PROJECT 2: INSTAGRAM USER ANALYTICS

SQL FUNDAMENTALS

NAME: K. Yaswanth
GMAIL: kunayaswanth123@gmail.com

To provide best report for the Marketing team and Investor Metrics using MySQL

SUBMITTING TO TRAINITY :)

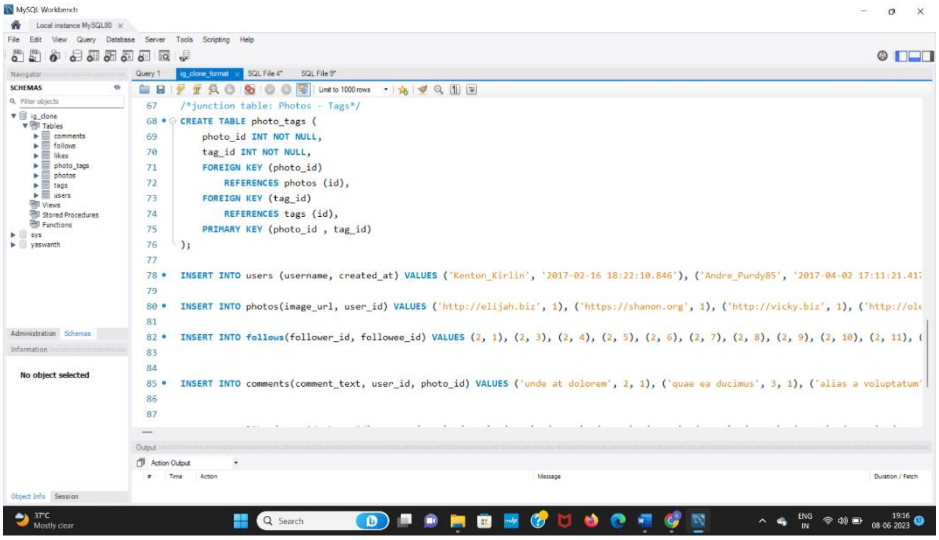
SPECIAL THANKS TO TRAINITY TEAM

Project Description:- To help the marketing team and investor metrics.

1. Here, we have ig_clone database and it has comments, likes, users, photo_tags, tags, photos are tables. We do operations on this tables.
2. Doing correct analysis by giving correct suggestions for the questions they asked using MySQL.
3. By querying the data of marketing team and by giving correct suggestions to the team helps the Instagram grow.
4. Here, we find most loyal users, inactive users, declaring contest winner, hashtag researching, launching ad campaigning questions
5. And finding the query solutions of investor metrics i.e. user engagement and finding Bots and fake accounts in users table.

Approach:-

1. The dataset provided in Instagram User Analytics in project 2. It is copied to MySQL workbench and execute each query.



```
/*junction table: Photos - Tags*/
CREATE TABLE photo_tags (
    photo_id INT NOT NULL,
    tag_id INT NOT NULL,
    FOREIGN KEY (photo_id)
        REFERENCES photos (id),
    FOREIGN KEY (tag_id)
        REFERENCES tags (id),
    PRIMARY KEY (photo_id , tag_id)
);

INSERT INTO users (username, created_at) VALUES ('Kenton_Kirlin', '2017-02-16 18:22:10.846'), ('Andre_Purdy85', '2017-04-02 17:11:21.41

INSERT INTO photos(image_url, user_id) VALUES ('http://elijah.biz', 1), ('https://shanon.org', 1), ('http://vicky.biz', 1), ('http://ol

INSERT INTO follows(follower_id, followee_id) VALUES (2, 1), (2, 3), (2, 4), (2, 5), (2, 6), (2, 7), (2, 8), (2, 9), (2, 10), (2, 11), (


INSERT INTO comments(comment_text, user_id, photo_id) VALUES ('unde at dolorem', 2, 1), ('quae ea ducimus', 3, 1), ('alias a voluptatum'
```

2. We use required tables based on our requirement like using joins, subqueries
3. With the help of sql fundamentals and knowledge, we solve the problems which were given in the project.

Tech-Stack Used:- MySQL workbench Version 8.0.33

Insights:-

1. While, doing this project. I gained practical experience through by applying joins and writing sub queries
2. Gained pratical experience, by writing common term expressions.
3. Learnt practically, how to make the report very good.
4. Learnt dayname() function, limit keyword and their usage through this project.
5. Learnt to think deeply.

Result : It is written at the EOF

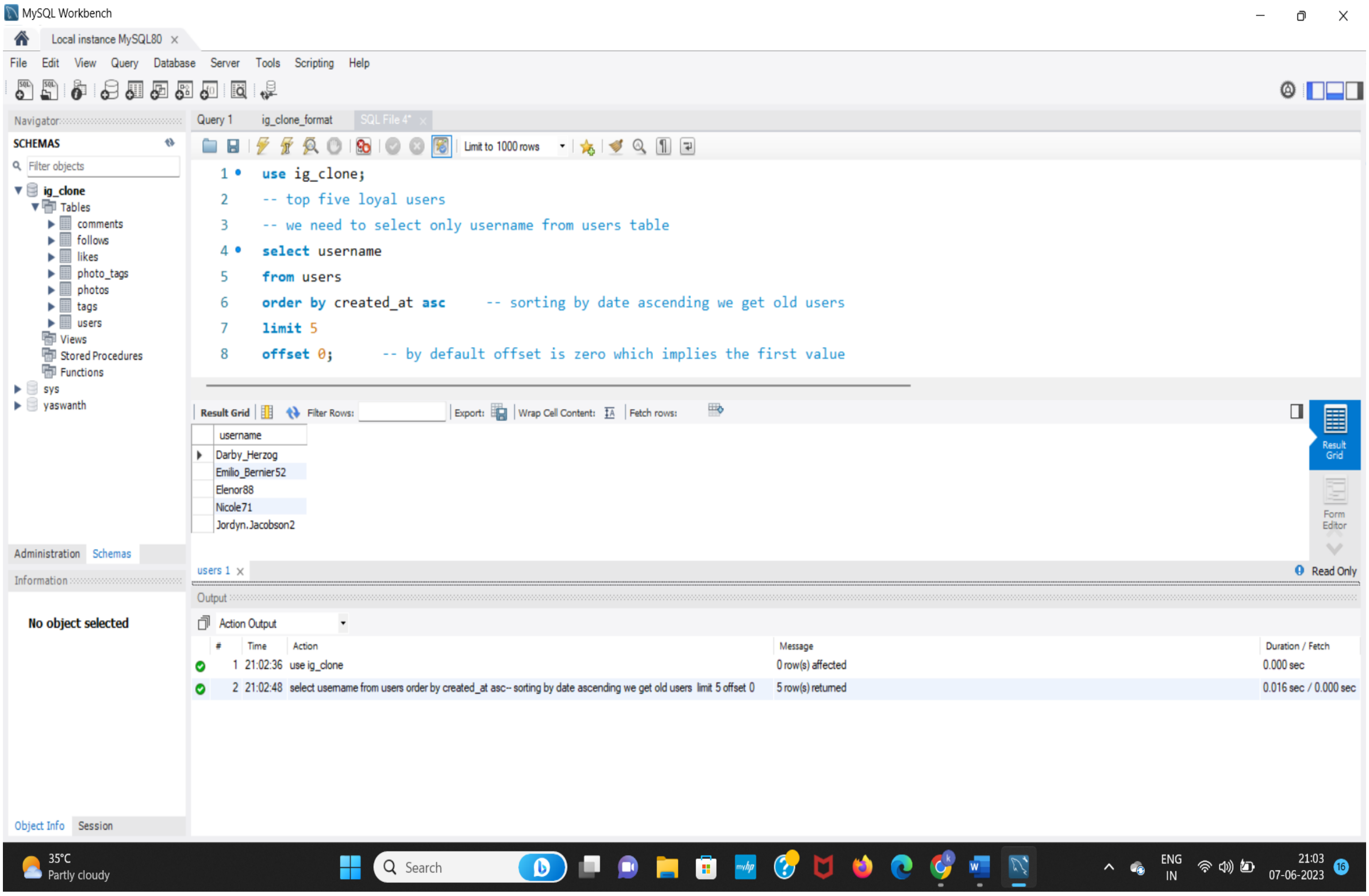
A) To Help the Marketing Team:-

1. **Rewarding Most Loyal Users:** People who have been using the platform for the longest time. Your Task: Find the 5 oldest users of the Instagram from the database provided

   Ans: To find the oldest users of Instagram from users table. So, sort the date (order by created_at) in ascending order which is ascending. We need to find the names of top 5 people. So, we use limit 5 and by default it is offset 0.

SQL QUERY :-

```
select username
from users
order by created_at asc
limit 5
offset 0;
```



```
use ig_clone;
-- top five loyal users
-- we need to select only username from users table
select username
from users
order by created_at asc     -- sorting by date ascending we get old users
limit 5
offset 0;       -- by default offset is zero which implies the first value
```

username
Darby_Herzog
Emilio_Bernier52
Elenor88
Nicole71
Jordyn.Jacobson2

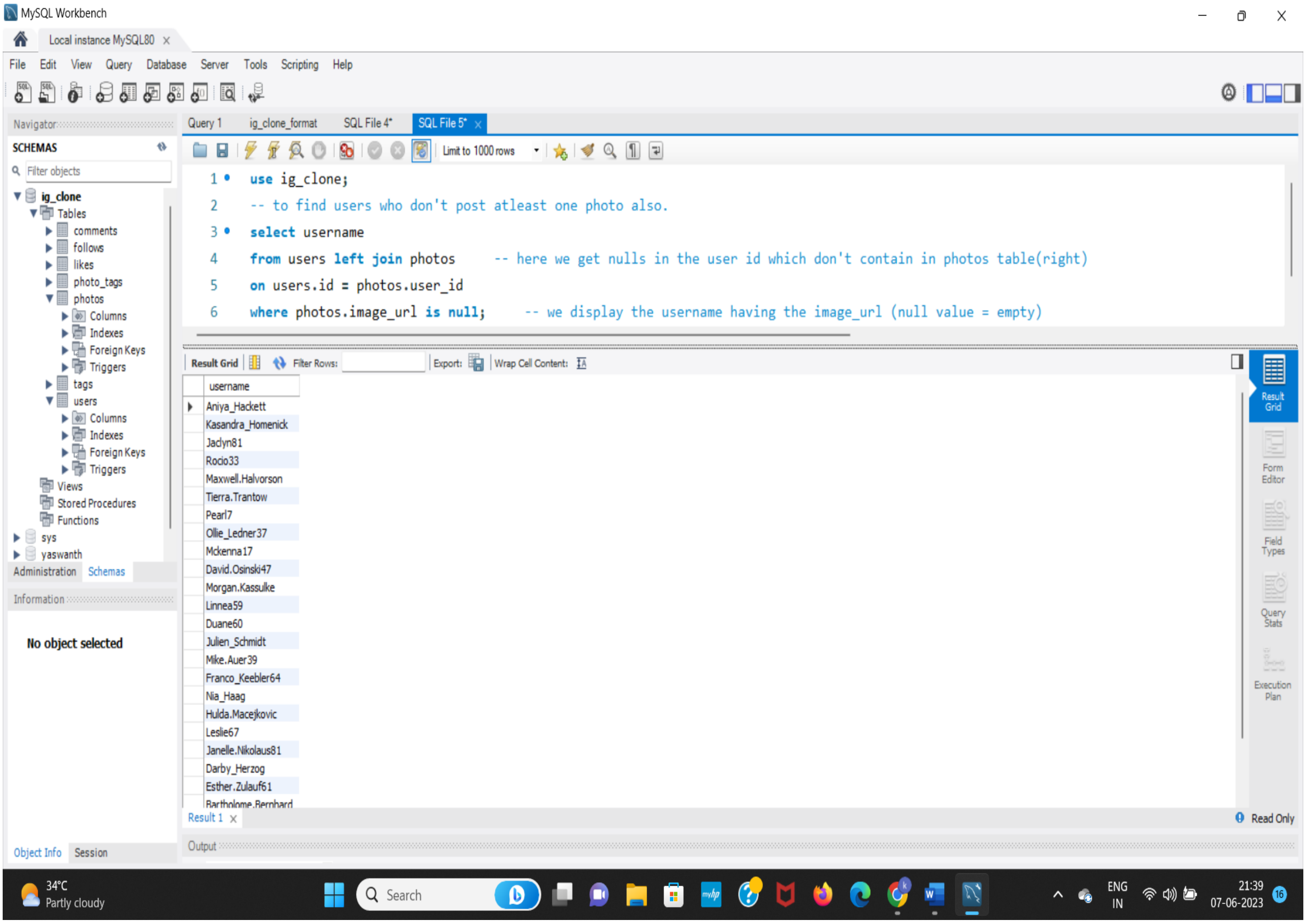
```
#  Time      Action                                                                                                Message               Duration / Fetch
1  21:02:36  use ig_clone                                                                                          0 row(s) affected     0.000 sec
2  21:02:48  select username from users order by created_at asc-- sorting by date ascending we get old users  limit 5 offset 0   5 row(s) returned     0.016 sec / 0.000 sec
```

2. Remind Inactive Users to Start Posting: By sending them promotional emails to post their 1st photo. Your Task: Find the users who have never posted a single photo on Instagram

Ans. To find who don't post even a single photo. We use left join users and photos have user ids as common, then find the photo urls as null and finally we select that names. If to count username we use count(username)

<u>SQL QUERY: -</u>

```
select username    -- to count the users = count(username)
from users left join photos
on users.id = photos.user_id
where photos.image_url is null;
```



```
use ig_clone;
-- to find users who don't post atleast one photo also.
select username
from users left join photos      -- here we get nulls in the user id which don't contain in photos table(right)
on users.id = photos.user_id
where photos.image_url is null;      -- we display the username having the image_url (null value = empty)
```

```
username
Aniya_Hackett
Kasandra_Homenick
Jaclyn81
Rocio33
Maxwell.Halvorson
Tierra.Trantow
Pearl7
Ollie_Ledner37
Mckenna17
David.Osinski47
Morgan.Kassulke
Linnea59
Duane60
Julien_Schmidt
Mike.Auer39
Franco_Keebler64
Nia_Haag
Hulda.Macejkovic
Leslie67
Janelle.Nikolaus81
Darby_Herzog
Esther.Zulauf61
Bartholome.Bernhard
```

3. **Declaring Contest Winner:** The team started a contest and the user who gets the most likes on a single photo will win the contest now they wish to declare the winner. Your Task: Identify the winner of the contest and provide their details to the team

Ans. To find the winner(username) of contest

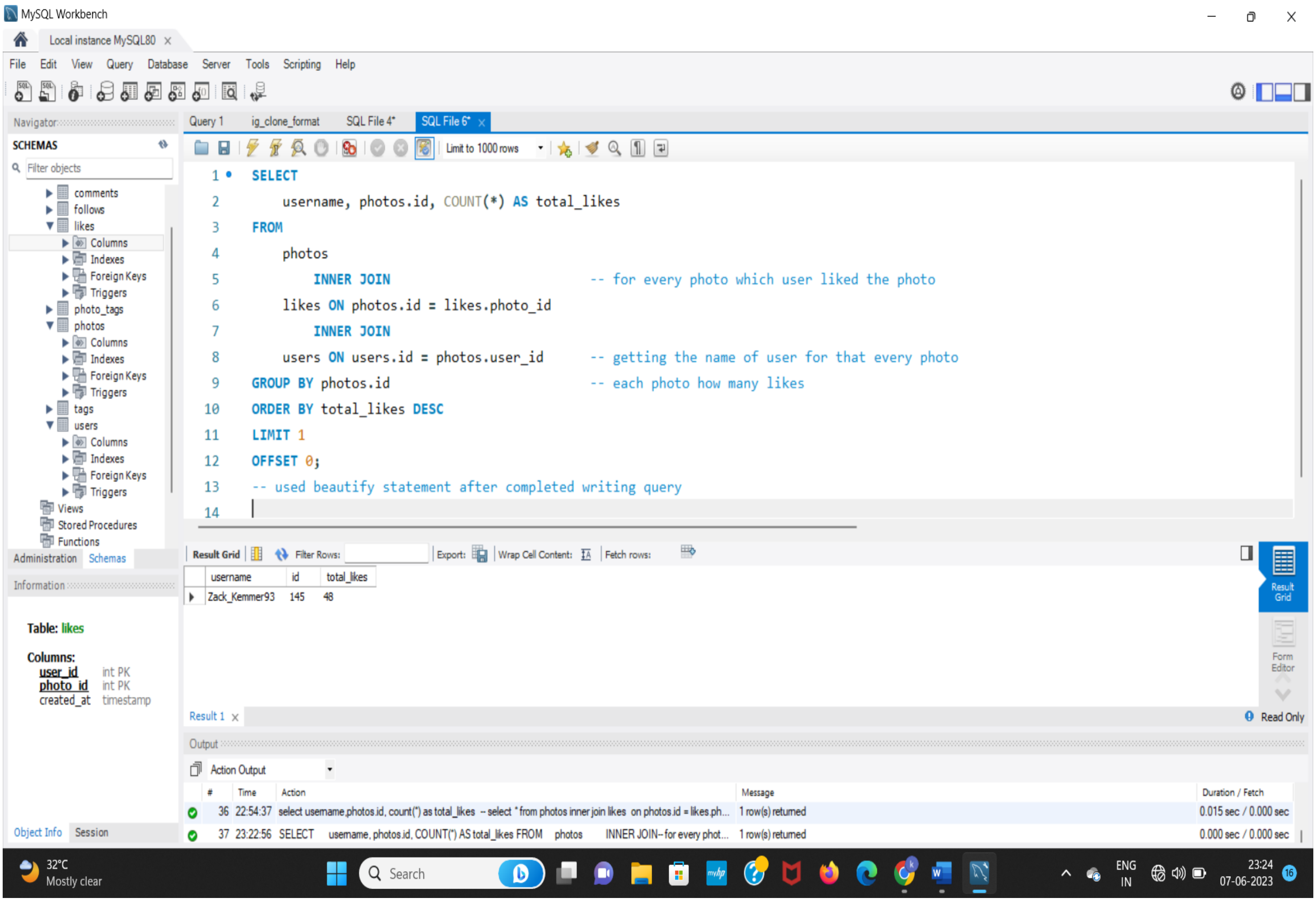
1. For single photo how many likes
   To get that we join photos table and likes table then we get for every photo which user liked the photo. We group the photo_id then perform count (*) as total_likes aggregation then we get for every single photo how many likes are there.
2. We find each photo which user that belongs
   So, we inner join with user table and equal the user id with photos.user_id

3. Then we add username in select column. We sort the table with total_likes descending. Then we get the first row with maximum likes of each photo, and preceding the username it.

SQL QUERY:-

```
SELECT
    username, photos.id, COUNT(*) AS total_likes
FROM
    photos
        INNER JOIN
    likes ON photos.id = likes.photo_id
        INNER JOIN
    users ON users.id = photos.user_id
GROUP BY photos.id
ORDER BY total_likes DESC
LIMIT 1;
```

Note: use beautify option in workbench



```sql
SELECT
    username, photos.id, COUNT(*) AS total_likes
FROM
    photos
        INNER JOIN              -- for every photo which user liked the photo
    likes ON photos.id = likes.photo_id
        INNER JOIN
    users ON users.id = photos.user_id      -- getting the name of user for that every photo
GROUP BY photos.id                          -- each photo how many likes
ORDER BY total_likes DESC
LIMIT 1
OFFSET 0;
-- used beautify statement after completed writing query
```

username	id	total_likes
Zack_Kemmer93	145	48

Table: likes

Columns:
user_id int PK
photo_id int PK
created_at timestamp

```
#   Time      Action                                                                                       Message            Duration / Fetch
36  22:54:37  select username,photos.id, count(*) as total_likes -- select * from photos inner join likes on photos.id = likes.ph...  1 row(s) returned  0.015 sec / 0.000 sec
37  23:22:56  SELECT    username, photos.id, COUNT(*) AS total_likes FROM    photos       INNER JOIN-- for every phot...  1 row(s) returned  0.000 sec / 0.000 sec
```

Zack_kemmer93 user got most likes for the photo of id 145 he is the winner

4. **Hashtag Researching:** A partner brand wants to know, which hashtags to use in the post to reach the most people on the platform. Your Task: Identify and suggest the top 5 most commonly used hashtags on the platform

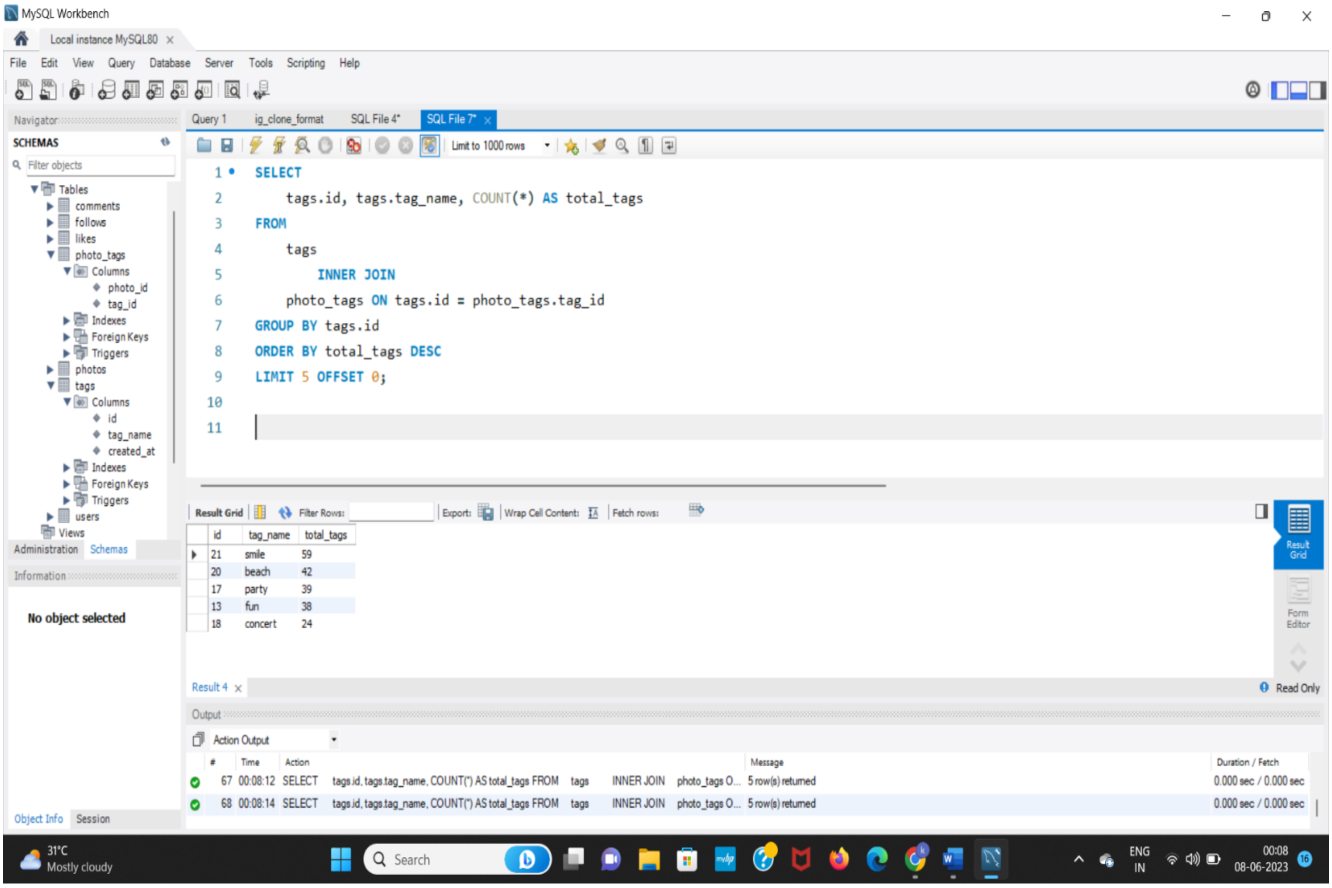
Ans: To find most commonly used hashtags in the post

1. Firstly, join the tags table with photo_tags table on having same tag_id in each table, group by tag_id then perform aggregation count(*) to get total_tags used in all the posts.
2. Sort by total tags descending order, then we need top 5, so we use limit 5 (offset 0 which is first value).

SQL QUERY:-

```
SELECT
    tags.id, tags.tag_name, COUNT(*) AS total_tags
FROM
    tags
        INNER JOIN
    photo_tags ON tags.id = photo_tags.tag_id
GROUP BY tags.id
ORDER BY total_tags DESC
LIMIT 5 OFFSET 0;
```

Note: use beautify option in workbench



```
SELECT
    tags.id, tags.tag_name, COUNT(*) AS total_tags
FROM
    tags
        INNER JOIN
    photo_tags ON tags.id = photo_tags.tag_id
GROUP BY tags.id
ORDER BY total_tags DESC
LIMIT 5 OFFSET 0;
```

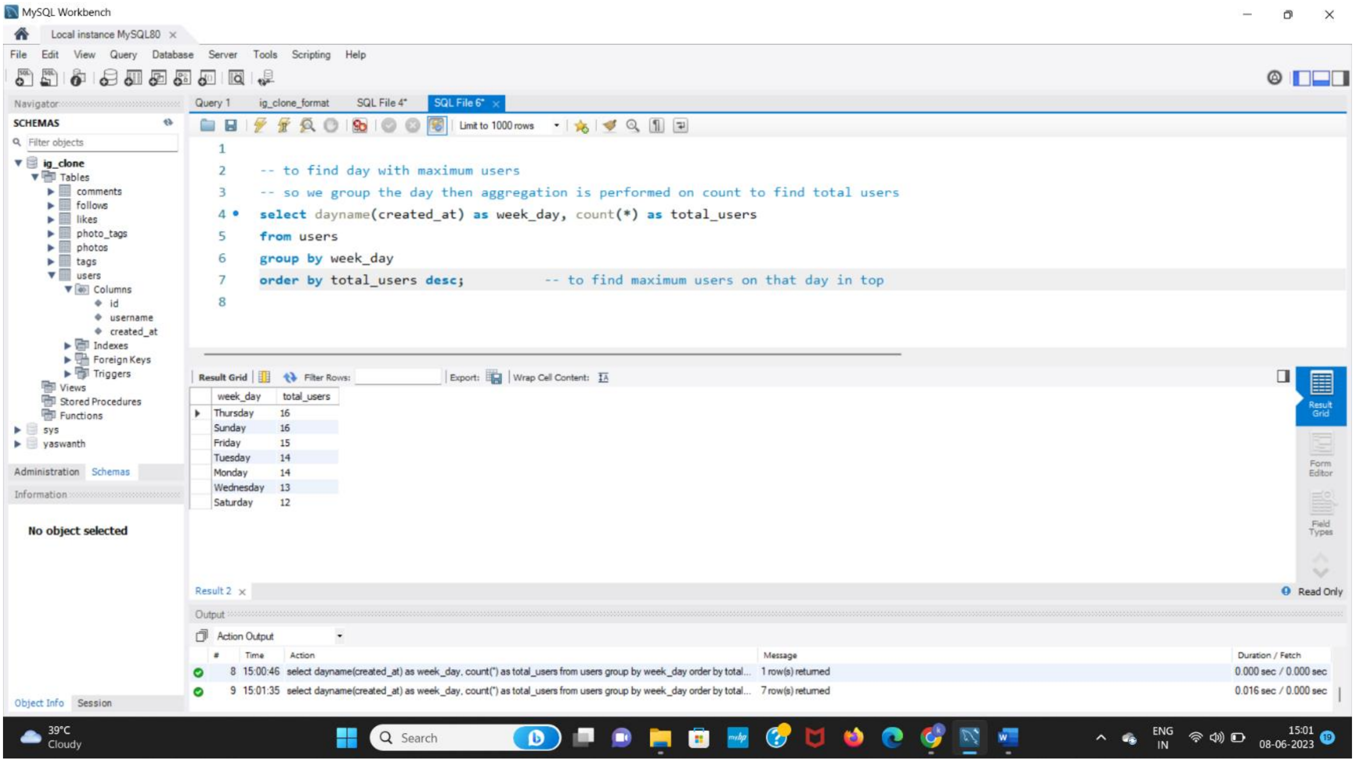
id	tag_name	total_tags
21	smile	59
20	beach	42
17	party	39
13	fun	38
18	concert	24

```
#   Time      Action                                                                       Message            Duration / Fetch
67  00:08:12  SELECT  tags.id, tags.tag_name, COUNT(*) AS total_tags FROM  tags  INNER JOIN  photo_tags O...  5 row(s) returned  0.000 sec / 0.000 sec
68  00:08:14  SELECT  tags.id, tags.tag_name, COUNT(*) AS total_tags FROM  tags  INNER JOIN  photo_tags O...  5 row(s) returned  0.000 sec / 0.000 sec
```

5. **Launch AD Campaign:** The team wants to know, which day would be the best day to launch ADs. Your Task: What day of the week do most users register on? Provide insights on when to schedule an ad campaign

Ans: To find the day where most users registered

1. From date we find which day it is, using dayname(created_at) function and find all the days
2. Then to find the single day we group the day and then perform aggregation count for each day how many users.



```
-- to find day with maximum users
-- so we group the day then aggregation is performed on count to find total users
select dayname(created_at) as week_day, count(*) as total_users
from users
group by week_day
order by total_users desc;          -- to find maximum users on that day in top
```

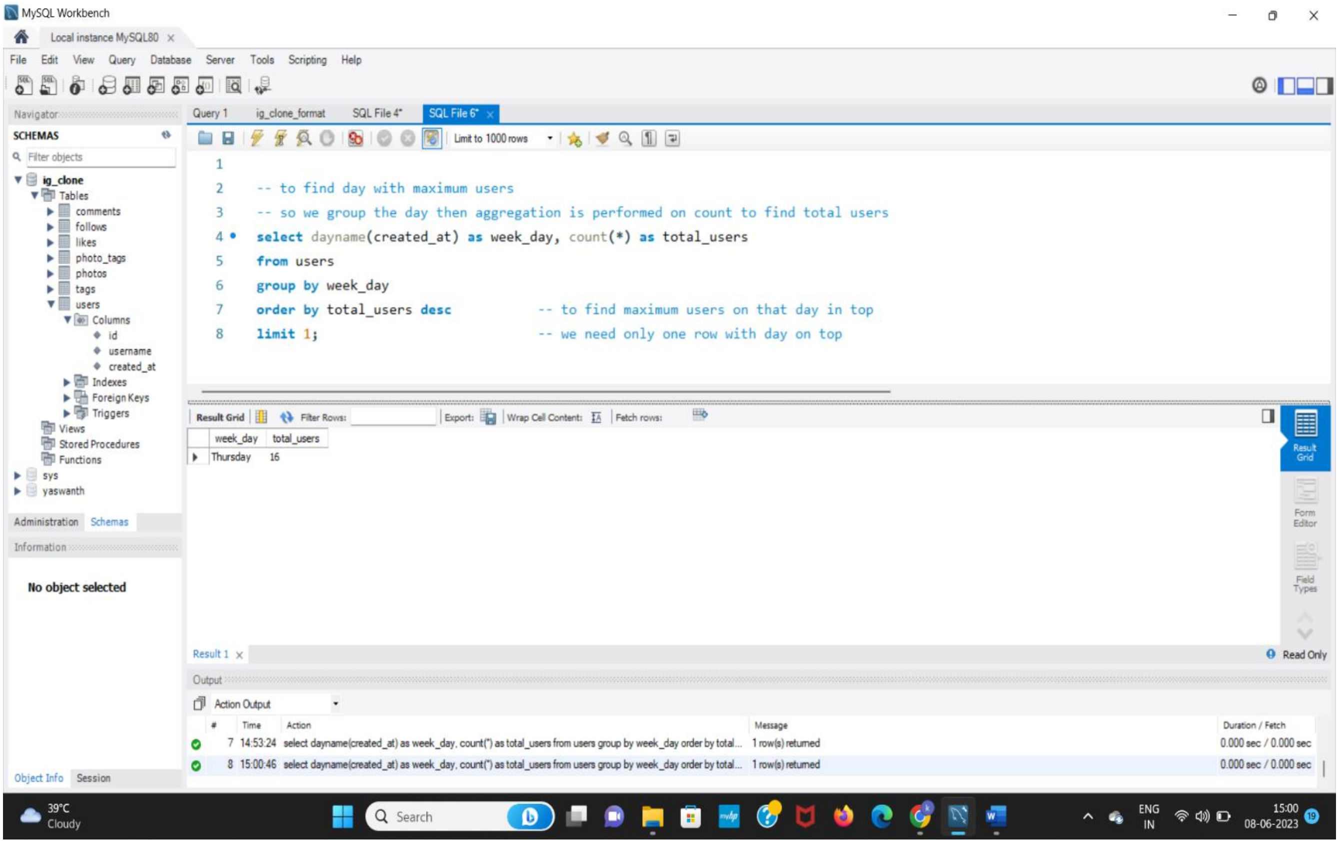
week_day	total_users
Thursday	16
Sunday	16
Friday	15
Tuesday	14
Monday	14
Wednesday	13
Saturday	12

```
#  Time      Action                                                                                               Message           Duration / Fetch
8  15:00:46  select dayname(created_at) as week_day, count(*) as total_users from users group by week_day order by total...  1 row(s) returned  0.000 sec / 0.000 sec
9  15:01:35  select dayname(created_at) as week_day, count(*) as total_users from users group by week_day order by total...  7 row(s) returned  0.016 sec / 0.000 sec
```

SQL QUERY:-

select dayname(created_at) as week_day, count(*) as total_users
from users
group by week_day
order by total_users desc;

Here, we get every day with total users on that day
We conclude that maximum users registered on Thursday and Sunday with maximum of each day 16.
So, we can start the ad campaign either Thursday or Sunday;



```sql
-- to find day with maximum users
-- so we group the day then aggregation is performed on count to find total users
select dayname(created_at) as week_day, count(*) as total_users
from users
group by week_day
order by total_users desc          -- to find maximum users on that day in top
limit 1;                           -- we need only one row with day on top
```

week_day	total_users
Thursday	16

```
#  Time      Action                                                                                        Message            Duration / Fetch
7  14:53:24  select dayname(created_at) as week_day, count(*) as total_users from users group by week_day order by total...  1 row(s) returned  0.000 sec / 0.000 sec
8  15:00:46  select dayname(created_at) as week_day, count(*) as total_users from users group by week_day order by total...  1 row(s) returned  0.000 sec / 0.000 sec
```

Using limit 1 we find the top result.

B. Investor Metrics:

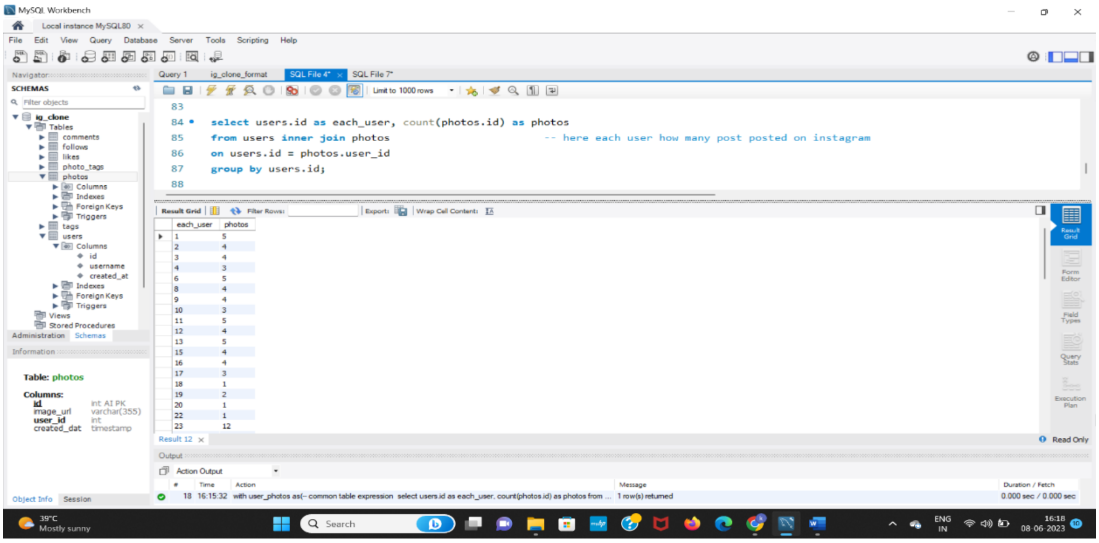
1. **User Engagement:** Are users still as active and post on Instagram or they are making fewer posts. Your Task: Provide how many times does average user posts on Instagram. Also, provide the total number of photos on Instagram/total number of users

Ans: Contains two answers.

i. **To provide how many times does average user posts on Instagram** to find 1 user on average post how many photos on Instagram.

1. Firstly, finding each user how many posts he posts on Instagram
2. To find average, summing the total posts divide by no of users which is average.

(explanation screenshot)



```
select users.id as each_user, count(photos.id) as photos
from users inner join photos                              -- here each user how many post posted on instagram
on users.id = photos.user_id
group by users.id;
```

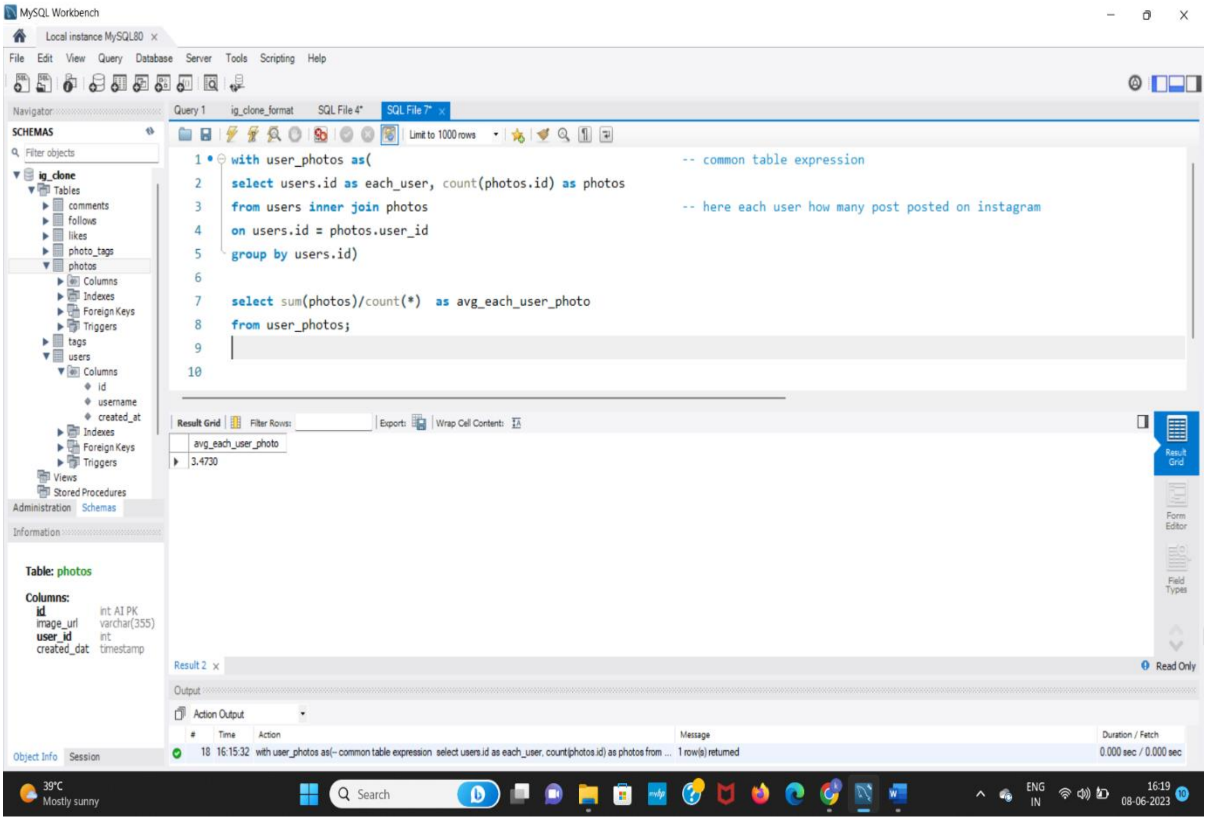
each_user	photos
1	5
2	4
3	4
4	3
6	5
8	4
9	4
10	3
11	5
12	4
13	5
15	4
16	4
17	3
18	1
19	2
20	1
22	1
23	12

```
18  16:15:32  with user_photos as(-- common table expression  select users.id as each_user, count(photos.id) as photos from ...  1 row(s) returned  0.000 sec / 0.000 sec
```

Each user how many posts posted

We create this table as common table expression as user_photos. Then find the average as

Sum(photos) / count(*) [each user] (no of users)



```
with user_photos as(                                    -- common table expression
select users.id as each_user, count(photos.id) as photos
from users inner join photos                            -- here each user how many post posted on instagram
on users.id = photos.user_id
group by users.id)

select sum(photos)/count(*)  as avg_each_user_photo
from user_photos;
```

```
avg_each_user_photo
3.4730
```

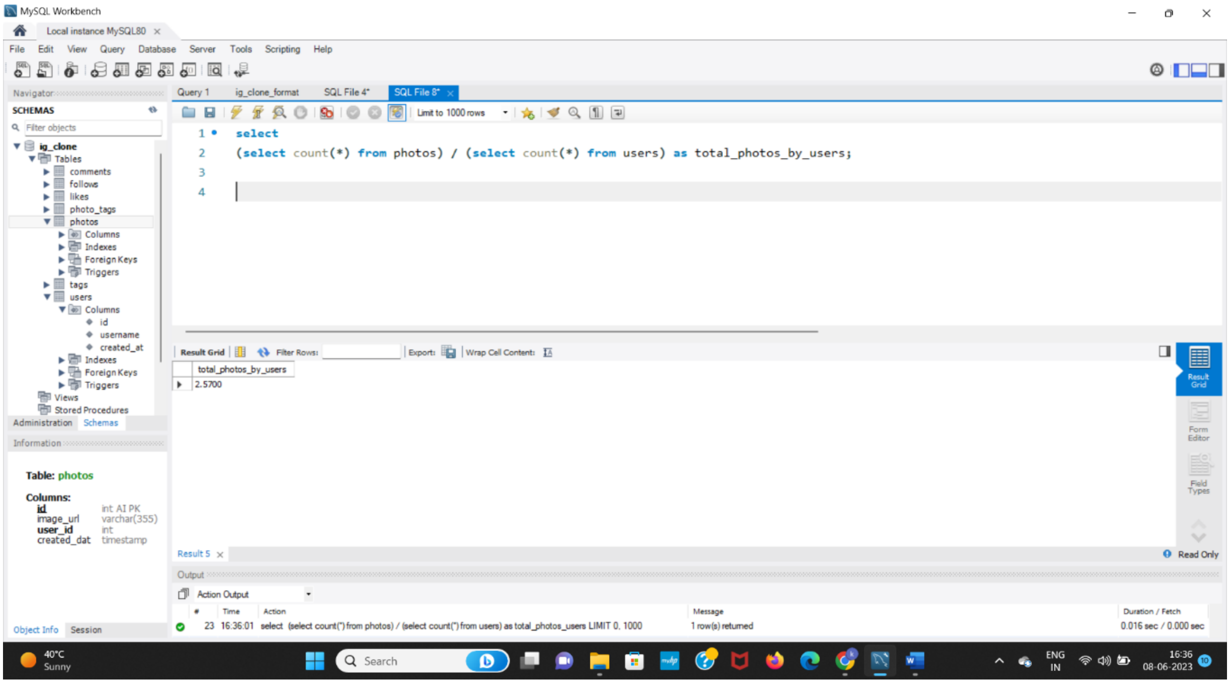
```
#   Time      Action                                                                                              Message            Duration / Fetch
18  16:15:32  with user_photos as(-- common table expression  select users.id as each_user, count(photos.id) as photos from ...  1 row(s) returned  0.000 sec / 0.000 sec
```

<u>Each user on average posts 3 photos posted. (original output)</u>

ii) provide the total number of photos on Instagram/total number of users

ans: select count(*) from photos -- total photos on photos table

select count(*) from users -- total users on users table



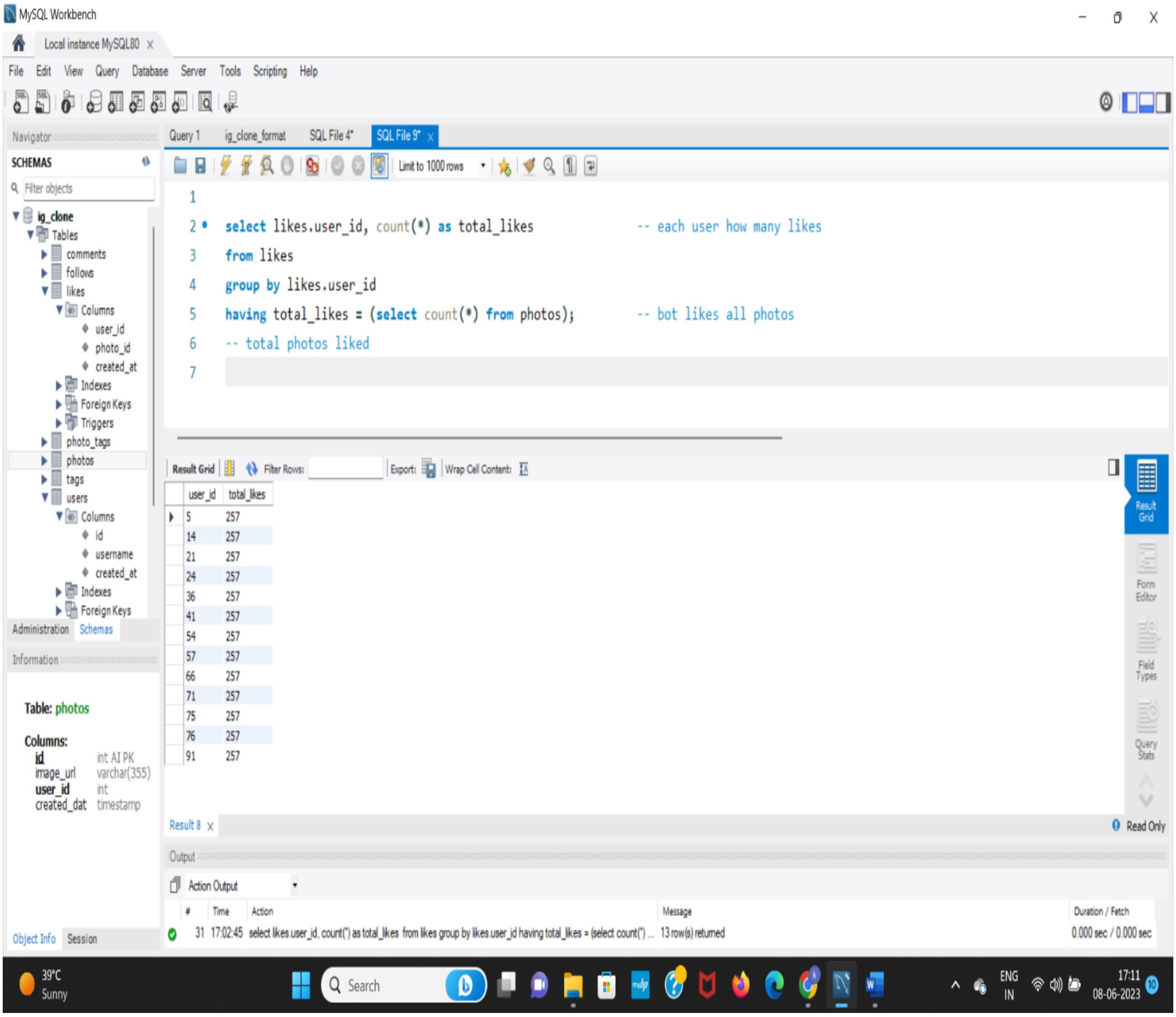
```
select
(select count(*) from photos) / (select count(*) from users) as total_photos_by_users;
```

```
total_photos_by_users
2.5700
```

```
#   Time      Action                                                                                   Message            Duration / Fetch
23  16:36:01  select  (select count(*) from photos) / (select count(*) from users) as total_photos_users LIMIT 0, 1000  1 row(s) returned  0.016 sec / 0.000 sec
```

2. Bots & Fake Accounts: The investors want to know if the platform is crowded with fake and dummy accounts. Your Task: Provide data on users (bots) who have liked every single photo on the site (since any normal user would not be able to do this).

Ans:- Here, if the user likes all the photos that implies the user is having fake account.



```sql

select likes.user_id, count(*) as total_likes          -- each user how many likes
from likes
group by likes.user_id
having total_likes = (select count(*) from photos);        -- bot likes all photos
-- total photos liked

```

user_id	total_likes
5	257
14	257
21	257
24	257
36	257
41	257
54	257
57	257
66	257
71	257
75	257
76	257
91	257

Table: photos

Columns:
id int AI PK
image_url varchar(355)
user_id int
created_dat timestamp

```
#   Time      Action                                                                                            Message            Duration / Fetch
31  17:02:45  select likes.user_id, count(*) as total_likes from likes group by likes.user_id having total_likes = (select count(*) ...  13 row(s) returned  0.000 sec / 0.000 sec
```

This user_id are fake user_ids.

Result:-

By doing this project, I have gained knowledge on how to interact with the database and get our wanted information through sql. And how to write complex queries and make them in easily understandable format of humans.

Overall, through this steps I have successfully completed the project on marketing and Investor metrics.

Thank you Trainity 😊

Happy Learning :>)