaining number required to reach 200,000 posts after summing the first 4 queries.

Output CSVS; https://github.com/ConallButler/CA675-Assignment-1/tree/main/CSVs

Task 2&3

Pig was selected for initial ETL tasks as dealing with line-breaks contained in data can be difficult using Hive.

2.2.1, 2.2.2, and 2.2.3 were identified as tasks that could be easily completed using SQL queries; Hive was selected as such

--Load CSVs output by Stack Exchange Queries using schema adjusted for Pig datatypes[3]

```
Query1 = LOAD 'CA675-Assignment-1/csvs/query1.csv' USING
org.apache.pig.piggybank.storage.CSVExcelStorage(',', 'YES_MULTILINE', 'WINDOWS', 'SKIP_INPUT_HEADER')
AS (Appendix 3)
```

--Filter by required fields to reduce resources required, clean line-breaks from Body to allow easy processing in Hive

```
Queries = UNION Query1, Query2, Query3, Query4, Query5;
Queries_Filtered_Cleaned1 = FOREACH Queries GENERATE Id, Score, OwnerUserId, Title, REPLACE(Body,'\n', '')
AS Body;
```

--Clean HTML tags so as not to interfere with counting of terms in 2.2.3 and 2.3

```
Queries_Filtered_Cleaned2 = FOREACH Queries_Filtered_Cleaned1 GENERATE Id, Score, OwnerUserId, Title,
REPLACE(Body,'<.*?.>', '') AS Body;
```

--Store filtered, cleaned data for use in later tasks.

```
STORE Queries_Filtered_Cleaned2 INTO 'CA675-Assignment-1/csvs/Queries_Filtered_Cleaned' USING
org.apache.pig.piggybank.storage.CSVExcelStorage(',' ,'NO_MULTILINE' ,'UNIX', 'SKIP_OUTPUT_HEADER');
```

--Hive was used to complete the remainder of Task 2&3, data loaded as below

```
CREATE DATABASE sequeries;
USE sequeries;
CREATE EXTERNAL TABLE Queries_Filtered_Cleaned
(Id int, Score int, OwnerUserId int, Title string, Body string)
ROW FORMAT DELIMITED
FIELDS TERMINATED BY ','
LOCATION 'CA675-Assignment-1/csvs/Queries_Filtered_Cleaned';
```

Screenshots

https://github.com/ConallButler/CA675-Assignment-1/blob/main/Screenshots/2.%20PIG%20ETL%201.png
https://github.com/ConallButler/CA675-Assignment-1/blob/main/Screenshots/3.%20Pig%20ETL%202.png
https://github.com/ConallButler/CA675-Assignment-1/blob/main/Screenshots/3.1%20PIG%20ETL%20terminal%20info

Task 2.2.1

--Display Id, Score, Owneruserid and Title for top 10 posts, by score.

```
SELECT id,score,owneruserid,title  FROM queries_filtered_cleaned SORT BY Score DESC LIMIT 10;
```

```
Total MapReduce CPU Time Spent: 11 seconds 150 msec
OK
id      score   owneruserid     title
11227809        25903   87234   Why is processing a sorted array faster than processing an unsorted array?
927358  23303   89904   How do I undo the most recent local commits in Git?
2003505 18475   95592   How do I delete a Git branch locally and remotely?
292357  12812   6068    What is the difference between 'git pull' and 'git fetch'?
231767  11528   18300   "What does the ""yield"" keyword do?"
477816  10902   12870   What is the correct JSON content type?
348170  10062   14069   How do I undo 'git add' before commit?
5767325 9899    364969  How can I remove a specific item from an array?
6591213 9764    338204  How do I rename a local Git branch?
1642028 9545    87234   "What is the ""-->"" operator in C/C++?"
Time taken: 42.662 seconds, Fetched: 10 row(s)
hive>
```

Screenshot;

https://github.com/ConallButler/CA675-Assignment-1/blob/main/Screenshots/5.%20hive%202.2.1.png

Task 2.2.2

--Sum score for each distinct user, display top 10 by summed score. Null OwnerUserId values ignored, these correspond to blanks in the source data.

```
SELECT owneruserId, SUM(score) AS totalScore
FROM queries_filtered_cleaned
WHERE owneruserId IS NOT NULL
GROUP BY owneruserId
ORDER BY totalScore DESC LIMIT 10;
```

```
Ended Job = job_1635188457710_0036
MapReduce Jobs Launched:
Stage-Stage-1: Map: 1  Reduce: 1   Cumulative CPU: 11.37 sec   HDFS Read: 187942450 HDFS Write: 2656301 SUCCESS
Stage-Stage-2: Map: 1  Reduce: 1   Cumulative CPU: 6.14 sec   HDFS Read: 2664018 HDFS Write: 325 SUCCESS
Total MapReduce CPU Time Spent: 17 seconds 510 msec
OK
owneruserid     totalscore
87234   37624
4883    28779
9951    26764
6068    25889
89904   23978
51816   23680
49153   20183
179736  19483
95592   19440
63051   19316
Time taken: 44.964 seconds, Fetched: 10 row(s)
```

Screenshots

https://github.com/ConallButler/CA675-Assignment-1/blob/main/Screenshots/6.%20hive%202.2.2(1).png

https://github.com/ConallButler/CA675-Assignment-1/blob/main/Screenshots/7.%20Hive%202.2.2(2).png

Task 2.2.3

--Count distinct users with “Cloud” or “cloud” in the body of one of their posts.

```
SELECT COUNT (DISTINCT owneruserid) AS (Users_that_use_the_word_cloud)
FROM queries_filtered_cleaned
WHERE body RLIKE '(cloud|Cloud)';
```

```
hive> SELECT COUNT (DISTINCT owneruserid) AS (Users_that_use_the_word_cloud)
    > FROM queries_filtered_cleaned
    > WHERE body RLIKE '(cloud|Cloud)';
Query ID = conall_20211026134038_20833842-aaea-4615-8396-9e434b81c5ae
Total jobs = 1
Launching Job 1 out of 1
Number of reduce tasks determined at compile time: 1
In order to change the average load for a reducer (in bytes):
  set hive.exec.reducers.bytes.per.reducer=<number>
In order to limit the maximum number of reducers:
  set hive.exec.reducers.max=<number>
In order to set a constant number of reducers:
  set mapreduce.job.reduces=<number>
Starting Job = job_1635188457710_0037, Tracking URL = http://conall-NS14A8:8088/proxy/application_1635188457710_0037/
Kill Command = /usr/local/hadoop/bin/mapred job  -kill job_1635188457710_0037
Hadoop job information for Stage-1: number of mappers: 1; number of reducers: 1
2021-10-26 13:40:44,837 Stage-1 map = 0%,  reduce = 0%
2021-10-26 13:40:52,051 Stage-1 map = 100%,  reduce = 0%, Cumulative CPU 6.09 sec
2021-10-26 13:40:57,191 Stage-1 map = 100%,  reduce = 100%, Cumulative CPU 8.88 sec
MapReduce Total cumulative CPU time: 8 seconds 880 msec
Ended Job = job_1635188457710_0037
MapReduce Jobs Launched:
Stage-Stage-1: Map: 1  Reduce: 1   Cumulative CPU: 8.88 sec   HDFS Read: 187939430 HDFS Write: 103 SUCCESS
Total MapReduce CPU Time Spent: 8 seconds 880 msec
OK
users_that_use_the_word_cloud
325
Time taken: 19.628 seconds, Fetched: 1 row(s)
hive>
```

Screenshot

https://github.com/ConallButler/CA675-Assignment-1/blob/main/Screenshots/8.%20Hive%202.2.3.png

Hive Outputs Spreadsheet format

https://github.com/ConallButler/CA675-Assignment-1/blob/main/Outputs/Hive%20Outputs.ods

Hive Queries Source Code

https://github.com/ConallButler/CA675-Assignment-1/blob/main/Source%20Code/4.%20Hive%20Queries

Task 4

Apache Pig was selected for task 4 due to ease of development in Pig Latin, and lack of familiarity with java, required for Hadoop/MadReduce.

Pg33 was referenced for the outline of TF-IDF Pig Latin script
https://courses.cs.ut.ee/MTAT.08.036/2017_fall/uploads/Main/L4_Pig_2017.pdf

Final outputs were 10 tab delimited text files of Schema (Term, TF-IDF), one for each user. Top 10 were selected and combined in excel (top 10 can be done in Pig using LIMIT 10, all terms generated for completeness);

Pig Text Outputs
https://github.com/ConallButler/CA675-Assignment-1/tree/main/Outputs/HDFS%20files/CA675-Assignment-1/csvs

Output xlsx
https://github.com/ConallButler/CA675-Assignment-1/blob/main/Outputs/Top10Users%20TF-IDF.xlsx

Source Code
https://github.com/ConallButler/CA675-Assignment-1/blob/main/Source%20Code/5.%20Pig%20TF-IDF

Screenshots
https://github.com/ConallButler/CA675-Assignment-1/blob/main/Screenshots/9.%20TFIDF%201.png

https://github.com/ConallButler/CA675-Assignment-1/blob/main/Screenshots/10.%20TFIDF%202.png

Appenidx 1. Stack Exchange Schema

(Id:int, PostType:tinyint, AcceptedAnswerId:int, ParentId:int, CreationDate:datetime, DeletionDate:datetime, Score:int, ViewCount:int, Body: nvarchar (max), OwnerUserId:int, OwnerDisplayName:nvarchar (40), LastEditorUserId:int, LastEditorDisplayName:nvarchar (40), LastEditDate:datetime, LastActivityDate:datetime, Title:nvarchar (250), Tags:nvarchar (250), AnswerCount:int, CommentCount:int, FavoriteCount:int, ClosedDate:datetime, CommunityOwnedDate:datetime, ContentLicense:varchar (12))

Appenidx 2. Stack Exchange Query

https://data.stackexchange.com/stackoverflow/query/new

Appenidx 3. Pig Schema

(Id:int, PostTypeId:int, AcceptedAnswerId:int, ParentId:int, CreationDate:datetime, DeletionDate:datetime, Score:int, ViewCount:int, Body: chararray, OwnerUserId:int, OwnerDisplayName:chararray, LastEditorUserId:int, LastEditorDisplayName:chararray, LastEditDate:datetime, LastActivityDate:datetime, Title:chararray, Tags:chararray, AnswerCount:int, CommentCount:int, FavoriteCount:int, ClosedDate:datetime, CommunityOwnedDate:datetime, ContentLicense:chararray)

Appenidx 4. Screenshots in chronological order https://github.com/ConallButler/CA675-Assignment-1/tree/main/Screenshots

Appenidx 5. Source Code https://github.com/ConallButler/CA675-Assignment-1/tree/main/Source%20Code

Appenidx 6. HDFS File Content https://github.com/ConallButler/CA675-Assignment-1/tree/main/Outputs/HDFS%20files/CA675-Assignment-1/csv