

CA675 - Assignment 1

Minh Nhat Vu
DCU student ID: 20214265
Email: minh.vu3@mail.dcu.ie

Link Github: <https://github.com/NhatVu/DCU-CA675-Assignment1>

Task 1

Dataset is obtained from <https://data.stackexchange.com/>. Because of the restriction, the site is only allowed to get 50000 records per query. To get enough 200,000 records, the query must be done multiple times.

The first query will count the number of results to records. When it nearly reaches 50000, I will retrieve the result.

Sample counting query:

```
select count(*)  
from posts  
where posts.ViewCount <= 80000 and posts.ViewCount > 57000;
```

Then, using the query below to get the data

```
select *  
from posts  
where posts.ViewCount <= 80000 and posts.ViewCount > 57000;
```

The last query will exceed 200,000 a bit. I use pandas to remove the abundance. For my dataset, I divide ViewCount into chunks that list below

```
+Ifn, 128k, 80k, 57k, 44k, 41k
```

Preprocessing data:

Because data contains multiple lines or commas in a column. This leads to errors when importing data to the Hive table. We need to remove all lines in columns, then use OpenCSVSerde to parse CSV files correctly. But the disadvantage when using OpenCSVSerde is it converts all columns to the String field.

Check file *Task1_preprocess.py* to know more about code for removing line breaks. One important note is that different OSs use different line break characters.

Dataset description:

There are many fields for this dataset. For the sake of this assignment, I will care about some relevant fields. It includes OwnerUserId, Score, Title, Body

Otherwise, there are many interesting fields like ViewCount, PostTypeId (the question is 1 and answer is 2), tags, answer count, comment count

Note: OwnerUserId is empty because user has been disabled account.

Task 2 & 3

In this task, I will use Hive. These tasks look like ad hoc queries. Furthermore, it's easier when processing structured data with HiveQL language.

Link code: <https://github.com/NhatVu/DCU-CA675-Assignment1/blob/main/code/Task2%263.sql>

2.2.1. The top 10 posts by score

Step descriptions: sort score by desc, then select posts based on that sorted order.

Query:

```
select id, cast(score as int), title
from posts
order by cast(score as int) desc
limit 10;
```

2.2.2. The top 10 users by total post score

Step description: First, group by OwnerUserId, then calculate the sum of the score for each group. Next, sort by total score. Finally, query based on that sorted order.

Query:

```
select OwnerUserId, sum(cast(score as int)) as s
from posts
where OwnerUserId != ""
group by OwnerUserId
order by s desc
limit 10;
```

2.2.3. The number of distinct users, who used the word "cloud" in one of their Posts

Task description:

This required finding the word “cloud” in both title and body. The original form looks like (OwnerUserId, title + body).

Then, I split title + body into word by space and comma. I have another table like (OwnerUserId, word).

Finally, I count distinct OwnerUserId who has the word “cloud”.

Query:

```
select count(distinct(OwnerUserId))
from posts
lateral view explode(split(concat(title, " ", body), ' |,')) lateralTable as
word
where word = "cloud";
```

Task 4

Link Github: <https://github.com/NhatVu/DCU-CA675-Assignment1/blob/main/code/Task4.sql>

First, we need to calculate TF-IDF. Input has form: (OwnerUserId, title + body)

We need to split title and body to word to achieve this schema: (OwnerUserId, word)

```
create or replace view flatten_word
as
select cast(OwnerUserId as int) as OwnerUserId, trim(word) as word
from posts
lateral view explode(split(concat(title, " ", body), ' |,')) lateralTable
as word
where word != "";
```

Next, we calculate TF and save it to view tf. For simplicity, I choose log normalization. I will create tf view like (OwnerUserId, word, termFreq)

```
create or replace view tf
as
select OwnerUserId, word, log(10, count(*) + 1) as termFreq
from flatten_word
group by OwnerUserId, word;
```

Then, I calculate IDF and produce schema: (word, idf)

```
create or replace view idf
as
```

```

select word, numberAppearInUser, numberUser, log(10,
numberUser/(numberAppearInUser + 1)) + 1 as idf
from (
select word, count(distinct(OwnerUserId)) as numberAppearInUser
from flatten_word
group by word) as a
cross join (
select count(distinct (OwnerUserId)) as numberUser
from flatten_word) as b;

```

Finally, I combine the tf view and the idf view to have the final result. Because calculating tf idf cost a lot of time, I created a tf idf table to save this result.

```

create external table if not exists tfidf (
  OwnerUserId int,
  word String,
  tfidf double
);

```

```

INSERT OVERWRITE table tfidf
select OwnerUserId, tf.word, tf.termFreq * idf.idf as tfidf
from tf
join idf
on tf.word = idf.word;

```

For getting the top 10 words for each user in the top 10, we use this query. It contains several steps. First, I will filter out all rows that do not belong to the top 10 users. Then, I partition the dataset by OwnerUserId and apply the rank function on it, with descending order by tfidf score. Finally, filter row with rank less than 10.

```

select * from (
select OwnerUserId, word, tfidf, rank() over(partition by OwnerUserId order
by tfidf desc) as rn
from tfidf as T
where T.OwnerUserId in (
select cast(OwnerUserId as int) from (
select OwnerUserId, sum(cast(score as int)) as s
from posts
where OwnerUserId != ""
group by OwnerUserId
order by s desc
limit 10

```

```
) as B  
)  
) as A  
where A.rn <= 10;
```

Appendix

This section provides screenshots for each task above.

Screenshot task 1

Upload posts dataset to cloud. Then using count the number of records

The screenshot shows the Google Cloud Platform interface. The top navigation bar includes 'Google Cloud Platform', 'My First Project', and a search bar. The left sidebar shows 'Cloud Storage' selected. The main content area displays 'Bucket details' for a bucket named 'posts' in the 'trusty-diorama-326917-warehouse' project. The bucket is located in 'europe-west2 (London)' and is 'Standard' storage class. Below the bucket details, there are tabs for 'OBJECTS', 'CONFIGURATION', 'PERMISSIONS', 'PROTECTION', and 'LIFECYCLE'. The 'OBJECTS' tab is active, showing a list of objects. The objects are CSV files with names like 'QueryResults-128k-49479-clean.csv', 'QueryResults-41k-15495-clean-cut.', 'QueryResults-44k-47793-clean.csv', 'QueryResults-57k-46579-clean.csv', and 'QueryResults-80k-41921-clean.csv'. Each object has a size, type, creation date, storage class, last modified date, and a public status. Below the object list, there is a 'Filter' section and a 'Show deleted data' toggle. At the bottom, there is a 'CLOUD SHELL' terminal window. The terminal shows the following commands and output:

```
0: jdbc:hive2://localhost:10000> select count(id) from posts;
+-----+
| _c0 |
+-----+
| 200000 |
+-----+
1 row selected (0.972 seconds)
0: jdbc:hive2://localhost:10000> select count(id) from posts;
+-----+
| _c0 |
+-----+
| 200000 |
+-----+
1 row selected (47.442 seconds)
0: jdbc:hive2://localhost:10000>
```

Screenshot task 2.2.1

The screenshot shows the Google Cloud Platform terminal window. The terminal displays the following SQL query and its results:

```
0: jdbc:hive2://localhost:10000> select id, cast(score as int), title
. . . . .> from posts
. . . . .> order by cast(score as int) desc
. . . . .> limit 10;
+-----+-----+-----+
| id | score | title |
+-----+-----+-----+
| 11227809 | 25893 | Why is processing a sorted array faster than processing an unsorted array? |
| 927358 | 23274 | How do I undo the most recent local commits in Git? |
| 2003505 | 18451 | How do I delete a Git branch locally and remotely? |
| 292357 | 12796 | What is the difference between 'git pull' and 'git fetch'? |
| 231767 | 11512 | What does the "yield" keyword do? |
| 477816 | 10894 | What is the correct JSON content type? |
| 348170 | 10045 | How do I undo 'git add' before commit? |
| 5767325 | 9877 | How can I remove a specific item from an array? |
| 6591213 | 9747 | How do I rename a local Git branch? |
| 1642028 | 9539 | What is the "-->" operator in C/C++? |
+-----+-----+-----+
10 rows selected (23.174 seconds)
```

Screenshot task 2.2.2

```
CLOUD SHELL
Terminal (trusty-diorama-326917) x (trusty-diorama-326917) + - Open editor

-----+-----+
10 rows selected (23.174 seconds)
0: jdbc:hive2://localhost:10000> select OwnerUserId, sum(cast(score as int)) as s
. . . . .> from posts
. . . . .> where OwnerUserId != ""
. . . . .> group by OwnerUserId
. . . . .> order by s desc
. . . . .> limit 10;
-----+-----+
| owneruserid | s |
-----+-----+
| 87234.0 | 37606 |
| 4883.0 | 28739 |
| 9951.0 | 26728 |
| 6068.0 | 25860 |
| 89904.0 | 23949 |
| 51816.0 | 23428 |
| 49153.0 | 20156 |
| 179736.0 | 19454 |
| 95592.0 | 19413 |
| 63051.0 | 19295 |
-----+-----+
10 rows selected (26.937 seconds)
```

Screenshot task 2.2.3

```
0: jdbc:hive2://localhost:10000> select count(distinct(OwnerUserId))
. . . . .> from posts
. . . . .> lateral view explode(split(concat(title," ", body), ' |, ')) lateralTable as word
. . . . .> where word = "cloud";
-----+-----+
| _c0 |
-----+-----+
| 112 |
-----+-----+
1 row selected (42.987 seconds)
0: jdbc:hive2://localhost:10000>
```

Screenshot task 4

Save the tfidf table on Hadoop.

Google Cloud Platform My First Project Search products and resources

Cloud Storage

Browser Monitoring Settings

Bucket details

trusty-diorama-326917-warehouse

Location: europe-west2 (London) Storage class: Standard Public access: Subject to object ACLs Protection: None

OBJECTS CONFIGURATION PERMISSIONS PROTECTION LIFECYCLE

Buckets > trustworthy-diorama-326917-warehouse > datasets > tfidf

UPLOAD FILES UPLOAD FOLDER CREATE FOLDER MANAGE HOLDS DOWNLOAD DELETE

Filter by name prefix only Filter Filter objects and folders Show deleted data

Name	Size	Type	Created	Storage class	Last modified	Put
000000_0	146.3 MB	application/octet-stream	22 Oct 20...	Standard	22 Oct 202...	No
000001_0	164.4 MB	application/octet-stream	22 Oct 20...	Standard	22 Oct 202...	No
000002_0	165.4 MB	application/octet-stream	22 Oct 20...	Standard	22 Oct 202...	No

Result for getting top 10 words for each top 10 OwnerUserId

CLOUD SHELL
Terminal (trusty-diorama-326917) x + ▾ [Open editor](#)

```
.....> ) as A
.....> where A.rn <= 10;
```

a.owneruserid	a.word	a.tfidf	a.rn
9951	NAS	3.055185426732032	1
9951	url	2.810818144584617	2
9951	closures	2.805346576451386	3
9951	<code>{%	2.7954785626232748	4
9951	ancestor_id	2.7546706125402265	5
9951	l);\$ext	2.7546706125402265	5
9951	<code>u'</code>	2.7546706125402265	5
9951	metaclasses	2.670653730074415	8
9951	'keep	2.670653730074415	8
9951	my_dict	2.49508418456695	10
51816	<code>Layers</code>	4.879193103460541	1
51816	STEP	4.582778140098048	2
51816	(4.154124505352929	3
51816	Point2	4.035519505671264	4
51816	Python?	4.027103346480242	5
51816	Equals	3.999114186348142	6
51816)	3.7311865976651104	7
51816	obj	3.6819455251224817	8
51816	1e-9;float	3.476007301481081	9
51816	vector3	3.294769184902168	10
63051	use:</p><pre><code>cmd	3.476007301481081	1
63051	pts/1	3.055185426732032	2
63051	<code>cut</code>	2.9672075828388196	3
63051	Bash	2.78456141598918	4