1. 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.
 Your Task: Identify the five oldest users on Instagram from the provided database.

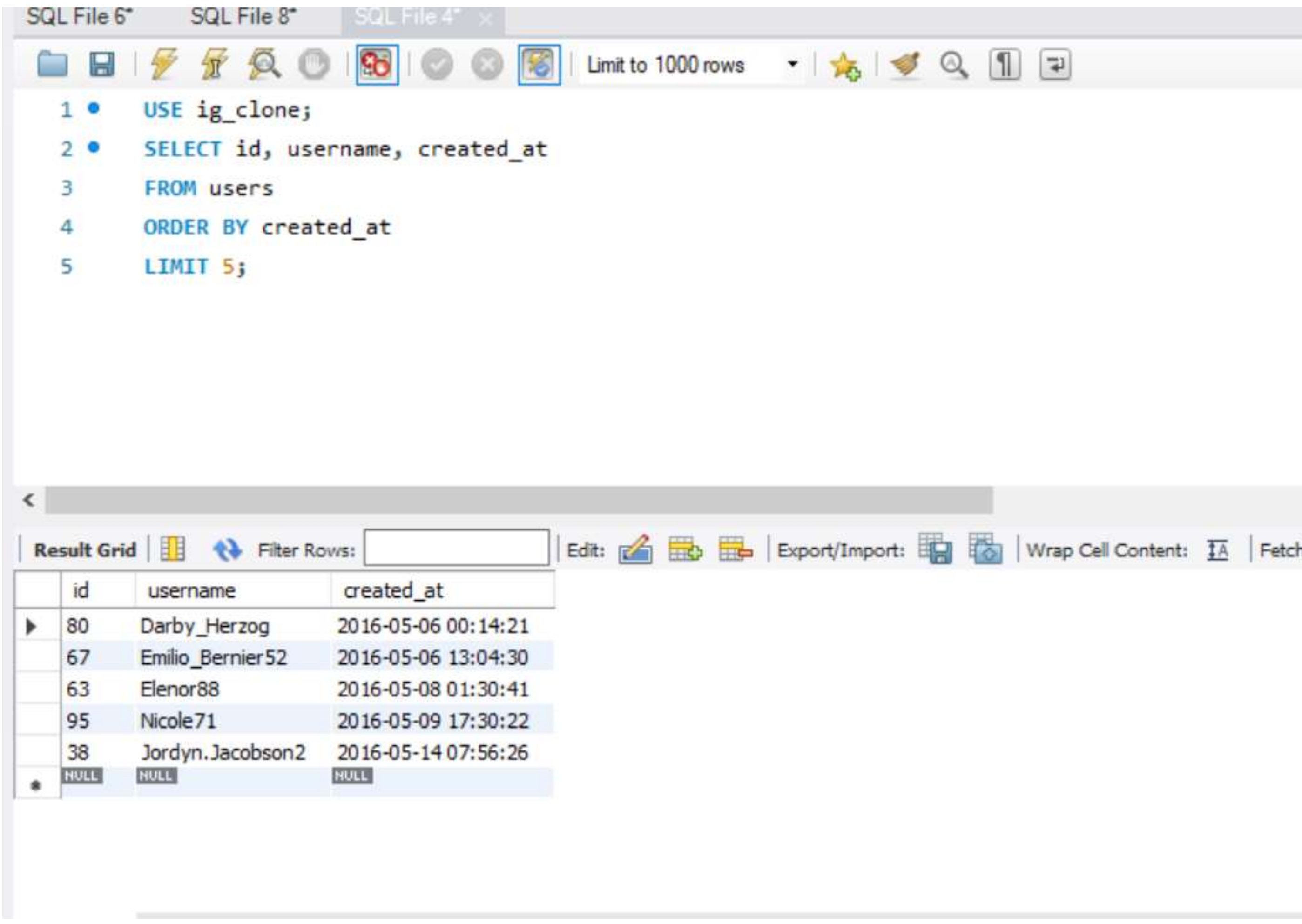
   USE ig_clone;

SELECT id, username, created_at

FROM users

ORDER BY created_at

LIMIT 5;



```
USE ig_clone;
SELECT id, username, created_at
FROM users
ORDER BY created_at
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
```

2.Inactive User Engagement: The team wants to encourage inactive users to start posting by sending them promotional emails.

SELECT u.username

FROM ig_clone.users u

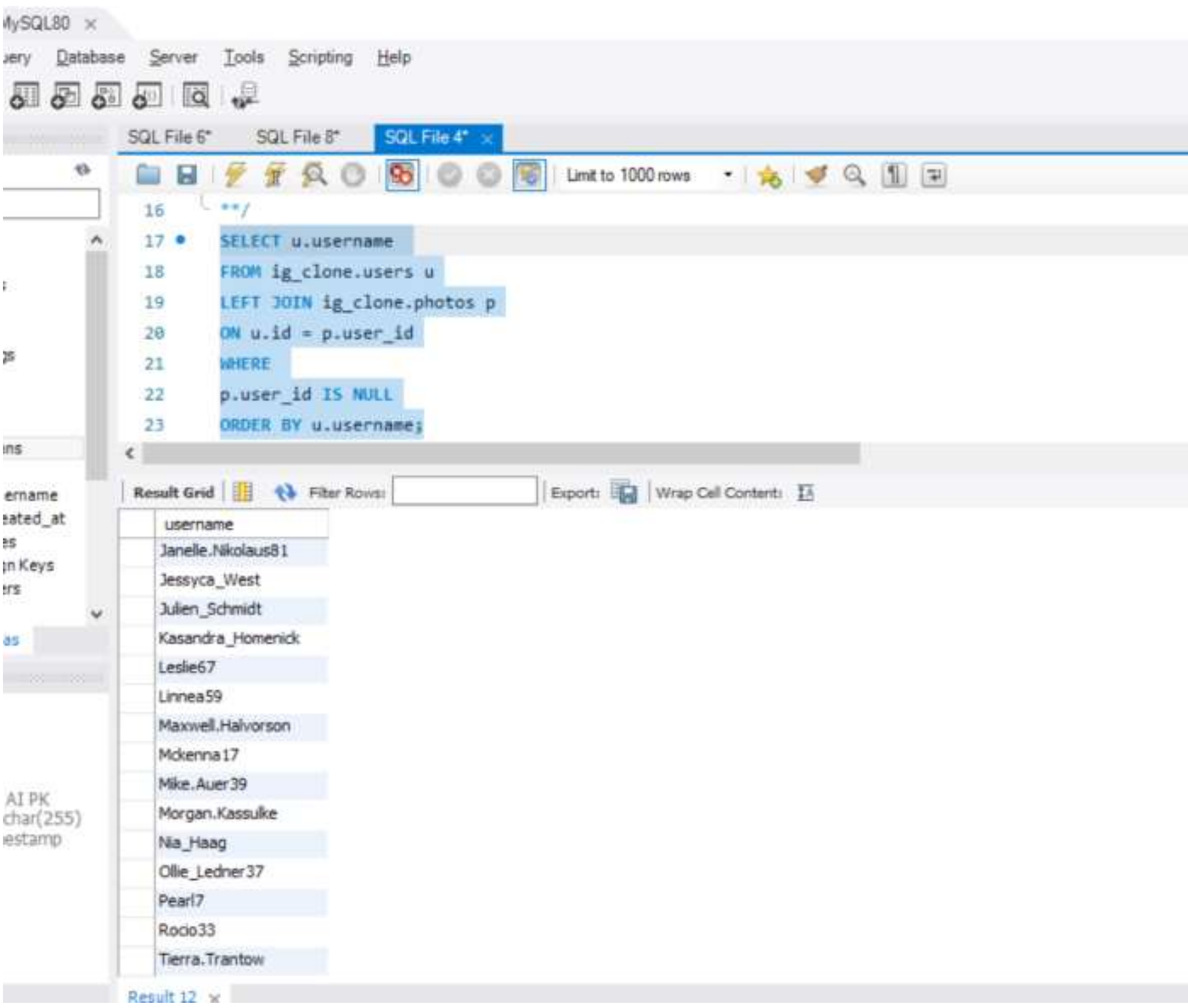
LEFT JOIN ig_clone.photos p

ON u.id = p.user_id

WHERE

p.user_id IS NULL

ORDER BY u.username;



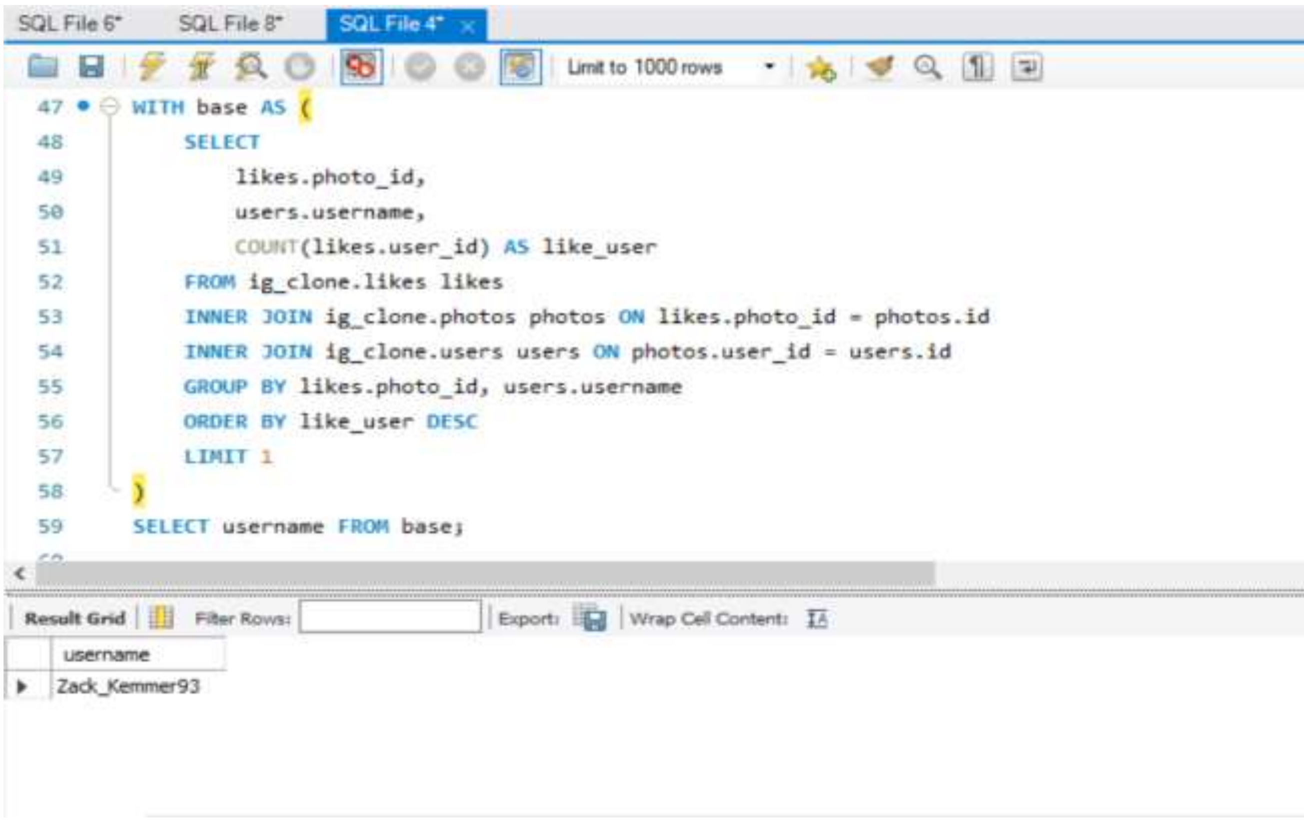
```
**/
SELECT u.username
FROM ig_clone.users u
LEFT JOIN ig_clone.photos p
ON u.id = p.user_id
WHERE
p.user_id IS NULL
ORDER BY u.username;
```

```
username
Janelle.Nikolaus81
Jessyca_West
Julien_Schmidt
Kasandra_Homenick
Leslie67
Linnea59
Maxwell.Halvorson
Mckenna17
Mike.Auer39
Morgan.Kassulke
Nia_Haag
Ollie_Ledner37
Pearl7
Rocio33
Tierra.Trantow
```

3. **Contest Winner Declaration:** The team has organized a contest where the user with the most likes on a single photo wins.
Your Task: Determine the winner of the contest and provide their details to the team.

WITH base AS (

```
    SELECT
        likes.photo_id,
        users.username,
        COUNT(likes.user_id) AS like_user
    FROM ig_clone.likes likes
    INNER JOIN ig_clone.photos photos ON likes.photo_id = photos.id
    INNER JOIN ig_clone.users users ON photos.user_id = users.id
    GROUP BY likes.photo_id, users.username
    ORDER BY like_user DESC
    LIMIT 1
)
SELECT username FROM base;
```

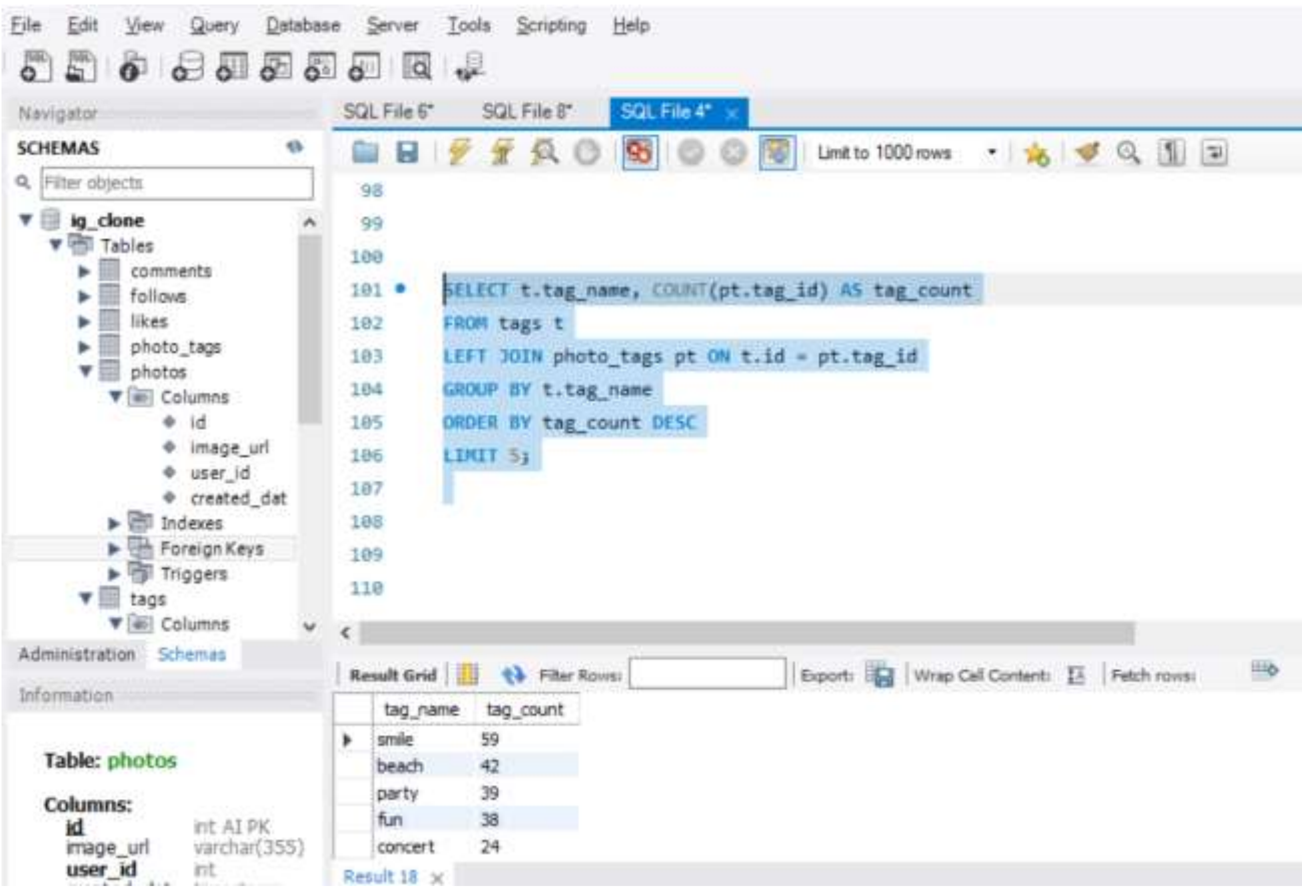


```
WITH base AS (
    SELECT
        likes.photo_id,
        users.username,
        COUNT(likes.user_id) AS like_user
    FROM ig_clone.likes likes
    INNER JOIN ig_clone.photos photos ON likes.photo_id = photos.id
    INNER JOIN ig_clone.users users ON photos.user_id = users.id
    GROUP BY likes.photo_id, users.username
    ORDER BY like_user DESC
    LIMIT 1
)
SELECT username FROM base;
```

```
username
Zack_Kemmer93
```

4. **Hashtag Research:** A partner brand wants to know the most popular hashtags to use in their posts to reach the most people.
Your Task: Identify and suggest the top five most commonly used hashtags on the platform.

```
 SELECT t.tag_name, COUNT(pt.tag_id) AS tag_count
FROM tags t
LEFT JOIN photo_tags pt ON t.id = pt.tag_id
GROUP BY t.tag_name
ORDER BY tag_count DESC
LIMIT 5;
```

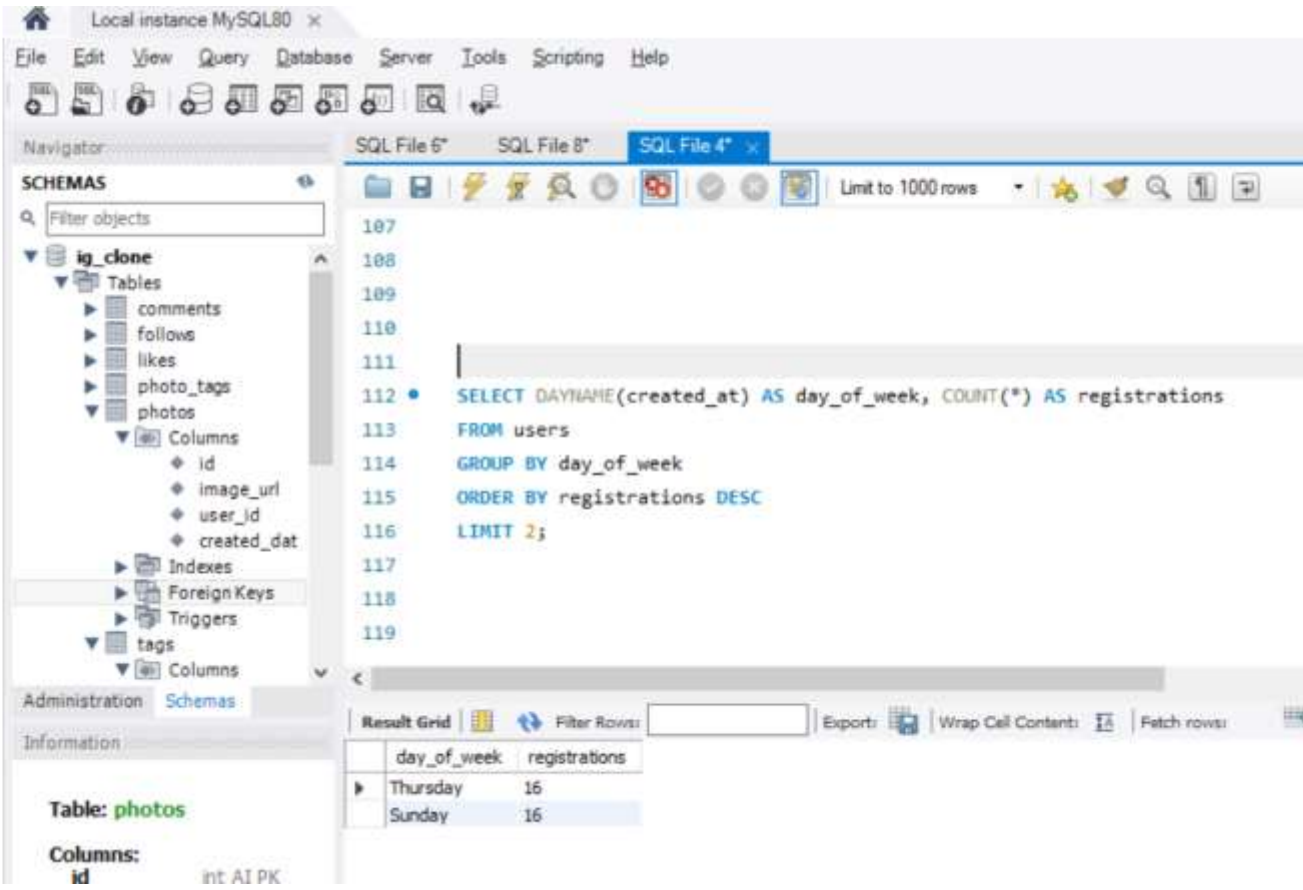


```
SELECT t.tag_name, COUNT(pt.tag_id) AS tag_count
FROM tags t
LEFT JOIN photo_tags pt ON t.id = pt.tag_id
GROUP BY t.tag_name
ORDER BY tag_count DESC
LIMIT 5;
```

tag_name	tag_count
smile	59
beach	42
party	39
fun	38
concert	24

5. **Ad Campaign Launch:** The team wants to know the best day of the week to launch ads.

Your Task: Determine the day of the week when most users register on Instagram. Provide insights on when to schedule an ad campaign.



```
SELECT DAYNAME(created_at) AS day_of_week, COUNT(*) AS registrations
FROM users
GROUP BY day_of_week
ORDER BY registrations DESC
LIMIT 2;
```

day_of_week	registrations
Thursday	16
Sunday	16

Table: photos

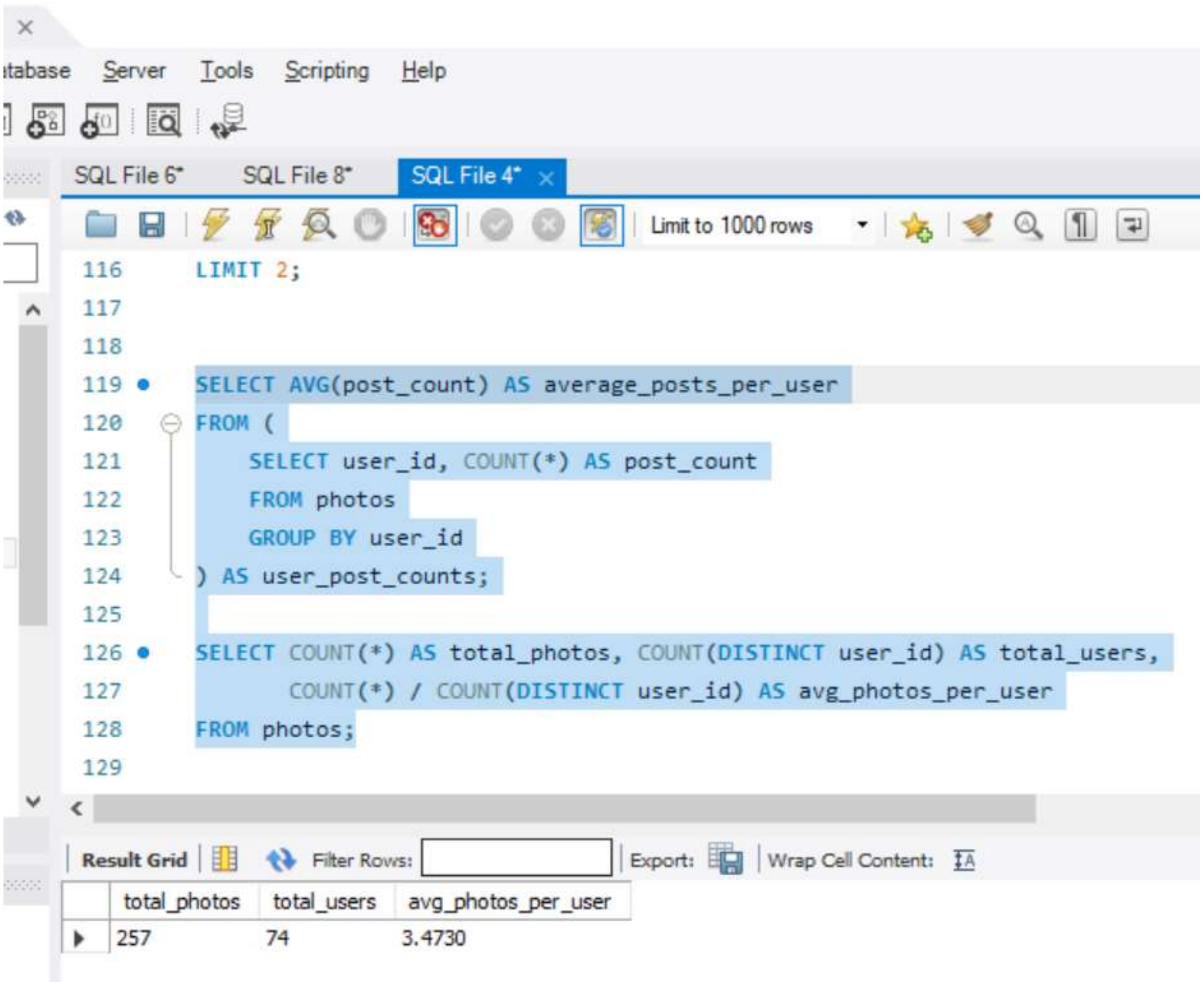
Columns:
id int AI PK

Investor Metrics:

1. **User Engagement: Investors want to know if users are still active and posting on Instagram or if they are making fewer posts.**
   Your Task: Calculate the average number of posts per user on Instagram. Also, provide the total number of photos on Instagram divided by the total number of users.

```
SELECT AVG(post_count) AS average_posts_per_user
FROM (
   SELECT user_id, COUNT(*) AS post_count
   FROM photos
   GROUP BY user_id
) AS user_post_counts;

SELECT COUNT(*) AS total_photos, COUNT(DISTINCT user_id) AS total_users,
     COUNT(*) / COUNT(DISTINCT user_id) AS avg_photos_per_user
FROM photos;
```



```
LIMIT 2;


SELECT AVG(post_count) AS average_posts_per_user
FROM (
    SELECT user_id, COUNT(*) AS post_count
    FROM photos
    GROUP BY user_id
) AS user_post_counts;

SELECT COUNT(*) AS total_photos, COUNT(DISTINCT user_id) AS total_users,
       COUNT(*) / COUNT(DISTINCT user_id) AS avg_photos_per_user
FROM photos;
```

```
total_photos  total_users  avg_photos_per_user
257           74           3.4730
```

2. **Bots & Fake Accounts: Investors want to know if the platform is crowded with fake and dummy accounts.**
   Your Task: Identify users (potential bots) who have liked every single photo on the site, as this is not typically possible for a normal user.

```
SELECT l.user_id, u.username
FROM users u
JOIN (
   SELECT user_id, COUNT(DISTINCT photo_id) AS liked_photos
   FROM likes
   GROUP BY user_id
) l ON u.id = l.user_id
WHERE l.liked_photos = (SELECT COUNT(*) FROM photos);
```

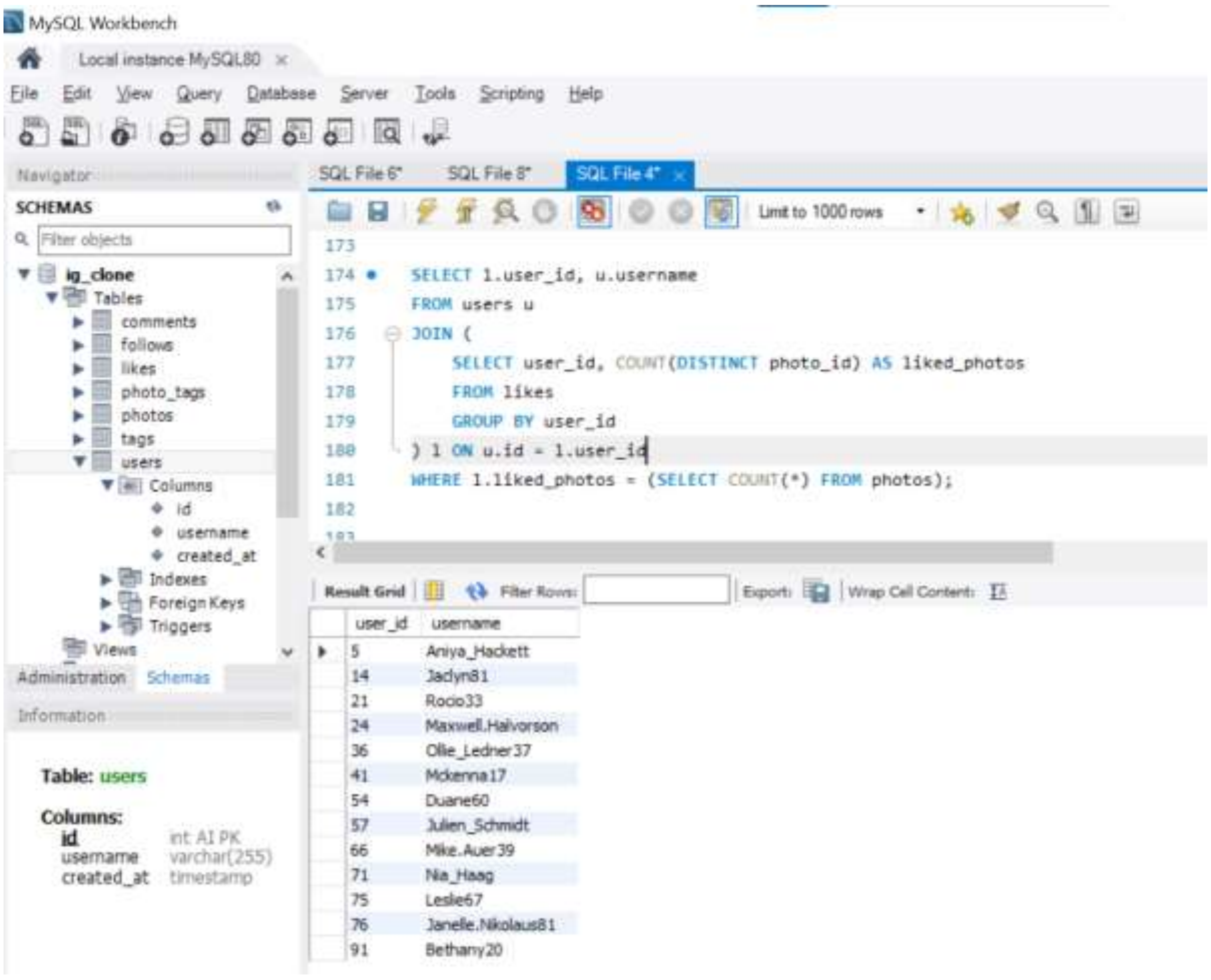


Table: users

Columns:

id	int AI PK
username	varchar(255)
created_at	timestamp

```sql
SELECT l.user_id, u.username
FROM users u
JOIN (
    SELECT user_id, COUNT(DISTINCT photo_id) AS liked_photos
    FROM likes
    GROUP BY user_id
) l ON u.id = l.user_id
WHERE l.liked_photos = (SELECT COUNT(*) FROM photos);
```

user_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