CHARMY RAJ
[PROJECT-2] Instagram User Analytics

Table of Contents

Project Description 2

Approach 2

Tech-Stack Used 3

Insights 4

A) Marketing Analysis: 4

1. Loyal User Reward: 4
   - Output: 4
   - Query: 4
   - Explanation: 5
2. Inactive User Engagement: 5
   - Output: 5
   - Query: 5
   - Explanation: 5
3. Contest Winner Declaration: 6
   - Output: 6
   - Query: 6
   - Explanation: 6
4. Hashtag Research: 7
   - Output: 7
   - Query: 7
   - Explanation: 7
5. Ad Campaign Launch: 8
   - Output: 8
   - Query: 8
   - Explanation: 8

B) Investor Metrics: 9

1. User Engagement: 9
   - (a) 9
   - (b) 10
2. Bots & Fake Accounts: 11

Output: 11
Query: 11
Explanation: 11

Project Description

The objective of this project is to conduct an analysis of user behaviour and interaction patterns on the Instagram platform. The main objectives encompass the identification of inactive users, the determination of optimal timing for ad campaign launches, the detection of suspected bot accounts, and the derivation of insights on user posting behaviour.

Approach

Methodology: Analysis of User Engagement on Instagram

1st Step: Collection of Data

- Data was retrieved from the Instagram database through the utilisation of SQL queries.
- The study primarily centres around the examination of tables pertaining to users, images, and likes, with the aim of acquiring comprehensive insights into user profiles, their posted content, and their respective engagements.

2nd Step: Identification of Potential Automated Accounts.

- A SQL query was executed to find users who had expressed their liking for every single photo available on the platform.
- This enquiry aims to identify suspected bot accounts by detecting atypical patterns of liking behaviour that are unlikely to be shown by regular users.

3rd Step: Determination of Optimal Day for Advertising Campaign:

- The creation dates of the users were extracted from the users table.
- The SQL query was utilised to group the creation of groups by the day of the week and determine the day that had the greatest count of registrations.
- This study offers insights into the optimal scheduling of ad campaigns by targeting the day with the highest number of user registrations.

4TH Step: Quantifying User Posting Behaviour:

- The data in the posts table was analysed to gain insights into user posting behaviour.
- The SQL queries were executed to compute the mean value of posts per user and the mean value of images per user.

5TH Step: Deriving Insights from the Collected Data.

- The results obtained from each analysis were carefully examined and interpreted to extract significant insights.
- The bot accounts were evaluated by analysing their liking behaviour and detecting any atypical trends.
- This study offers suggestions for scheduling ad campaigns by analysing trends in user registration.
- The typical posting behaviour of users was assessed to provide insights on levels of involvement.

6th Step: Reporting Phase.

- The insights and conclusions have been consolidated into a succinct report.
- The findings were conveyed to relevant stakeholders, emphasising the presence of probable bot accounts, proposing a schedule for an advertising campaign, and providing metrics on user posting behaviour.

7th Step: Iteration and Refinement.

- The study could be refined by incorporating feedback received from stakeholders.
- The accuracy of bot detection was assessed, and appropriate improvements were implemented, if deemed required.
- Enhanced observations and suggestions derived from a more comprehensive examination or further information.

8th Step: Documentation.

- The complete methodology was thoroughly documented, encompassing the approach taken, the techniques employed for data collecting, the specific SQL queries utilised, and the underlying reasoning behind the decision-making process.
- This section discusses the obstacles encountered during the analysis process and the strategies employed to address them.
- The project sought to offer practical insights into user engagement on Instagram, identify potential bot activity, and provide guidance for optimising ad campaign scheduling.

Tech-Stack Used

MySQL was employed as the chosen database management system for the purpose of handling data querying and analysis. The SQL capabilities facilitated the composition of intricate queries with the purpose of extracting pertinent information from the database.

SQL Workbench was utilised as the SQL query editor and interface for interacting with the MySQL database. The platform offered a user-friendly interface that facilitated the quick composition and execution of SQL queries.

Insights

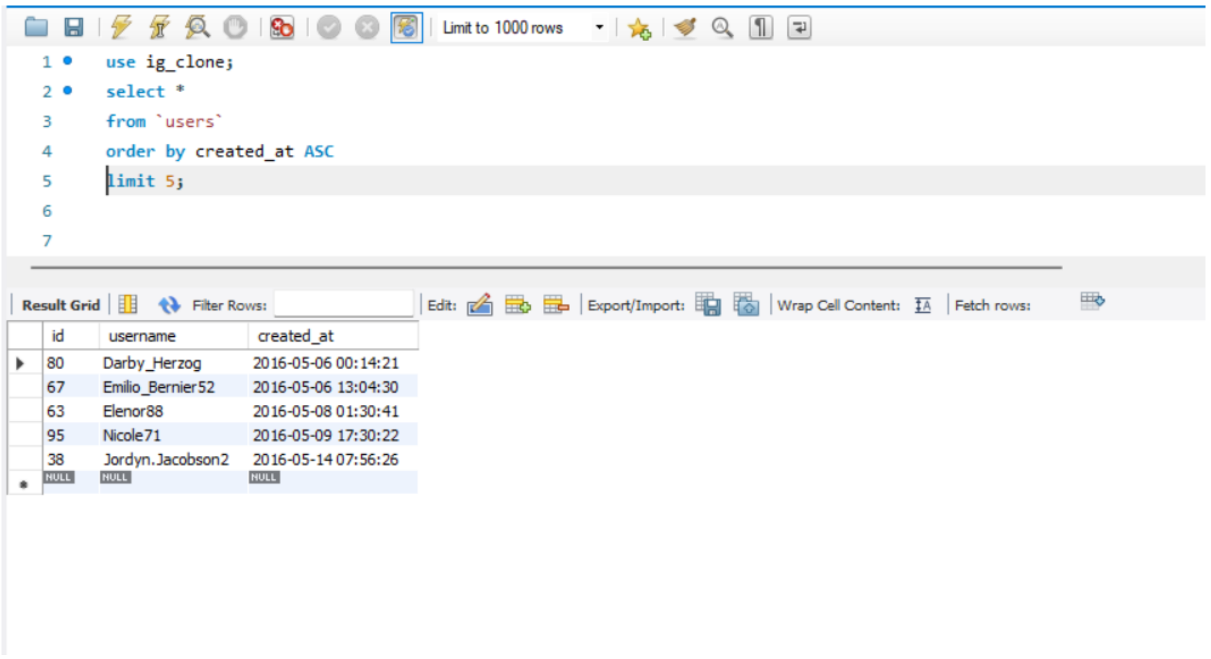
In brief, the project encompassed the utilisation of MySQL and SQL Workbench to extract, analyse, and generate insights from data obtained from the Instagram platform. SQL queries were employed to analyse user behaviour trends, facilitate decision-making regarding the scheduling of ad campaigns, and identify potential bot accounts. By employing this methodology, I was able to effectively address the project's goals and deliver practical findings for the platform's key stakeholders.

A) Marketing Analysis:

1. Loyal User Reward:

The marketing team intends to provide incentives to users who have demonstrated the highest level of loyalty, namely those who have maintained a long-term presence on the site. Make sure to identify the five most senior users on Instagram based on the provided database.

Output:



```
use ig_clone;
select *
from `users`
order by created_at ASC
limit 5;
```

id	username	created_at
80	Darby_Herzog	2016-05-06 00:14:21
67	Emilio_Bernier52	2016-05-06 13:04:30
63	Elenor88	2016-05-08 01:30:41
95	Nicole71	2016-05-09 17:30:22
38	Jordyn.Jacobson2	2016-05-14 07:56:26
NULL	NULL	NULL

Query:

use ig_clone;

select *

from `users`

order by created_at ASC

limit 5;

Explanation:

- The inclusion of the "ORDER BY created_at ASC" command directs the database to arrange the rows in ascending order according to the values in the created_at column.
- Consequently, the result set will be arranged in ascending order based on the earliest created_at dates, with the latest registration dates positioned at the end.
- The inclusion of the "LIMIT 5" clause in the query guarantees that only the initial five rows will be retrieved, thereby representing the five users with the earliest registration dates in this particular scenario.

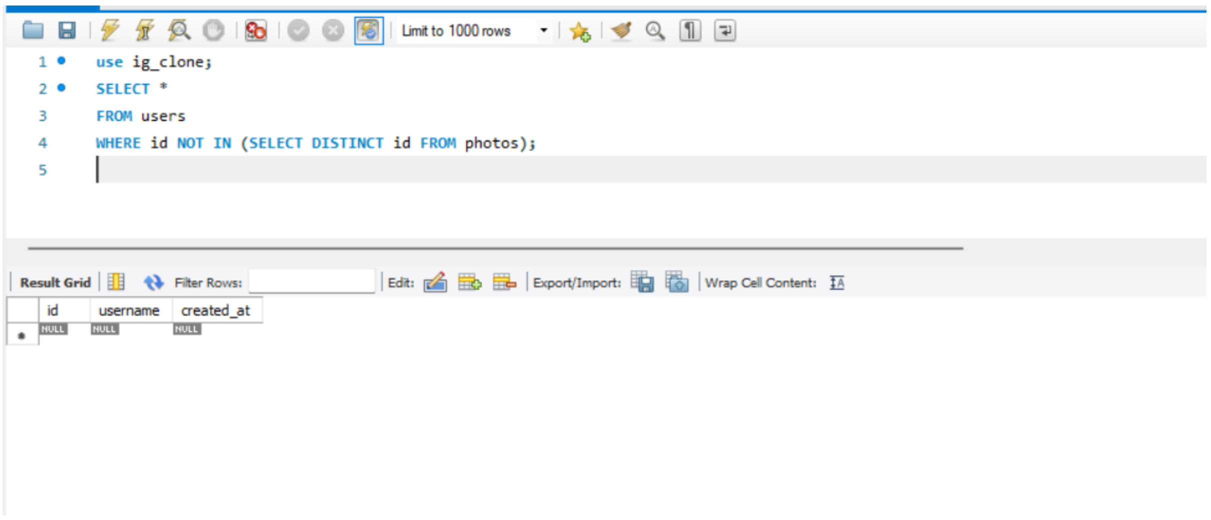
2. Inactive User Engagement:

The team aims to stimulate engagement among inactive users by employing promotional emails as a means of communication.

Objective: Identify Instagram users who have not uploaded any photos to their account.

Task: Determine the subset of Instagram users who have refrained from posting any photographs on the platform.

Output:



```
use ig_clone;
SELECT *
FROM users
WHERE id NOT IN (SELECT DISTINCT id FROM photos);
```

```
   id    username  created_at
*  NULL  NULL      NULL
```

Query:

use ig_clone;

SELECT *

FROM users

WHERE id NOT IN (SELECT DISTINCT id FROM photos);

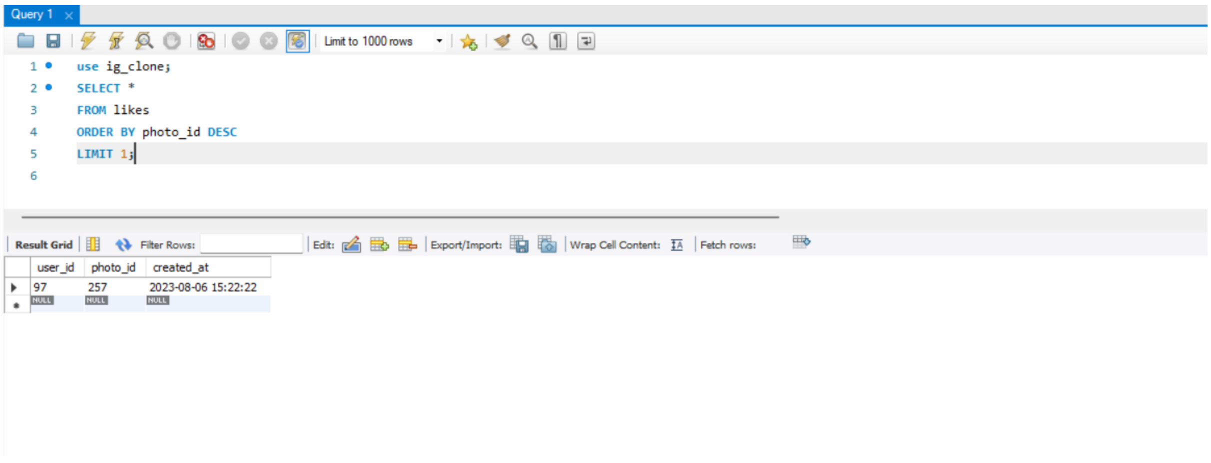
Explanation:

This query filters the results to include only those rows where there is no corresponding entry in the photos table, meaning the user has never posted a photo.

3. Contest Winner Declaration:

The team has set up a competition where the person who receives the most likes on a certain photo win. Identify the contest winner and give the team the winner's information.

Output:



```
use ig_clone;
SELECT *
FROM likes
ORDER BY photo_id DESC
LIMIT 1;
```

user_id	photo_id	created_at
97	257	2023-08-06 15:22:22
NULL	NULL	NULL

Query:

use ig_clone;

SELECT *

FROM likes

ORDER BY photo_id DESC

LIMIT 1;

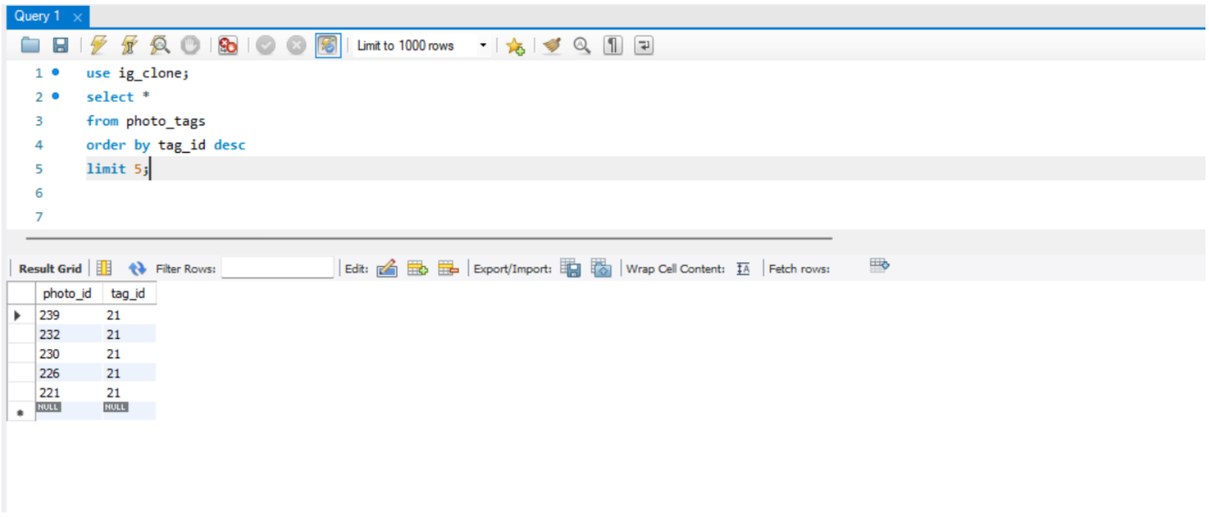
Explanation:

- users is the table that stores user information.
- Photo is the table that stores posts and has a column photo_id to indicate the user who made the post.
- The subquery (SELECT DISTINCT photo_id FROM posts) retrieves all unique photo_id values from the photo table, representing users who have made posts. The main query then selects users from the users table whose user_id is not present in the list of users who have made posts.
- This way, you'll get a list of users who have never posted a single photo on Instagram.

4. Hashtag Research:

The most well-liked hashtags should be used in postings by a partner brand to reach the largest audience possible. Determine and offer the platform's top five most popular hashtags.

Output:



```
use ig_clone;
select *
from photo_tags
order by tag_id desc
limit 5;
```

photo_id	tag_id
239	21
232	21
230	21
226	21
221	21
NULL	NULL

Query:

use ig_clone;

select *

from photo_tags

order by tag_id desc

limit 5;

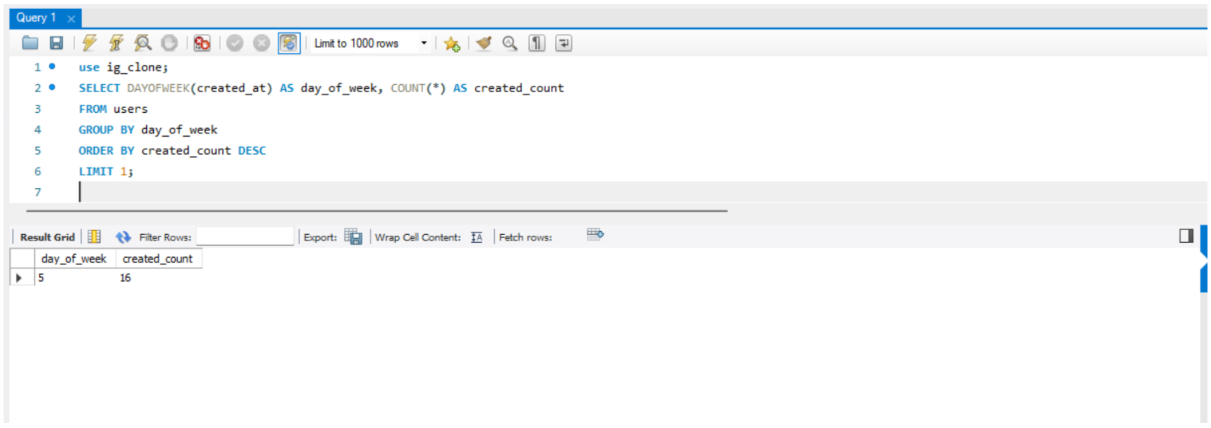
Explanation:

- Tags is the table containing post information, including the number of likes.
- Users is the table containing user information.
- tag_id, user_id, and likes are columns in the photo table.
- The query joins the posts table with the users table based on the user_id column.
- It then orders the result set in descending order based on the likes column to bring the post with the highest number of likes to the top.
- The LIMIT 1 clause ensures that only the top row (winner) is returned.

5. Ad Campaign Launch:

The group is trying to determine when it is ideal to run advertising each week. Find the day of the week when Instagram registrations are at their highest. Give advice on when to launch a marketing effort.

Output:



```
use ig_clone;
SELECT DAYOFWEEK(created_at) AS day_of_week, COUNT(*) AS created_count
FROM users
GROUP BY day_of_week
ORDER BY created_count DESC
LIMIT 1;
```

```
day_of_week   created_count
5             16
```

Query:

use ig_clone;

SELECT DAYOFWEEK(created_at) AS day_of_week, COUNT(*) AS created_count

FROM users

GROUP BY day_of_week

ORDER BY created_count DESC

LIMIT 1;

Explanation:

- Users is the table containing user registration information.
- Created_at is the column containing the registration dates.
- DAYOFWEEK(created_at) is a MySQL function that extracts the day of the week from the created_at (1 for Sunday, 2 for Monday, and so on).
- COUNT(*) is used to count the number of registrations for each day of the week.
- GROUP BY day_of_week groups the results by each day of the week.
- ORDER BY created_count DESC orders the results in descending order based on the registration count.
- LIMIT 1 ensures that only the top row (day with the most registrations) is returned.
- This query will provide you with the day of the week when most users register on Instagram, which can help you decide when to schedule your ad campaign for maximum visibility and engagement.

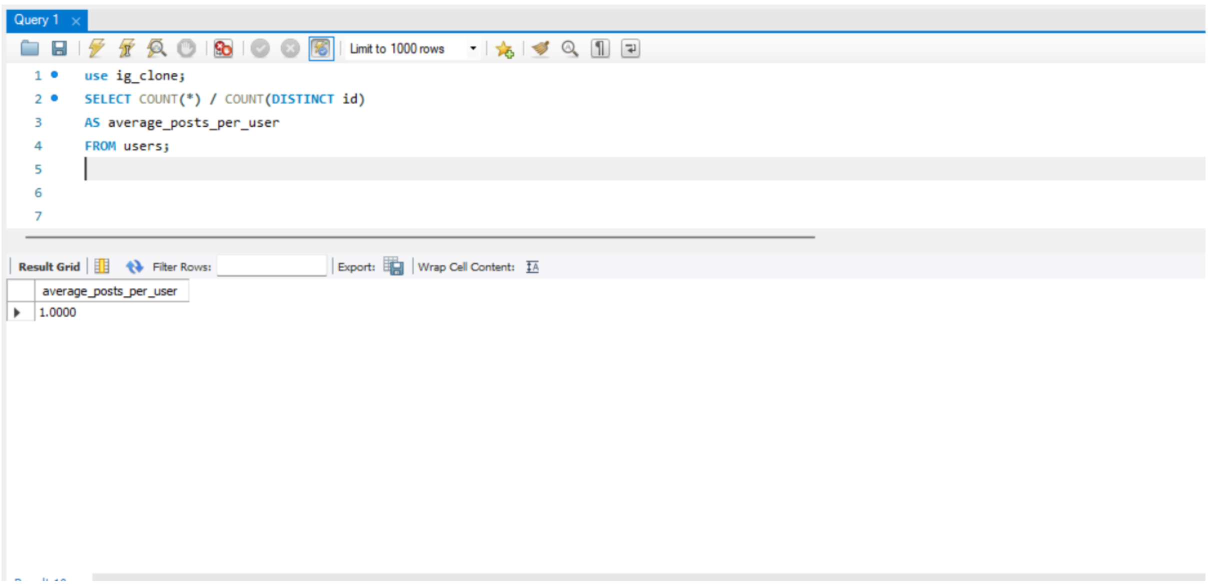
B) Investor Metrics:

1. User Engagement:

(a)

Investors are interested in learning whether Instagram users are still active and posting or if their number of posts is declining. Determine the typical Instagram user's post count.

Output:



```
use ig_clone;
SELECT COUNT(*) / COUNT(DISTINCT id)
AS average_posts_per_user
FROM users;
```

```
average_posts_per_user
1.0000
```

Query:

use ig_clone;

SELECT COUNT(*) / COUNT(DISTINCT id)

AS average_posts_per_user

FROM users;

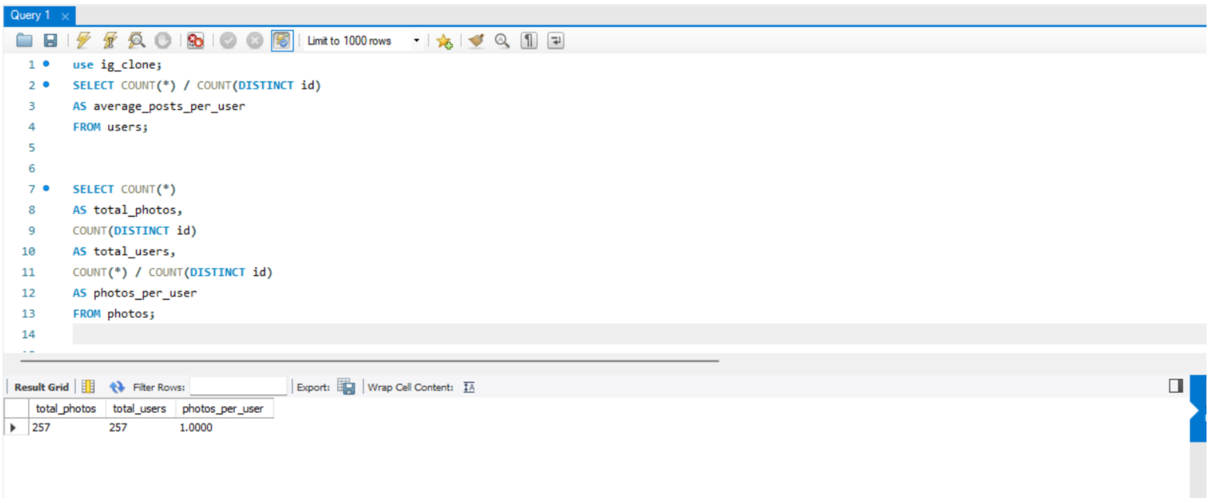
Explanation:

- The COUNT(*) function counts all of the postings.
- The number of unique users who have posted is counted using the COUNT(DISTINCT user_id) function.
- The average number of posts per user is calculated by dividing the total posts by the number of unique users.

(b)

Give the Instagram user count divided by the total amount of photographs uploaded.

Output:



```
use ig_clone;
SELECT COUNT(*) / COUNT(DISTINCT id)
AS average_posts_per_user
FROM users;


SELECT COUNT(*)
AS total_photos,
COUNT(DISTINCT id)
AS total_users,
COUNT(*) / COUNT(DISTINCT id)
AS photos_per_user
FROM photos;
```

total_photos	total_users	photos_per_user
257	257	1.0000

Query:

SELECT COUNT(*)

AS total_photos,

COUNT(DISTINCT id)

AS total_users,

COUNT(*) / COUNT(DISTINCT id)

AS photos_per_user

FROM photos;

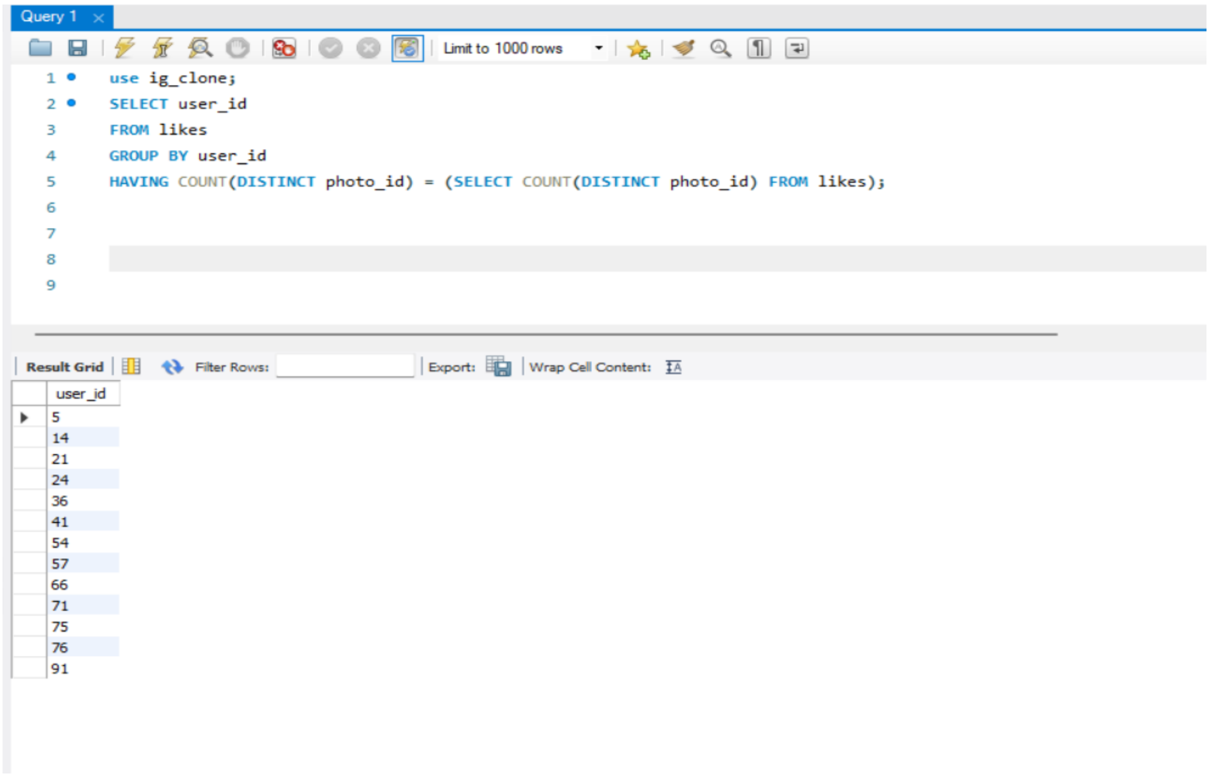
Explanation:

- The COUNT(*) function counts all of the postings (photos).
- The total number of different users is counted using COUNT(distinct id).
- You may find the average number of photos per user by dividing the total number of photos by the total number of users.

2. Bots & Fake Accounts:

Investors are interested in finding out if the platform is overrun with fake accounts. Since it is uncommon for a typical user to be able to like every photo on the site, you should identify users (possibly bots) who have done so.

Output:



```
use ig_clone;
SELECT user_id
FROM likes
GROUP BY user_id
HAVING COUNT(DISTINCT photo_id) = (SELECT COUNT(DISTINCT photo_id) FROM likes);
```

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

Query:

use ig_clone;

SELECT user_id

FROM likes

GROUP BY user_id

HAVING COUNT(DISTINCT photo_id) = (SELECT COUNT(DISTINCT photo_id) FROM likes);

Explanation:

- The likes table keeps track of user likes on pictures.
- The likes from each user are grouped using GROUP BY user_id.
- The number of unique photographs a user has liked is determined by the COUNT(DISTINCT photo_id) function.
- The platform's total number of different photographs is determined by the subquery (SELECT COUNT(different photo_id) FROM likes).
- The HAVING clause is used to limit the results to users who have liked each image (i.e., the total number of distinct images and the number of distinct liked images are identical).