

Project Report

Instagram Data
Analytics

Mukesh Dey



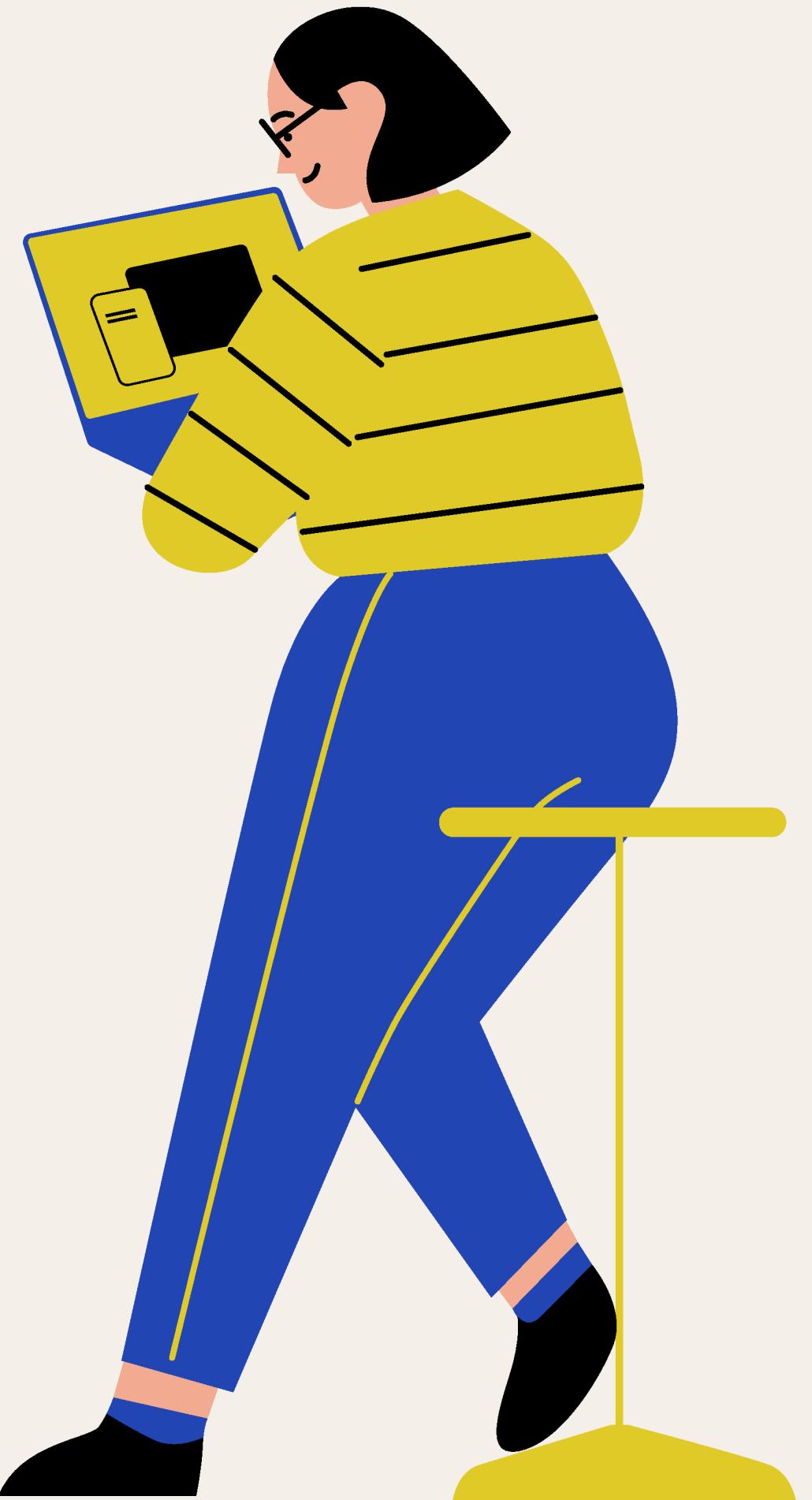
O1 - Introduction

O2 - Marketing Analytics

O3 - Investors Metrics

Data

Analytics



01 - Introduction

In this project, I'll be using SQL and MySQL Workbench as my tools to analyze Instagram user data and answer questions posted by the management team. My insights will help the product manager and the rest of the team make informed decisions about the future direction of the Instagram app.

My findings could potentially influence the future development of one of the world's most popular social media platforms.

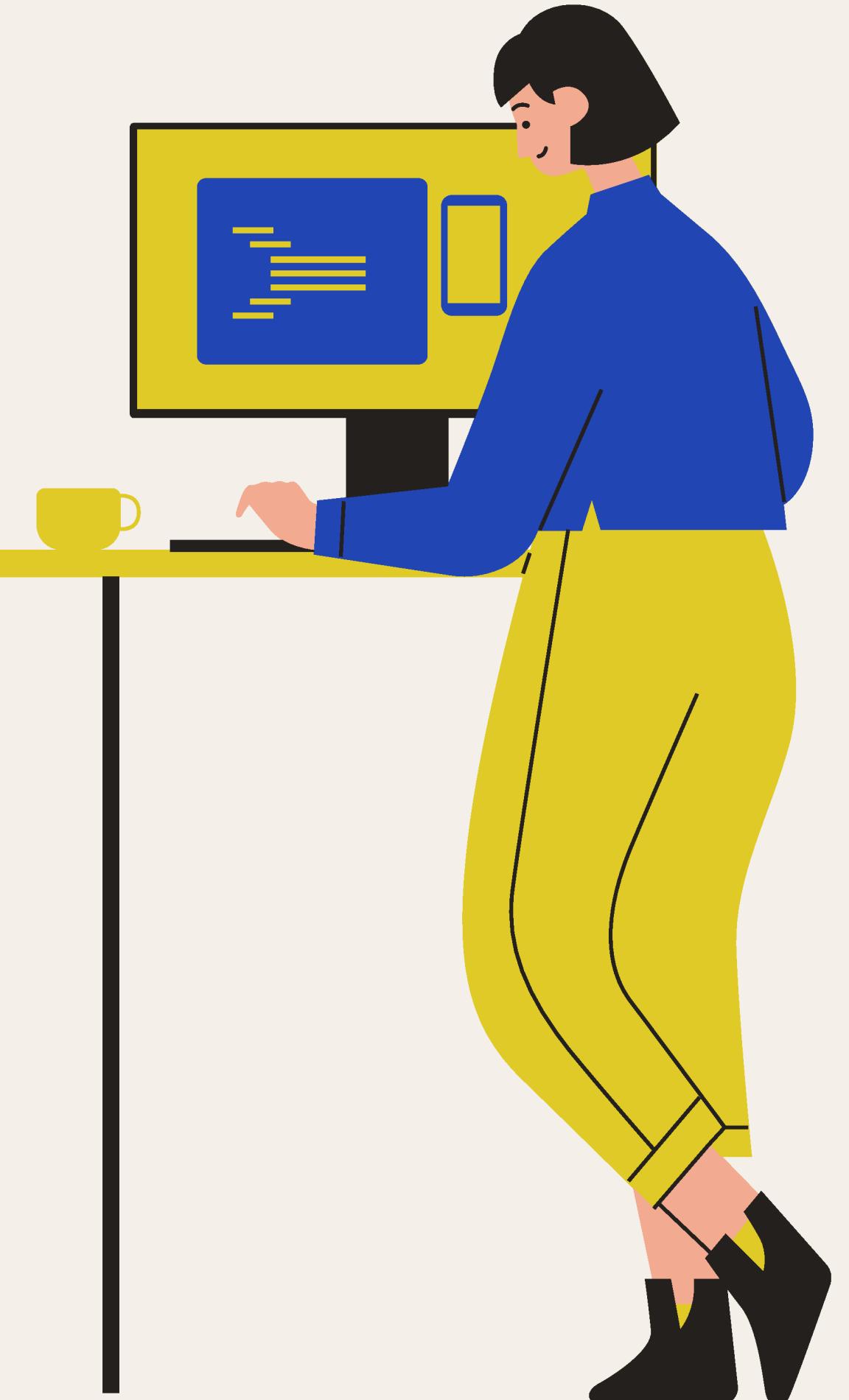
Analyzing data enables informed decision-making

Data

Analytics

O2 - Marketing Analysis

- Loyal User Reward.
- Inactive User Engagement.
- Contest Winner Declaration.
- Hashtag Research.
- Ad Campaign Launch.



• Loyal User Reward

The marketing team wants to reward the most loyal users, i.e., those who have been using the platform for the longest time. So for that the loyal reward will be given to five oldest users on Instagram.

The screenshot shows the MySQL Workbench interface. The left sidebar displays the database schema with two databases: 'ig_clone' and 'sakila'. The 'ig_clone' database contains tables like photo_tags, likes, follows, comments, photos, tags, and users. The 'sakila' database contains tables such as actor, address, category, city, country, customer, film, film_actor, film_category, film_text, and inventory. The main window shows a query editor titled 'Query 1' with the following SQL code:

```
3 •  SELECT * FROM users;
4
5 •  SELECT * FROM follows;
6  #Loyal User Reward
7
8 •  SELECT * FROM users ORDER BY created_at ASC LIMIT 5;
```

The result grid below the query editor shows the output of the last query, which retrieves the top 5 oldest users from the 'users' table. The results are:

	id	username	created_at
▶	180	Darby_Herzog	2016-05-06 00:14:21
	80	Darby_Herzog	2016-05-06 00:14:21
	280	Darby_Herzog	2016-05-06 00:14:21
	380	Darby_Herzog	2016-05-06 00:14:21
*	480	Darby_Herzog	2016-05-06 00:14:21
	HULL	HULL	HULL

So, here's
the top 5
oldest users
on
Instagram.

Data

Analytics

• Loyal User Reward Insights

- Top 5 users Account ID: **180, 80, 280, 380, 480.**
- Here, top 5 accounts which are created are with the same user name i.e. **Darby_Herzog**
- Account Creation time & date also are the same for 5 of the accounts id i.e. **2016-05-06 00:14:21**. The user created all the 5 accounts at the same time with different id but with same user_name.

Data

Analytics

• Inactive User Engagement

The team wants to encourage inactive users to start posting by sending them promotional emails. So, here's the users who have never posted a single photo on Instagram.

The screenshot shows the MySQL Workbench interface. The left sidebar displays the 'Schemas' tree, with 'ig_clone' selected, showing tables like photo_tags, likes, follows, comments, photos, tags, and users. Below it, the 'sakila' schema is shown with tables such as actor, address, category, city, country, customer, film, film_actor, film_category, film_text, inventory, and language. The main area contains a 'Query 1' tab with the following SQL code:

```
4
5 •  SELECT * FROM follows;
6 #Loyal User Reward
7
8 •  SELECT * FROM users ORDER BY created_at ASC LIMIT 5;
9
10 #Inactive Users Engagement
11
12 •  SELECT * FROM photos WHERE user_id IS NULL;
```

The 'Result Grid' pane shows a table with four columns: id, image_url, user_id, and created_dat. All values are null. The 'Output' pane at the bottom shows the execution log for the queries:

#	Time	Action	Message	Duration / Fetch
84	19:47:34	SELECT * FROM photos LIMIT 0, 1000	1000 row(s) returned	0.000 sec / 0.000 sec
85	19:49:54	SELECT * FROM photos WHERE user_id IS NULL LIMIT 0, 1000	0 row(s) returned	0.000 sec / 0.000 sec
86	19:50:15	SELECT * FROM photos LIMIT 0, 1000	1000 row(s) returned	0.016 sec / 0.000 sec
87	19:54:34	SELECT * FROM tags LIMIT 0, 1000	21 row(s) returned	0.000 sec / 0.000 sec
88	20:11:23	SELECT * FROM photos WHERE user_id IS NULL LIMIT 0, 1000	0 row(s) returned	0.000 sec / 0.000 sec

But, there's no user without a single photo posted on Instagram

Data

Analytics

• Inactive User Engagement Insights

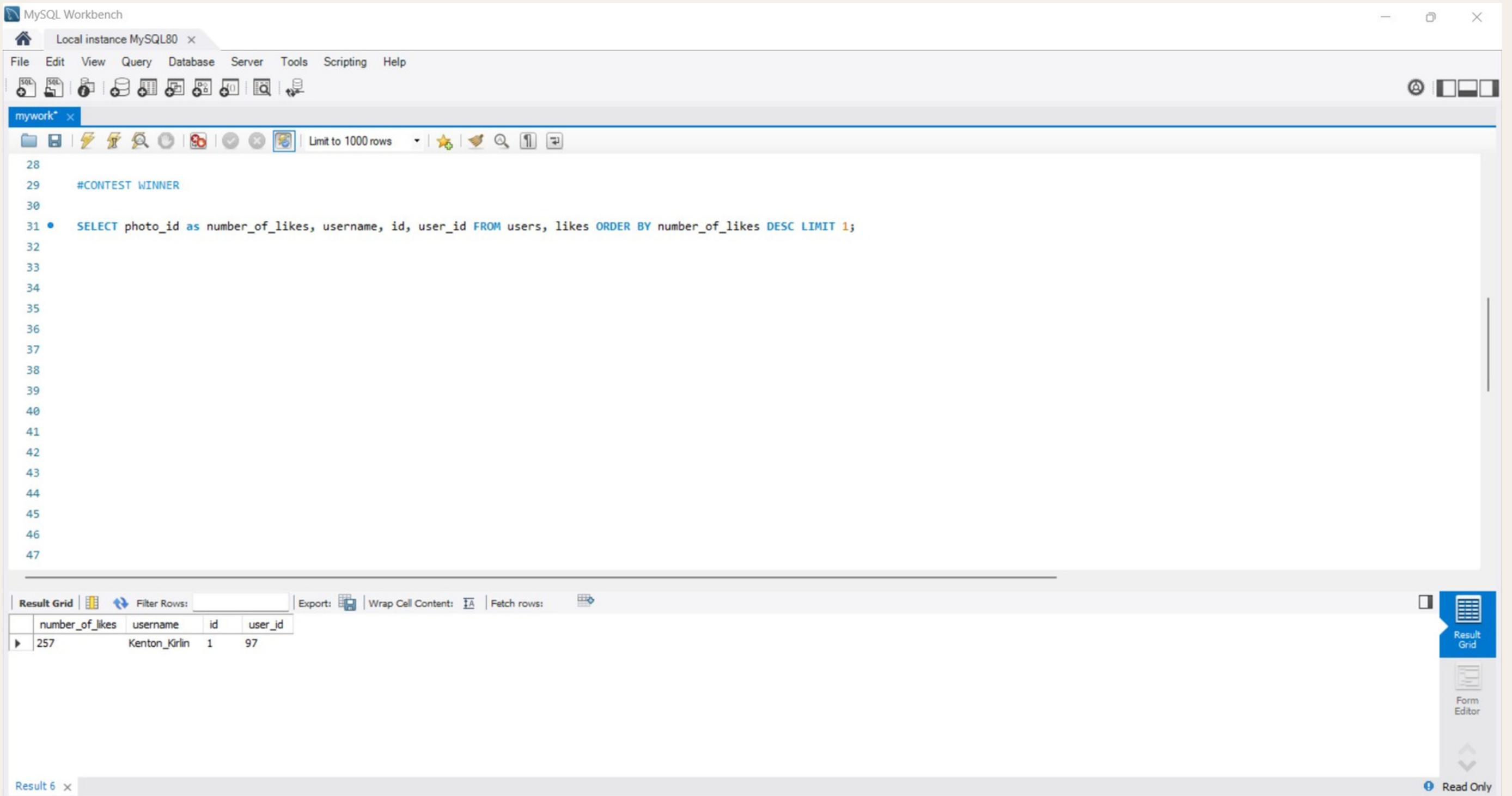
Based upon my observation from the database it has been observed that there are no inactive users in the Instagram Platform. So, team doesn't require to send any promotional email as all are the active users present here. For that reason only when SQL Query is run it shows “**NULL**” value.

Data

Analytics

• Contest Winner Declaration

The team has organized a contest where the user with the most likes on a single photo wins. So, I have to determine the winner of the contest and provide their details to the team.



MySQL Workbench
Local instance MySQL80 x

File Edit View Query Database Server Tools Scripting Help

mywork* x

28
29 #CONTEST WINNER
30
31 • SELECT photo_id as number_of_likes, username, id, user_id FROM users, likes ORDER BY number_of_likes DESC LIMIT 1;
32
33
34
35
36
37
38
39
40
41
42
43
44
45
46
47

Result Grid | Filter Rows: Export: Wrap Cell Content: Fetch rows: Result Grid

number_of_likes	username	id	user_id
257	Kenton_Krlin	1	97

Result 6 x Read Only

Here's the user with most no. of likes in a single photo.

Data

Analytics

• Contest Winner Declaration

Insights

Winner of the contest with most number of likes here is:

- Username: Kenton_Kirlin
- Account ID: 1
- Total no. of likes in single post : 257

So, here the winner for most no. of likes in a single photo i.e. **Kenton Kirlin**.

Data

Analytics

• Hashtag Research

A partner brand wants to know the most popular hashtags to use in their posts to reach the most people. So, team want to know the top five most commonly used hashtags on the platform.

The screenshot shows the MySQL Workbench interface. The query editor window contains the following SQL code:

```
40
41 • SELECT tag_name AS hashtag, COUNT(pt.tag_id) AS usage_count
42   FROM photo_tags pt JOIN tags ON pt.tag_id = tag_id
43 GROUP BY tag_name ORDER BY usage_count DESC LIMIT 5;
44
45
46
47
48
49
50
51
52
53
54
55
56
57
58
59
```

The result grid displays the following data:

hashtag	usage_count
sunset	501
sunrise	501
style	501
stunning	501
smile	501

Here's the
top five
most used
hashtags on
the
Instagram.

Data

Analytics

• Hashtag Research

Insights

The SQL Query used to find the top five hashtag which are used for the post is mentioned in the screenshot in previous slide.

The top five #hashtags which where used mainly are:

- **Sunset**
- **Sunrise**
- **Style**
- **Stunning**
- **Smile**

All the #hashtags where used for same number of times i.e. 501 times.

Data

Analytics

• Ad Campaign Launch

The team wants to know the best day of the week to launch ads. So, they want to know the day of the week when most users register on Instagram for scheduling the ad campaign.

The screenshot shows the MySQL Workbench interface. The Query editor contains the following SQL code:

```
8
9  #AD CAMPAIGN
10
11 • SELECT created_at as Most_Registers FROM USERS ORDER BY created_at DESC LIMIT 1;
```

The Result Grid shows the output of the query:

Most_Registers
2017-05-04 16:32:16

The Schemas pane shows the database structure, including the `ig_clone` schema which contains tables like `comments`, `follows`, `likes`, `photo_tags`, `photos`, `tags`, and `users`.

The bottom pane displays the Action Output log:

#	Time	Action	Message	Duration / Fetch
7	17:56:51	SELECT * FROM USERS LIMIT 0, 1000	700 row(s) returned	0.000 sec / 0.000 sec
8	18:03:55	SELECT created_at as Mostregisterday FROM USERS ORDER BY created_at DESC LIMIT 1	1 row(s) returned	0.000 sec / 0.000 sec
9	18:04:14	SELECT created_at as Mostregisterday FROM USERS ORDER BY created_at DESC LIMIT 0, 1000	700 row(s) returned	0.000 sec / 0.000 sec
10	18:04:27	SELECT * FROM USERS ORDER BY created_at LIMIT 0, 1000	700 row(s) returned	0.000 sec / 0.016 sec
11	18:10:00	SELECT * FROM USERS ORDER BY created_at DESC LIMIT 1	1 row(s) returned	0.000 sec / 0.000 sec
12	18:11:05	SELECT created_at as Most_Registers FROM USERS ORDER BY created_at DESC LIMIT 1	1 row(s) returned	0.000 sec / 0.000 sec

*The day on
which most
number of
users
registered
on the
Instagram.*

Data

Analytics

- Ad Campaign Launch

Insights

The SQL Query to find the week of the day when most of the users registered is mentioned in the previous slide screenshot.

The day of the week when most users registered is **2017-05-04 16:32:16**. Ad Campaign by the team can be scheduled in the given date so that more users can be engaged with that campaign.

Data

Analytics

03 – Investor Metrics

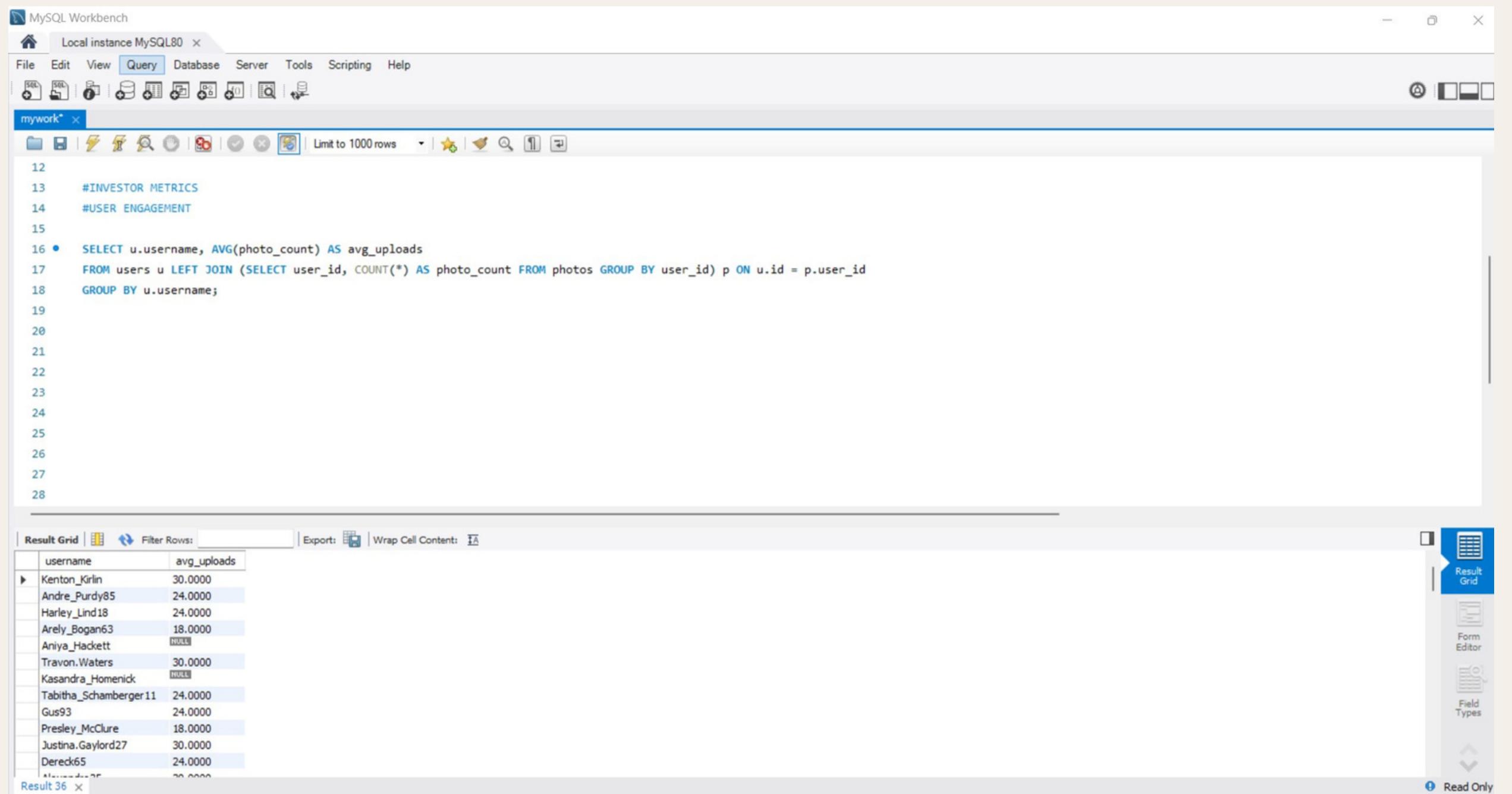
- User Engagement.
- Bots & Fake Accounts.



*Data analysis allows
for identifying
trends and patterns
within datasets.*

• User Engagement

Investors want to know if users are still active and posting on Instagram or if they are making fewer posts. So they wants me to calculate the average number of posts per user on Instagram and also to provide the total number of photos on Instagram divided by the total number of users.



The screenshot shows the MySQL Workbench interface. The query editor window contains the following SQL code:

```
12
13  #INVESTOR METRICS
14  #USER ENGAGEMENT
15
16 • SELECT u.username, AVG(photo_count) AS avg_uploads
17   FROM users u LEFT JOIN (SELECT user_id, COUNT(*) AS photo_count FROM photos GROUP BY user_id) p ON u.id = p.user_id
18   GROUP BY u.username;
19
20
21
22
23
24
25
26
27
28
```

The result grid displays the following data:

username	avg_uploads
Kenton_Kirlin	30.0000
Andre_Purdy85	24.0000
Harley_Lind18	24.0000
Arely_Bogan63	18.0000
Aniya_Hackett	NULL
Travon_Waters	30.0000
Kassandra_Homenick	NULL
Tabitha_Schamberger11	24.0000
Gus93	24.0000
Presley_McClure	18.0000
Justina_Gaylord27	30.0000
Derek65	24.0000
Alexander25	20.0000

Result Grid | Filter Rows: Export: Wrap Cell Content: Read Only

Here's the
average
number of
posts per
user on
Instagram

Data

Analytics

• User Engagement

Insights

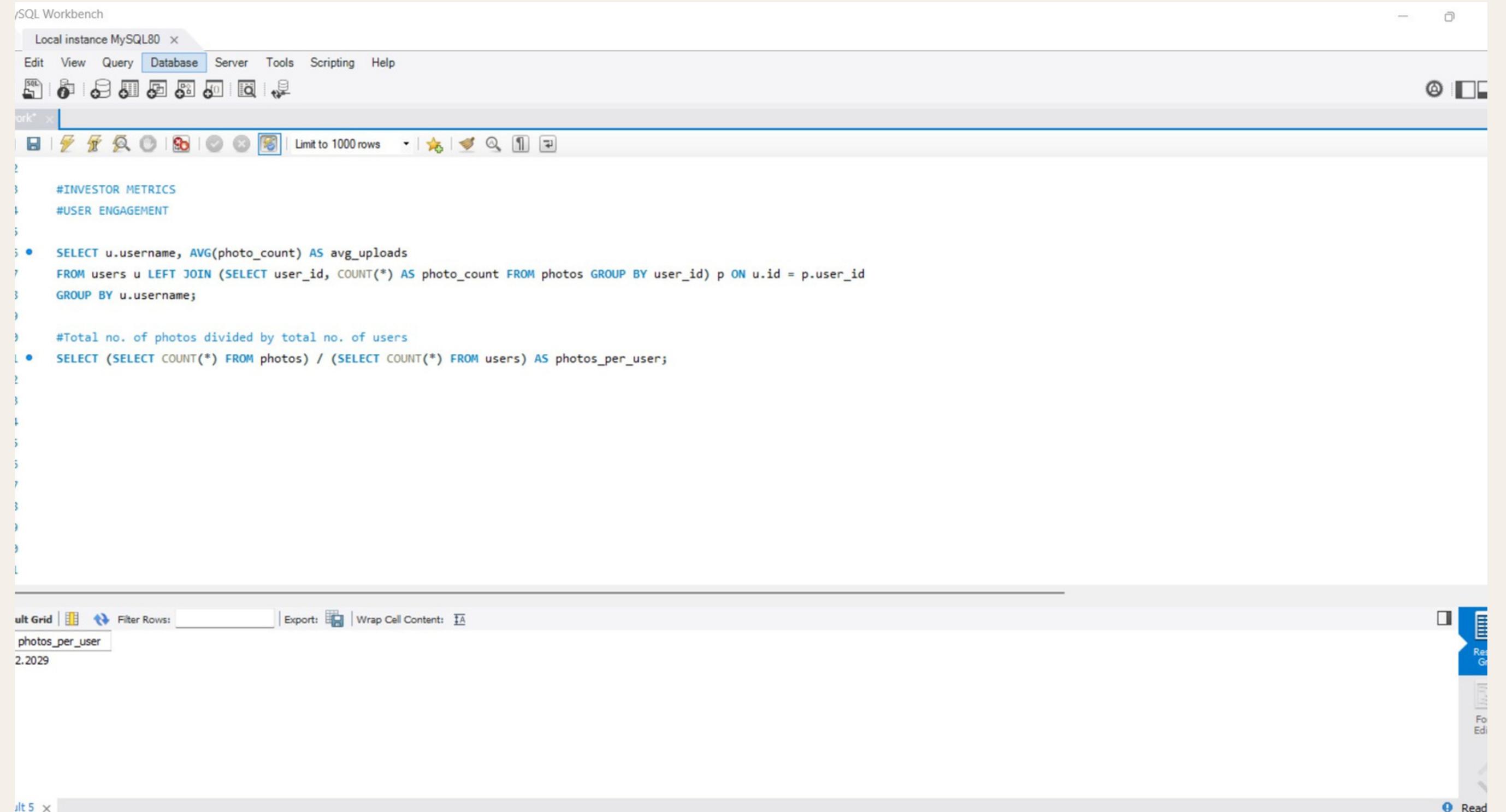
The SQL Query to calculate the average number of posts per user on Instagram is given in the screenshot of previous slide.

The average number of uploads per user is mentioned herein which ranges between 6 to 72 posts per user which varies user-to-user.

Data

Analytics

• User Engagement



The screenshot shows the MySQL Workbench interface with two tabs visible: 'Query' and 'Results'. The 'Query' tab contains the following SQL code:

```
#INVESTOR METRICS
#USER ENGAGEMENT

• SELECT u.username, AVG(photo_count) AS avg_uploads
  FROM users u LEFT JOIN (SELECT user_id, COUNT(*) AS photo_count FROM photos GROUP BY user_id) p ON u.id = p.user_id
  GROUP BY u.username;

• #Total no. of photos divided by total no. of users
• SELECT (SELECT COUNT(*) FROM photos) / (SELECT COUNT(*) FROM users) AS photos_per_user;
```

The 'Results' tab displays the output of the second query:

photos_per_user
2.2029

Here's the total number of photos on Instagram divided by the total number of users.

Data

Analytics

• User Engagement Insights

The SQL Query to calculate the total number of photos divided by total number of users on Instagram is given in the screenshot of previous slide to find the photos per user.

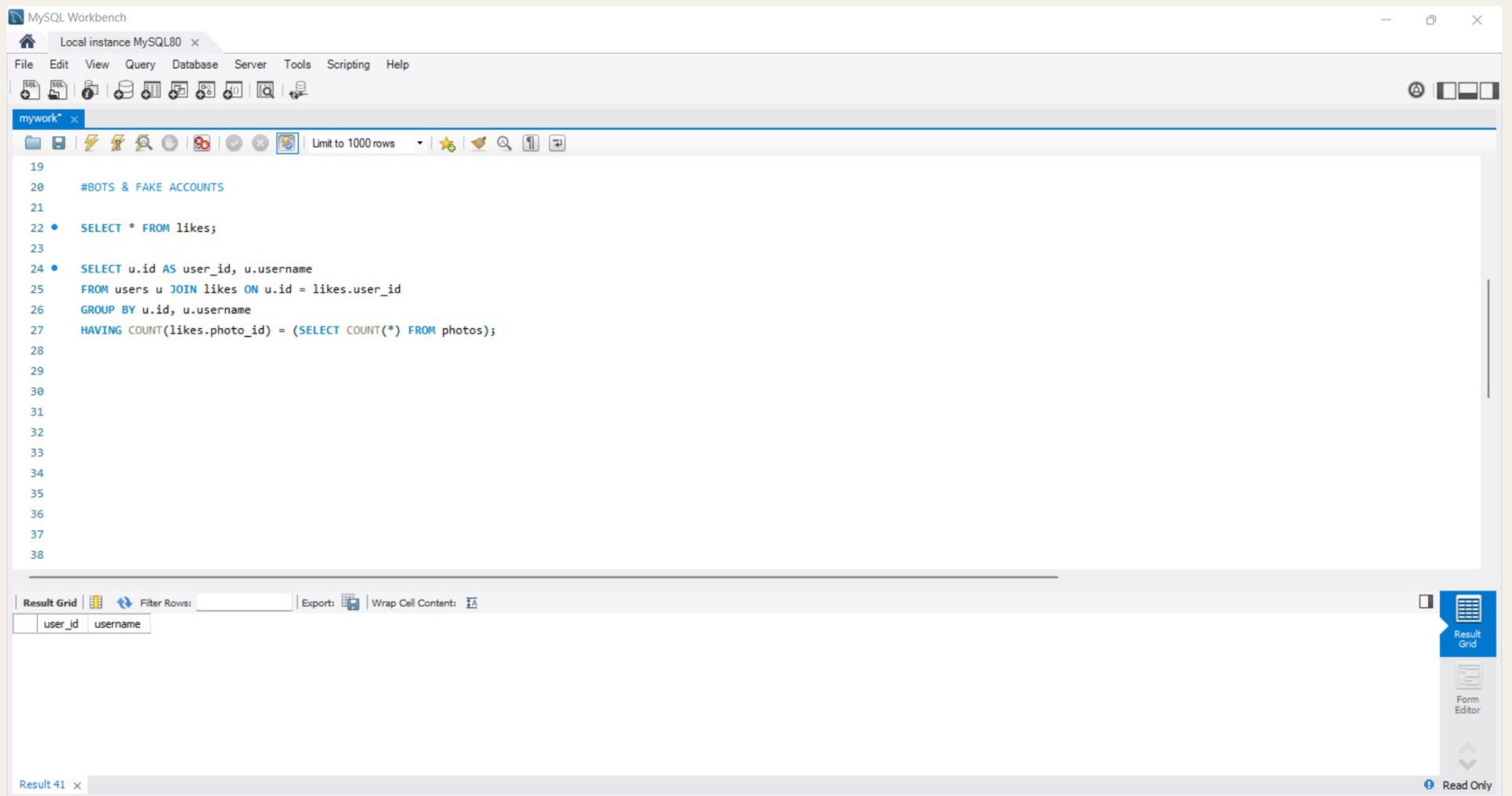
The number of photos per user is **2.2029**.

Data

Analytics

• Bots & Fake Accounts

Investors want to know if the platform is crowded with fake and dummy accounts. So, wants me to identify users (potential bots) who have liked every single photo on the site, as this is not typically possible for a normal user.



The screenshot shows the MySQL Workbench interface. The query editor window contains the following SQL code:

```
19
20  #BOTS & FAKE ACCOUNTS
21
22 • SELECT * FROM likes;
23
24 • SELECT u.id AS user_id, u.username
25   FROM users u JOIN likes ON u.id = likes.user_id
26   GROUP BY u.id, u.username
27   HAVING COUNT(likes.photo_id) = (SELECT COUNT(*) FROM photos);
28
29
30
31
32
33
34
35
36
37
38
```

The result grid below the query editor is empty, with the header row showing "user_id" and "username". A vertical sidebar on the right contains icons for Result Grid, Form Editor, and a "Read Only" status indicator.

Here's the output for bots & fake accounts. As all the users are genuine so the output is blank.

Data

Analytics

• Bots & Fake Accounts

Insights

The SQL Query to find where fake & dummy accounts are present in the platform is given in the screenshot in previous slide.

After running the query there is no output is been displayed in user_id and username column.

By which we can conclude that there is no fake & dummy accounts in the platform i.e. all the accounts are of real users.

..

Data

Analytics

Thanks



Mukesh Dey