SQL QUERIES FOR THE ANALYSIS

A) Marketing Analysis:

1. Loyal User Reward (Oldest 5 users) :

```
SELECT *
FROM users
ORDER BY created_at ASC
LIMIT 5 ;
```

2.Remind Inactive Users to Start Posting :

```
SELECT u.id,
u.username,
Count(p.user_id) AS 'no._of_posts'
FROM users u
LEFT JOIN photos p
ON u.id = p.user_id
GROUP BY u.id
HAVING Count(p.user_id) = 0;
```

3.Declaring Contest Winner :

```
SELECT
  photos.id,
  username,
  photos.image_url,
  COUNT(*) AS total
FROM photos
INNER JOIN likes
  ON likes.photo_id = photos.id
INNER JOIN users
```

```
  ON photos.user_id = users.id
GROUP BY photos.id
ORDER BY total DESC
LIMIT 1;
```

4. Hashtag Research :

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

5.Launch AD Campaign :

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

B) Investor Metrics:

1.User Engagement :

```
SELECT (SELECT Count(id)
FROM photos) / (SELECT Count(DISTINCT user_id)
FROM photos) AS Average_posts_per_User,
(SELECT Count(id)
FROM photos) / (SELECT Count(id)
FROM users) AS Ratio_of_Total_Posts_to_Total_Users;
```

2.Bots & Fake Accounts :

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