INSTAGRAM USER ANALYTICS WITH MYSQL

In this document, I have included all questions of this project, respective SQL queries which I have developed to solve those and explained the my own approach to solve those.

Question 1: Find the 5 oldest users of the Instagram from the database provided

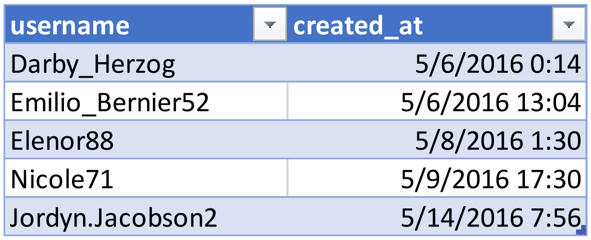
Query: select username,created_at

from users

order by created_at

limit 5;

Output:



username	created_at
Darby_Herzog	5/6/2016 0:14
Emilio_Bernier52	5/6/2016 13:04
Elenor88	5/8/2016 1:30
Nicole71	5/9/2016 17:30
Jordyn.Jacobson2	5/14/2016 7:56

Approach:

First I sorted the usernames based on their registration dates → then fetched the top 5 users using limit clause.

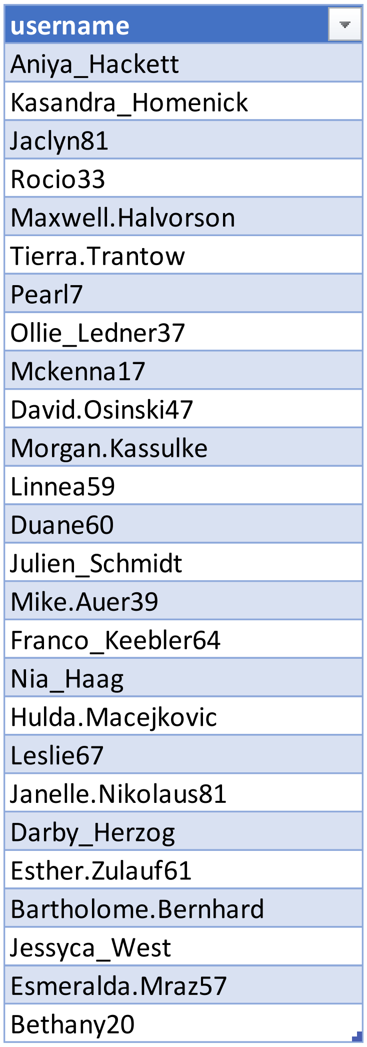
Question 2: Find the users who have never posted a single photo on Instagram

Query: select username from users

where id not in

(select user_id from photos);

Output:



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

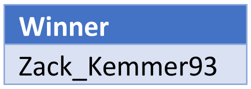
Approach: I used subqueries to find out the users who have posted photos from 'photos' table → then fetched the users who don't meet that criteria using NOT IN command.

Question 3: Identify the winner of the most liked photo contest and provide their details to the team

Query: select username Winner from users

```
where id = (select user_id from photos
        where id = (
        With highest_liked_photo as
                (select photo_id, count(user_id) from likes
                group by photo_id

                order by count(user_id) desc limit 1)

        select photo_id from highest_liked_photo));
```

Output:



Winner
Zack_Kemmer93

Approach:

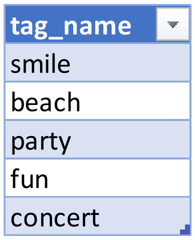
Using group by and order by clause I first found out the photo_id which got highest likes → Using CTE and subquery, I fetched the userid and name who have posted that particular photo.

Question 4: Identify and suggest the top 5 most commonly used hashtags on the platform

Query: select tag_name from tags where

```
id in (With most_used_tags as

                (select tag_id, count(photo_id) from photo_tags

                group by tag_id

                order by count(photo_id) desc limit 5)

        select tag_id from most_used_tags);
```

Output:



tag_name
smile
beach
party
fun
concert

Approach: Fetched the top 5 hashtag id which got highest number tags → Using a subquery retrieved the hashtag names of those id.

Question 5: What day of the week do most users register on? Provide insights on when to schedule an ad campaign

Query:

```
select time(created_at), weekday(created_at) Weekday from users
where weekday(created_at) in
        (With most_registered_weekdays as
                (select weekday(created_at) as weekday, count(id)  from users
                group by weekday(created_at)
                order by count(id) desc limit 2)
        select weekday from most_registered_weekdays);
```

Output:

time(created_at)	Weekday
18:22:11	3
17:11:21	6
13:26:14	6
16:25:49	6
16:32:16	3
1:34:14	3
7:15:03	3
18:51:57	6
20:28:12	3
3:25:22	3
15:42:20	3
9:51:26	3
17:25:45	6
18:04:45	6
7:57:44	3
11:40:27	3
21:23:37	6
3:10:22	6
12:42:31	6
5:58:22	6
23:12:48	3
22:03:45	3
1:30:41	6
23:08:31	3
20:09:27	6
9:26:09	3

13:14:11	3
2:31:23	6
17:44:43	6
7:50:51	6
21:40:10	3
14:57:28	6

Approach: Using group by and order by clause found that 2 weekdays has same number of registrations (3 & 6) → Rectrieved the timing of registrations using Date-time functions.

Question 6: Provide how many times does average user posts on Instagram. lso, provide the total number of photos on Instagram/total number of users

Query:

```
With count_post as
        (select user_id, count(id) as countposts from photos
        group by user_id
        order by count(id) desc)
select avg(countposts) Average_no_of_posts from count_post;
```

Output:

Average_no_of_posts
3.473

Approach: Used CTE to count the number of photos posted by each user → Fetched the average number of post which 3.473 (i.e 3-4 times)

```
select count(p.id) Total_photos, count(distinct u.id) Total_users
from photos p
right join users u
on p.user_id = u.id;
```

Output:

Total_photos	Total_users
257	100

Approach: used aggregation function and right join to fetch the output. (Inner join method was only retrieving the users who have posted at least one photo, but I needed all the unique user-ids for which right join was suitable.

Question 7: Provide data on users (bots) who have liked every single photo on the site

Query:

```
select username from users

where id in

        (With Bot_users as

                (select user_id, count(distinct photo_id) from likes

                group by user_id

                having count(distinct photo_id) = (select max(distinct photo_id) from likes))

  select user_id from Bot_users);
```

Output:

Bot_users
Aniya_Hackett
Jaclyn81
Rocio33
Maxwell.Halvorson
Ollie_Ledner37
Mckenna17
Duane60
Julien_Schmidt
Mike.Auer39
Nia_Haag
Leslie67
Janelle.Nikolaus81
Bethany20

Approach: Counted the distinct number of photos liked by each user → Included a condition to fetch the user id who gave likes to all photos