

Project: Instagram User Analytics

NAME: MUKESH CHANDRA KAMILA

Description:

In this project, we are working with the product team of Instagram and the product manager. We are asked to provide insights on the questions asked by the management team. Then with help of these insights, teams in the business can launch a new marketing campaign, develop features for Instagram app, track the success of the Instagram by analyse user engagement and improve the user experience altogether while helping the business to grow.

Approach:

To execute the project, we used SQL. So, the first approach was to run SQL queries to create a database using the raw data provided. Once the database was created, we run various sorting and data extracting queries to get the required insights.

Tech-Stack: PostgreSQL 6.19

Used:

PostgreSQL 6.19 was used in this project execution. The ease of access and set up with convenient user interface made it a good tool for the project.

Insights:

In this project, we learned about fundamentals of SQL. And how to analyse the problem statement and the functions we can use in SQL to solve the problem statement and write the queries to get required output. Following are the questions that has been answered with the help of SQL.

Result:

In this project, I have achieved and gain knowledge how to deal with data with help of SQL. And how to interact and run queries to get desired output from database.

A) Marketing Metrics

Q1. We want to reward our users who have been around the longest. Find the 5 oldest users?

Query:

```
select * from users
```

```
order by created_at asc
```

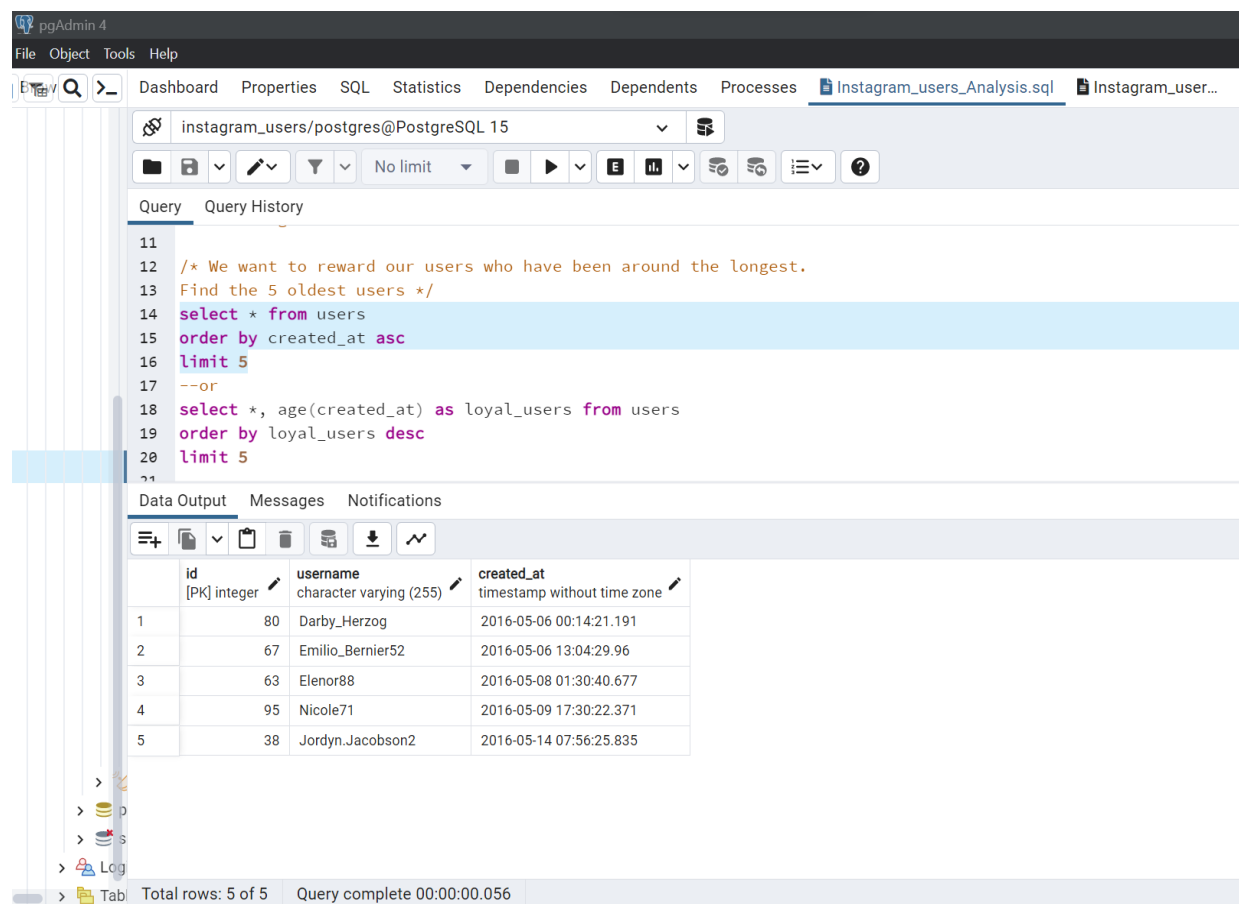
```
limit 5
```

--or

```
select *, age(created_at) as loyal_users from users
```

```
order by loyal_users desc
```

```
limit 5;
```



The screenshot shows the pgAdmin 4 interface. The SQL query editor contains the following code:

```
11
12 /* We want to reward our users who have been around the longest.
13 Find the 5 oldest users */
14 select * from users
15 order by created_at asc
16 limit 5
17 --or
18 select *, age(created_at) as loyal_users from users
19 order by loyal_users desc
20 limit 5
21
```

The 'Data Output' tab shows the results of the query in a table with 5 rows:

	id [PK] integer	username character varying (255)	created_at timestamp without time zone
1	80	Darby_Herzog	2016-05-06 00:14:21.191
2	67	Emilio_Bernier52	2016-05-06 13:04:29.96
3	63	Elenor88	2016-05-08 01:30:40.677
4	95	Nicole71	2016-05-09 17:30:22.371
5	38	Jordyn.Jacobson2	2016-05-14 07:56:25.835

The status bar at the bottom indicates 'Total rows: 5 of 5' and 'Query complete 00:00:00.056'.

Q2. We want to target our inactive users with an email campaign. Find the users who have never posted a single photo.

Query:

select username from users

left join photos

on users.id = photos.user_id

where photos.user_id is null;

The screenshot shows the pgAdmin 4 interface. The top menu bar includes File, Object, Tools, and Help. The main toolbar contains icons for various database actions. The central pane displays a SQL query in a text editor, with line numbers 22 through 29. The query is as follows:

```
22
23 /* We want to target our inactive users with an email campaign.
24 Find the users who have never posted a single photo */
25 select username from users
26 left join photos
27 on users.id = photos.user_id
28 where photos.user_id is null;
29
```

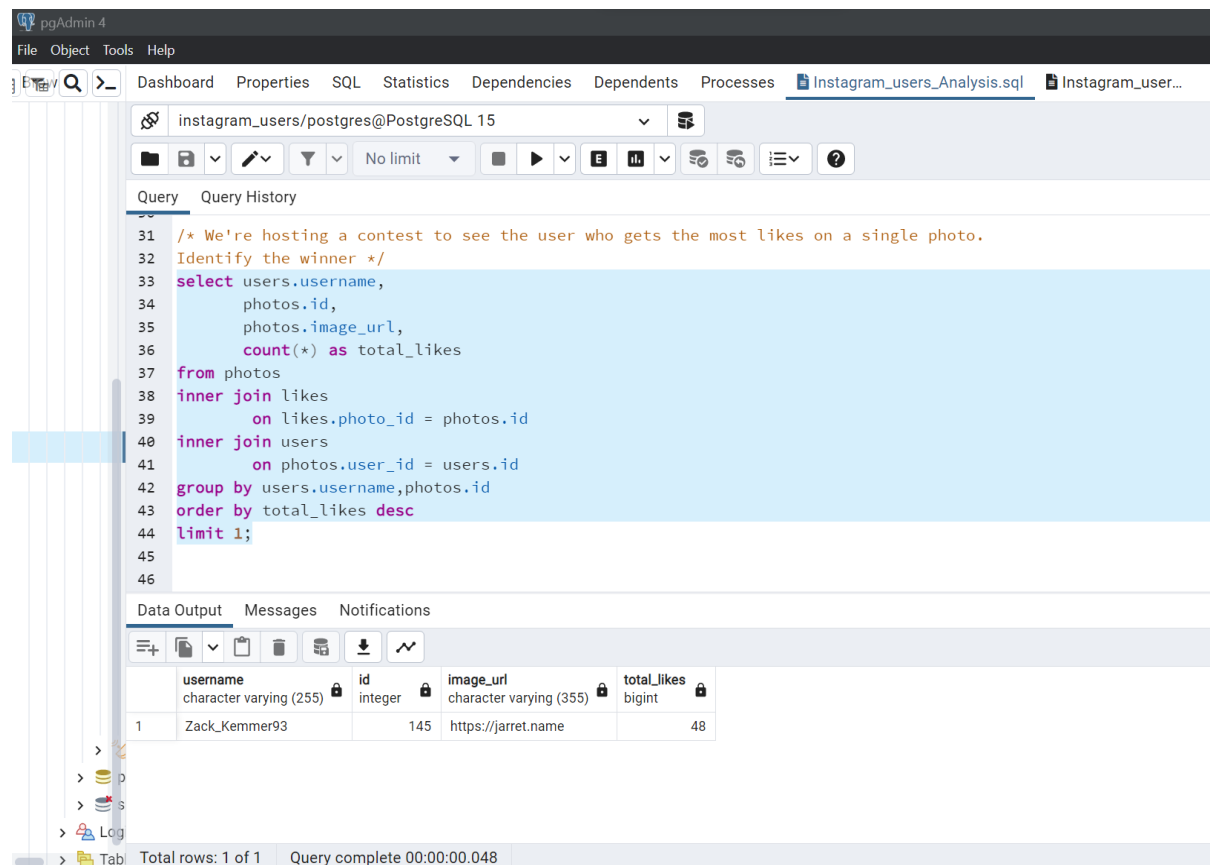
Below the query editor, the 'Data Output' tab is active, showing a table with the results of the query. The table has a single column 'username' of type 'character varying (255)'. The results are listed in a table with 12 rows, numbered 1 through 12. The status bar at the bottom indicates 'Total rows: 26 of 26' and 'Query complete 00:00:00.148'.

	username
1	Aniya_Hackett
2	Kassandra_Homenick
3	Jaclyn81
4	Rocio33
5	Maxwell.Halvorson
6	Tierra.Trantow
7	Pearl7
8	Ollie_Ledner37
9	Mckenna17
10	David.Osinski47
11	Morgan.Kassulke
12	Linnea59

Q3. We're hosting a contest to see the user who gets the most likes on a single photo. Identify the winner?

Query:

```
select users.username,  
       photos.id,  
       photos.image_url,  
       count(*) as total_likes  
from photos  
inner join likes  
       on likes.photo_id = photos.id  
inner join users  
       on photos.user_id = users.id  
group by users.username,photos.id  
order by total_likes desc  
limit 1;
```



The screenshot shows the pgAdmin 4 interface. The top menu bar includes File, Object, Tools, and Help. The main toolbar contains icons for various database actions. The 'Query' tab is active, displaying a SQL query that identifies the user with the most likes on a single photo. The query is as follows:

```
31 /* We're hosting a contest to see the user who gets the most likes on a single photo.  
32 Identify the winner */  
33 select users.username,  
34        photos.id,  
35        photos.image_url,  
36        count(*) as total_likes  
37 from photos  
38 inner join likes  
39     on likes.photo_id = photos.id  
40 inner join users  
41     on photos.user_id = users.id  
42 group by users.username,photos.id  
43 order by total_likes desc  
44 limit 1;  
45  
46
```

Below the query editor, the 'Data Output' tab shows the results of the query. The results are displayed in a table with the following columns: username, id, image_url, and total_likes. The table contains one row of data:

username	id	image_url	total_likes
Zack_Kemmer93	145	https://jarret.name	48

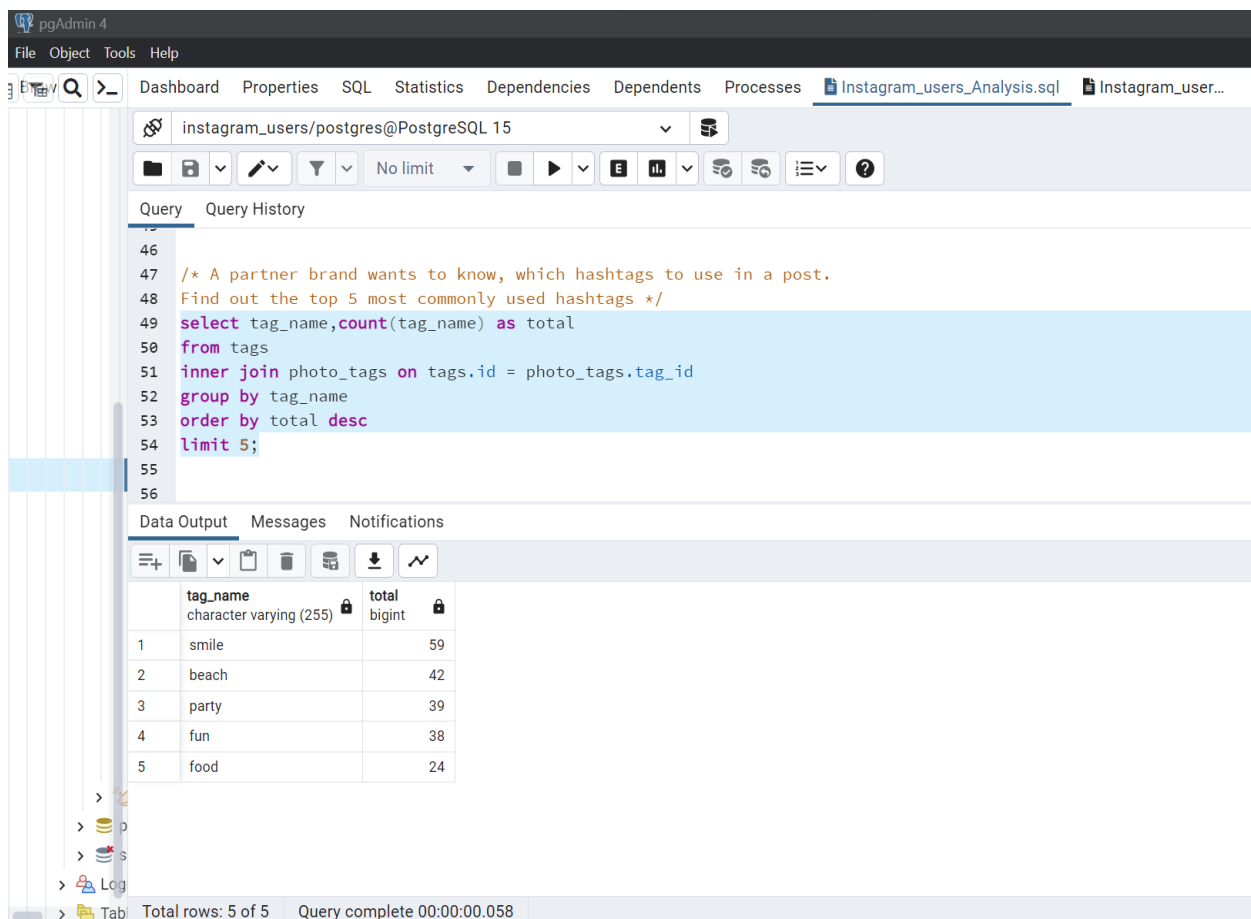
The status bar at the bottom indicates 'Total rows: 1 of 1' and 'Query complete 00:00:00.048'.

Q4. A partner brand wants to know, which hashtags to use in a post.

Find out the top 5 most commonly used hashtags?

Query:

```
select tag_name,count(tag_name) as total
from tags
inner join photo_tags on tags.id = photo_tags.tag_id
group by tag_name
order by total desc
limit 5;
```



The screenshot shows the pgAdmin 4 web interface. The top navigation bar includes 'File', 'Object', 'Tools', and 'Help'. Below it, a toolbar contains icons for various actions. The main panel is divided into two sections: 'Query' and 'Query History'. The 'Query' section displays the following SQL query:

```
/* A partner brand wants to know, which hashtags to use in a post.
Find out the top 5 most commonly used hashtags */
select tag_name,count(tag_name) as total
from tags
inner join photo_tags on tags.id = photo_tags.tag_id
group by tag_name
order by total desc
limit 5;
```

Below the query editor, the 'Data Output' tab is active, showing the results of the query in a table format:

	tag_name character varying (255)	total bigint
1	smile	59
2	beach	42
3	party	39
4	fun	38
5	food	24

The bottom status bar indicates 'Total rows: 5 of 5' and 'Query complete 00:00:00.058'.

Q5. What day of the week do most users register on?

We need to find out, which day would be the best day to launch Ads?

Query:

select

to_char(created_at,'day') as day,

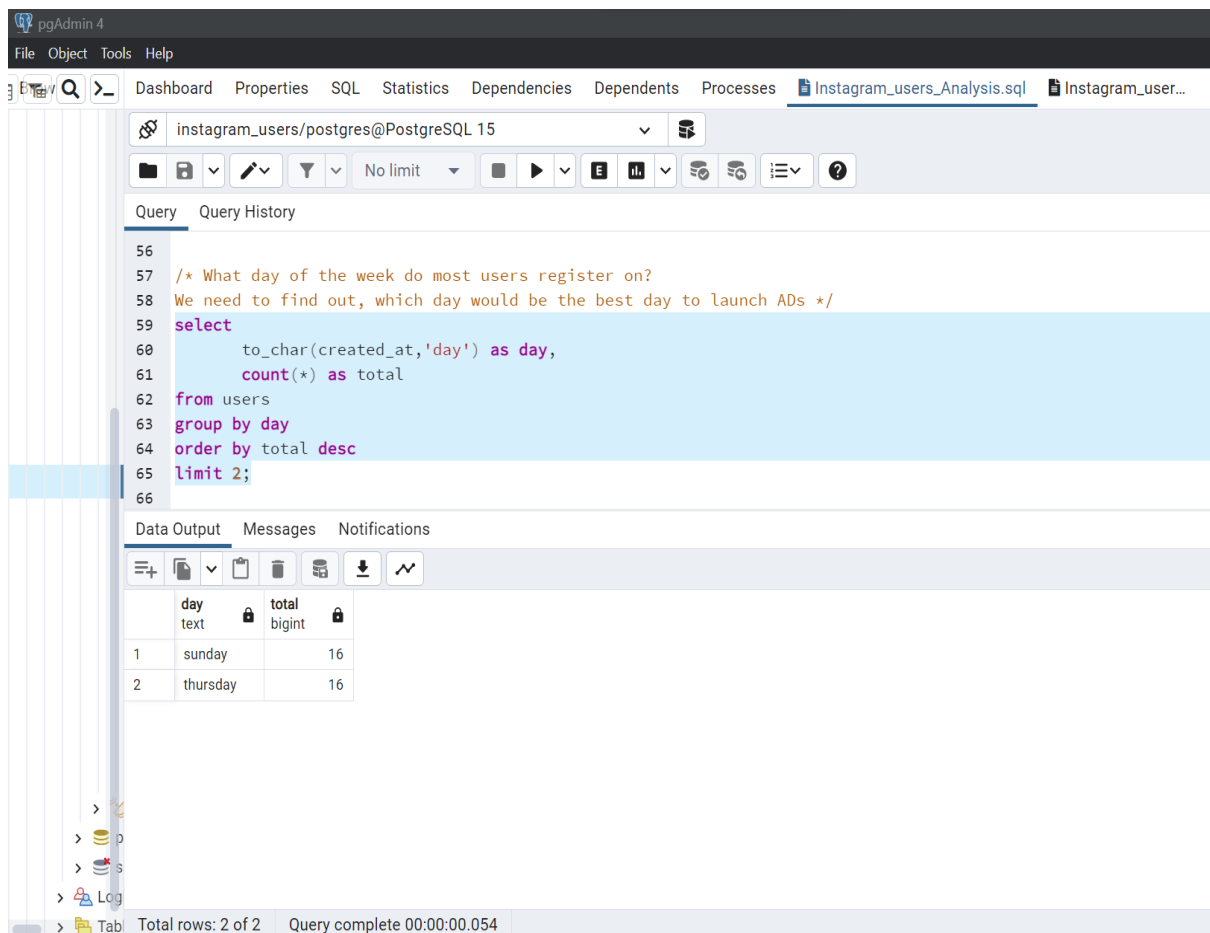
count (*) as total

from users

group by day

order by total desc

limit 2;



The screenshot shows the pgAdmin 4 interface. The SQL query editor contains the following code:

```
56
57 /* What day of the week do most users register on?
58 We need to find out, which day would be the best day to launch Ads */
59 select
60     to_char(created_at,'day') as day,
61     count(*) as total
62 from users
63 group by day
64 order by total desc
65 limit 2;
66
```

The query results are displayed in the Data Output tab, showing a table with two columns: 'day' (text) and 'total' (bigint). The results are as follows:

day	total
sunday	16
thursday	16

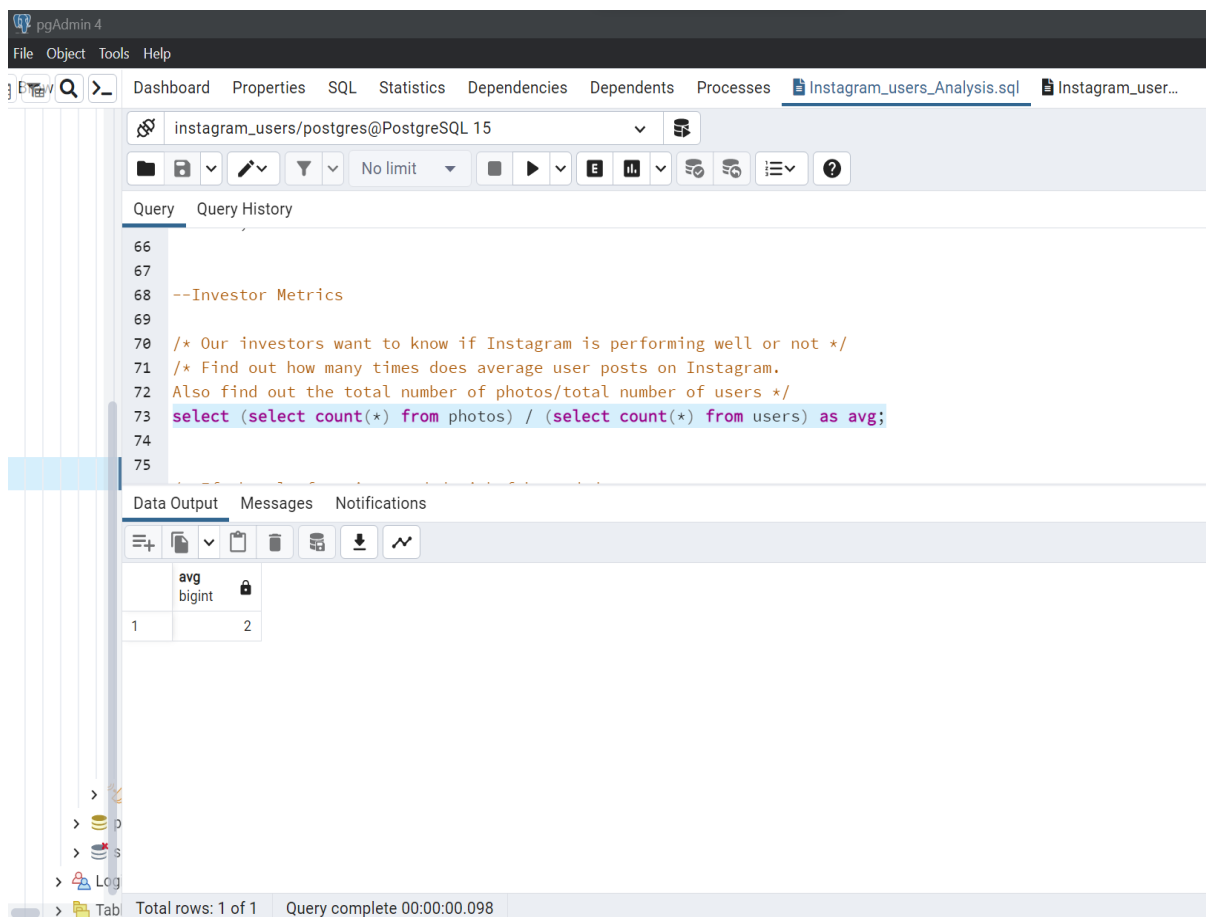
The status bar at the bottom indicates 'Total rows: 2 of 2' and 'Query complete 00:00:00.054'.

B) Investor Metrics

Q1: Our investors want to know if Instagram is performing well or not. Find out how many times does average user posts on Instagram. Also find out the total number of photos/total number of users.

Query:

select (select count (*) from photos) / (select count (*) from users) as avg;



The screenshot shows the pgAdmin 4 web interface. The top navigation bar includes 'File', 'Object', 'Tools', and 'Help'. Below it, a toolbar contains various icons for file operations, query execution, and data management. The main panel is divided into two tabs: 'Query' and 'Query History'. The 'Query' tab is active, displaying a SQL query in a text editor. The query is as follows:

```
66
67
68 --Investor Metrics
69
70 /* Our investors want to know if Instagram is performing well or not */
71 /* Find out how many times does average user posts on Instagram.
72 Also find out the total number of photos/total number of users */
73 select (select count(*) from photos) / (select count(*) from users) as avg;
74
75
```

Below the query editor, there are three tabs: 'Data Output', 'Messages', and 'Notifications'. The 'Data Output' tab is active, showing a table with one row and two columns. The table has a header row with 'avg' and 'bigint' (with a lock icon). The data row contains the values '1' and '2'.

avg	bigint
1	2

At the bottom of the interface, a status bar shows 'Total rows: 1 of 1' and 'Query complete 00:00:00.098'.

Q2. If the platform is crowded with fake and dummy accounts.

Find users who have liked every single photo on the site?

Query:

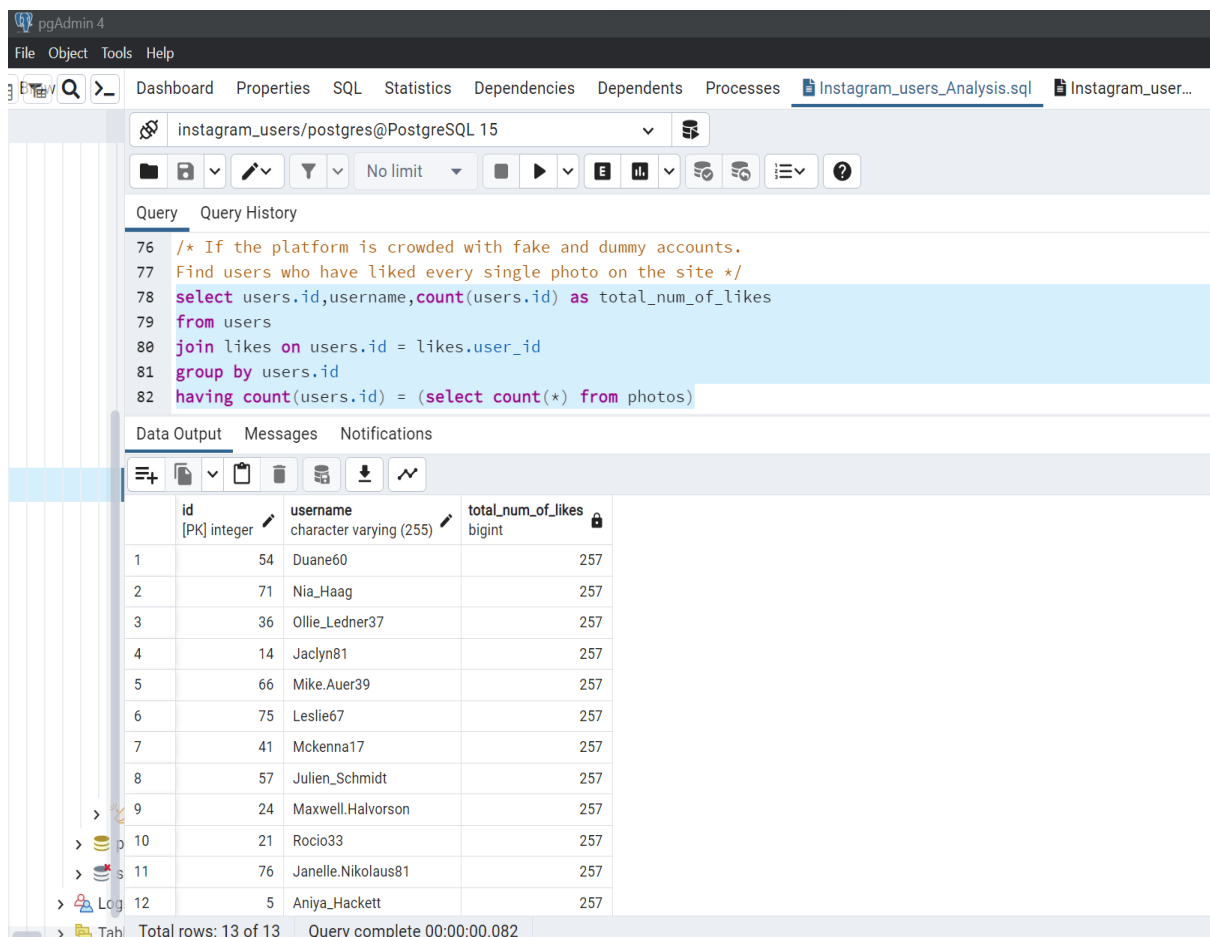
```
select users.id,username,count(users.id) as total_num_of_likes
```

```
from users
```

```
join likes on users.id = likes.user_id
```

```
group by users.id
```

```
having count(users.id) = (select count (*) from photos)
```



The screenshot shows the pgAdmin 4 interface. The query editor contains the following SQL code:

```
/* If the platform is crowded with fake and dummy accounts.
Find users who have liked every single photo on the site */
select users.id,username,count(users.id) as total_num_of_likes
from users
join likes on users.id = likes.user_id
group by users.id
having count(users.id) = (select count (*) from photos)
```

The query results are displayed in a table with the following columns: id (PK) integer, username character varying (255), and total_num_of_likes bigint. The results show 13 rows of data.

id	username	total_num_of_likes
54	Duane60	257
71	Nia_Haag	257
36	Ollie_Ledner37	257
14	Jaclyn81	257
66	Mike_Auer39	257
75	Leslie67	257
41	Mckenna17	257
57	Julien_Schmidt	257
24	Maxwell_Halvorson	257
21	Rocio33	257
76	Janelle.Nikolaus81	257
5	Aniya_Hackett	257

Total rows: 13 of 13 Query complete 00:00:00.082