Project Description

I contributed to a project that mimics the work of a data analyst at Instagram with emphasis on how users engage with the app. The objective was to drill down into user behavior and pull out insights that could inform various teams - ranging from marketing to product development - to make better decisions. It wasn't about SQL query writing; it was about understanding how data can inform actual business strategies and enhance user experience.

Approach

I took each business question and broke it into easier pieces, thinking about who asked it, what would they really want to know? I used MySQL to answer all the questions while knowing that what exact details they need. Moreover, I verified my query was not only technically correct but also told a story and actionable advice to the team. It was like cracking mini-mysteries with each question.

Tech Stack used

My main tool was MySQL Workbench—clean interface and easy to test queries. I've practiced with this tool from starting, so it is comfortable to use, thus saving time as I already know how to use it. Also I think providing right and accurate insights is more important than which tech stack is being used, provided that the particular software is not obsolete and is trusted.

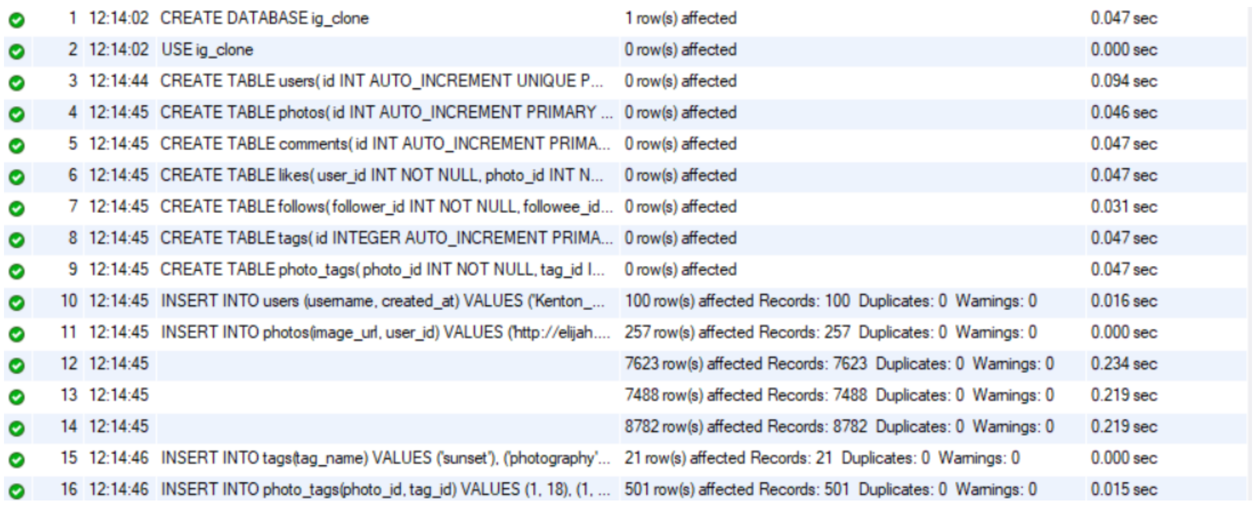
Insights

The experience of this project highlighted how data informs product decisions. One of the insights that I discovered was my ability to identify peak times of user engagement, as well as better understand what type of content was driving the most interactions. I also noticed indications in user retention when activity levels were categorized. What was fascinating is the findings, while they may appear small, still lead to a potential positive shift in actual user experience, and feature planning. Although my decisions on not taken serious, but after this I'm pretty confident that I can provide valuable information as well.

Results

Below is the desired output of all the questions that demands high accuracy and completeness.

Firstly, I've created the database and used it along with inserting all the provided data , with output.

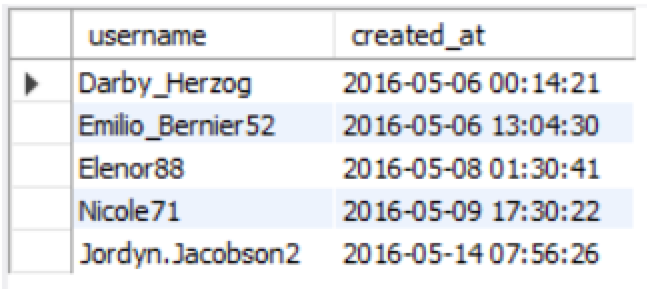


```
1   12:14:02  CREATE DATABASE ig_clone                                      1 row(s) affected                                              0.047 sec
2   12:14:02  USE ig_clone                                                  0 row(s) affected                                              0.000 sec
3   12:14:44  CREATE TABLE users( id INT AUTO_INCREMENT UNIQUE P...         0 row(s) affected                                              0.094 sec
4   12:14:45  CREATE TABLE photos( id INT AUTO_INCREMENT PRIMARY ...        0 row(s) affected                                              0.046 sec
5   12:14:45  CREATE TABLE comments( id INT AUTO_INCREMENT PRIMA...         0 row(s) affected                                              0.047 sec
6   12:14:45  CREATE TABLE likes( user_id INT NOT NULL, photo_id INT N...   0 row(s) affected                                              0.047 sec
7   12:14:45  CREATE TABLE follows( follower_id INT NOT NULL, followee_id...  0 row(s) affected                                            0.031 sec
8   12:14:45  CREATE TABLE tags( id INTEGER AUTO_INCREMENT PRIMA...         0 row(s) affected                                              0.047 sec
9   12:14:45  CREATE TABLE photo_tags( photo_id INT NOT NULL, tag_id I...   0 row(s) affected                                              0.047 sec
10  12:14:45  INSERT INTO users (username, created_at) VALUES ('Kenton_...  100 row(s) affected Records: 100  Duplicates: 0  Warnings: 0   0.016 sec
11  12:14:45  INSERT INTO photos(image_url, user_id) VALUES ('http://elijah....  257 row(s) affected Records: 257  Duplicates: 0  Warnings: 0  0.000 sec
12  12:14:45                                                                7623 row(s) affected Records: 7623  Duplicates: 0  Warnings: 0  0.234 sec
13  12:14:45                                                                7488 row(s) affected Records: 7488  Duplicates: 0  Warnings: 0  0.219 sec
14  12:14:45                                                                8782 row(s) affected Records: 8782  Duplicates: 0  Warnings: 0  0.219 sec
15  12:14:46  INSERT INTO tags(tag_name) VALUES ('sunset'), ('photography'...  21 row(s) affected Records: 21  Duplicates: 0  Warnings: 0   0.000 sec
16  12:14:46  INSERT INTO photo_tags(photo_id, tag_id) VALUES (1, 18), (1, ...  501 row(s) affected Records: 501  Duplicates: 0  Warnings: 0  0.015 sec
```

1. MySql query for Marketing Analysis

a) select username, created_at from users order by created_at asc limit 5;

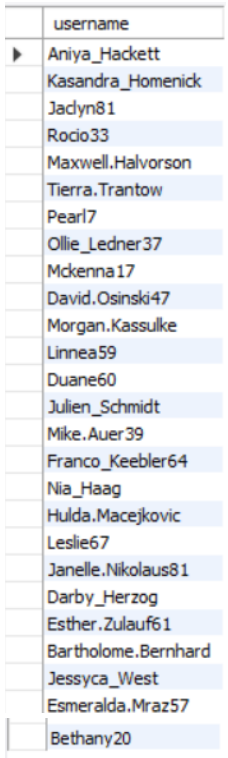
- asked – 5 oldest users, action – selecting users whose registrations are oldest with limit of 5 and for more efficiency, in ascending order.



```
username            created_at
Darby_Herzog        2016-05-06 00:14:21
Emilio_Bernier52    2016-05-06 13:04:30
Elenor88            2016-05-08 01:30:41
Nicole71            2016-05-09 17:30:22
Jordyn.Jacobson2    2016-05-14 07:56:26
```

b) select users.username from users left join photos on users.id = photos.user_id where photos.id is null;

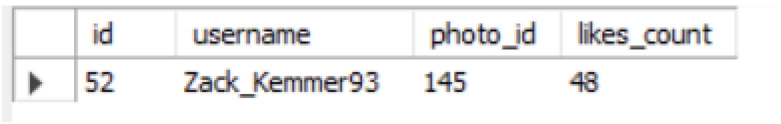
- asked – users who have never posted single photo, action – selecting user ids in photos whose value it null left joining on users id and photos user id, ensuring all ids are covered.



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
Jessyca_West
Esmeralda.Mraz57
Bethany20

c) select users.id, users.username, photos.id as photo_id, count(likes.user_id) as likes_count from photos
join users on photos.user_id = users.id
join likes on likes.photo_id = photos.id
group by photos.id
order by likes_count desc
limit 1;

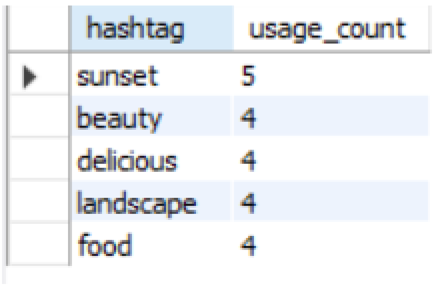
- asked – user with most likes on single photo, action – selecting username, photo id and count (likes), joining on users and likes ensuring there is no ambiguity and we get accurate result, also winner is only one so limit = 1.



id	username	photo_id	likes_count
52	Zack_Kemmer93	145	48

d) select tags.tag_name as hashtag, count(photo_tags.photo_id) as usage_count from tags join photo_tags on tags.id = photo_tags.photo_id group by tags.tag_name order by usage_count desc limit 5;

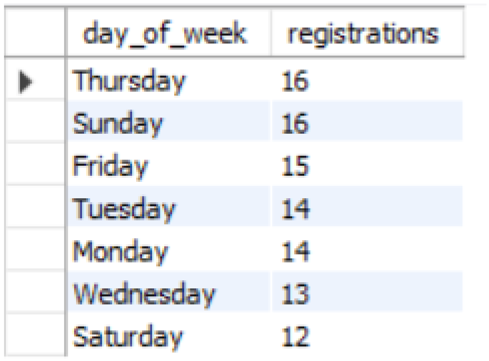
- asked – five most used hashtags, action – selecting tag names (as hashtags, good practice) and count, joining on photos so that all photos and tags are covered, for 5 limit = 5 and order by (good practice).



hashtag	usage_count
sunset	5
beauty	4
delicious	4
landscape	4
food	4

e) select dayname(created_at) as day_of_week, count(*) as registrations from users group by day_of_week order by registrations desc;

- asked – day of the week when user registrations are high, action – using date function to extract date from created_at and count to show numbers.

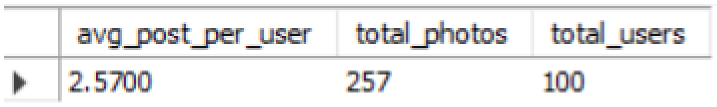


day_of_week	registrations
Thursday	16
Sunday	16
Friday	15
Tuesday	14
Monday	14
Wednesday	13
Saturday	12

2. MySql query for Investor Metrics

a) select count(photos.id) / count(distinct users.id) as avg_post_per_user, count(photos.id) as total_photos, count(distinct users.id) as total_users from users left join photos on users.id = photos.user_id;

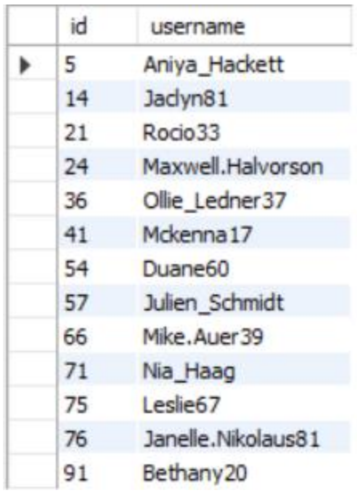
- asked – average posting of users, action – calculating average (for more efficiency round() can be used) and using distinct as to remove duplicates, left joining as all the users are covered.



avg_post_per_user	total_photos	total_users
2.5700	257	100

b) select users.id , users.username from users join likes on users.id = likes.user_id group by users.id, users.username having count(distinct likes.photo_id) = (select count(*) from photos);

- asked – dummy or fake account, action – (I didn't get it at first and after some brainstorming and searching, if from a single account every photo is being liked, which a normal human being will not do as of their likes and dislikes, there can be different approaches to this, not sure if this is most effective one) selecting username joining on users so that all accounts and their likes are covered , used subquery for matching total count, group by for grouping username for same value .



id	username
5	Aniya_Hackett
14	Jaclyn81
21	Rocio33
24	Maxwell.Halvorson
36	Ollie_Ledner37
41	Mckenna17
54	Duane60
57	Julien_Schmidt
66	Mike.Auer39
71	Nia_Haag
75	Leslie67
76	Janelle.Nikolaus81
91	Bethany20

Thank you.