

Project Description

This project aims to analyze user interactions and engagement patterns on Instagram to extract valuable insights. These insights will guide the product team in making data-driven decisions to

- Improve user experience
- Enhance engagement
- Support business growth

My Approach to work on the project

Data Extraction: Using MySQL to extract and analyze user data.

User Behavior Analysis: Identifying key engagement metrics, such as likes, comments, shares, and time spent on the platform.

Actionable Insights: Deriving trends and recommendations to assist the product, marketing, and development teams in refining strategies and enhancing user satisfaction.

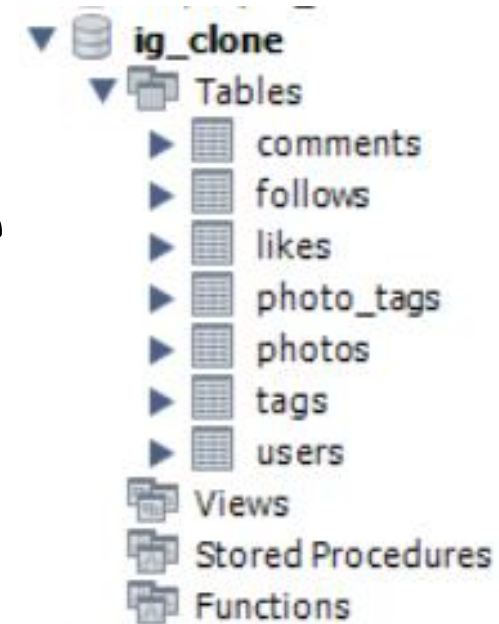
Decision Support: Presenting findings in a structured format to influence feature enhancements and business initiatives.

My Execution Approach towards the project

In further slide, I will showcase the execution steps and results

First I have created the database and tables as per the give data file executing the queries

Here is the screenshot of the same



Set A

Marketing Analysis

Q1

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.

Here is the query used to find the same and the output screenshot

```
93 • select * from users order by created_at asc limit 5;
```

Result Grid			
Filter Rows: <input type="text"/>			
	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

Q2

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

Your Task: Identify users who have never posted a single photo on Instagram

Here is the query used to find the same and the output screenshot

- ```
select u.id, u.username, p.id as Photo_id
from users u
left join photos p
on u.id = p.user_id
where p.id is null
order by u.id asc;
```

### Counting the inactive users

```
select count(u.id) as Never_Posted
from users u
left join photos p
on u.id = p.user_id
where p.id is null;
```

| Result Grid |              |
|-------------|--------------|
|             | Never_Posted |
| ▶           | 26           |

| Result Grid  |                    |          |
|--------------|--------------------|----------|
| Filter Rows: |                    |          |
| id           | username           | Photo_id |
| 5            | Aniya_Hackett      | NULL     |
| 7            | Kassandra_Homenick | NULL     |
| 14           | Jaclyn81           | NULL     |
| 21           | Rocio33            | NULL     |
| 24           | Maxwell.Halvorson  | NULL     |
| 25           | Tierra.Trantow     | NULL     |
| 34           | Pearl7             | NULL     |
| 36           | Ollie_Ledner37     | NULL     |
| 41           | Mckenna17          | NULL     |
| 45           | David.Osinski47    | NULL     |
| 49           | Morgan.Kassulke    | NULL     |
| 53           | Linnea59           | NULL     |
| 54           | Duane60            | NULL     |
| 57           | Julien_Schmidt     | NULL     |
| 66           | Mike.Auer39        | NULL     |
| 68           | Franco_Keebler64   | NULL     |
| 71           | Nia_Haag           | NULL     |
| 74           | Hulda.Macejkovic   | NULL     |

|    |                     |      |
|----|---------------------|------|
| 75 | Leslie67            | NULL |
| 76 | Janelle.Nikolaus81  | NULL |
| 80 | Darby_Herzog        | NULL |
| 81 | Esther.Zulauf61     | NULL |
| 83 | Bartholome.Bernhard | NULL |
| 89 | Jessyca_West        | NULL |
| 90 | Esmeralda.Mraz57    | NULL |
| 91 | Bethany20           | NULL |

### Q3

**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.

Here is the query used to find the same and the output screenshot

For this we will need to connect 3 tables

1. Photos to users to get the username of the photo id
2. Photos to likes to get which photo got the likes on the photos

The we need to sort the table in descending order after grouping them by photo id and user id



Here I have extracted top 3 posts with most likes and their usernames

```
select p.id as PhotoID, u.username as Username, count(l.user_id) as TotalLikes
from photos p
join users u on u.id = p.user_id
left join likes l on l.photo_id = p.id
group by p.id, u.username order by TotalLikes desc limit 3;
```

And the output is as below, obviously photo id no 145 and the corresponding name is the winner of the contest

| Result Grid  |         |                 |            |
|--------------|---------|-----------------|------------|
| Filter Rows: |         |                 |            |
|              | PhotoID | Username        | TotalLikes |
| ▶            | 145     | Zack_Kemmer93   | 48         |
|              | 182     | Adelle96        | 43         |
|              | 127     | Malinda_Streich | 43         |

### Q4

**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.

**Here we need to connect 2 tables together**

1. Tags
2. Photo\_tags

**using the tag\_id**

**Then we need to find which tag has got maximum on of usages in different photos**

```
select h.tag_name as TagName, count(t.photo_id) as UsageCount
from tags h
join photo_tags t on h.id = t.tag_id
group by h.tag_name
order by UsageCount desc limit 5;
```

Tags has been given alias 'h'  
Photo\_tags has been given alias 't'

| Result Grid |         |            | Filter Rows: |
|-------------|---------|------------|--------------|
|             | TagName | UsageCount |              |
| ▶           | smile   | 59         |              |
|             | beach   | 42         |              |
|             | party   | 39         |              |
|             | fun     | 38         |              |
|             | concert | 24         |              |

Q5

**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.

**Here is what we will need to extract to find the most suitable day for campaign launch**

**We will use the Users table as it has the date of creation as well**

**We need to find the most no. of IDs created on the particular day**

**Based on the user registration, we can suggest the days to run ad campaigns on**

Below is the query to get the day on which maximum no of users register on Instagram

```
select dayname(created_at) as CreationDay, count(username) as TotalUsers from users
group by CreationDay
order by TotalUsers desc;
```

| Result Grid |             |            | Filter Rows: |
|-------------|-------------|------------|--------------|
|             | CreationDay | TotalUsers |              |
| ▶           | Thursday    | 16         |              |
|             | Sunday      | 16         |              |
|             | Friday      | 15         |              |
|             | Tuesday     | 14         |              |
|             | Monday      | 14         |              |
|             | Wednesday   | 13         |              |
|             | Saturday    | 12         |              |

Total User count is as below

```
select count(id) as TotalUsers from users;
```

| Result Grid |            | Filter Rows: |
|-------------|------------|--------------|
|             | TotalUsers |              |
| ▶           | 100        |              |

It indicates that on Thursdays and Sundays, maximum no of users register themselves. And these can be the best days to float any ad campaign.

**Set B**

**Investor Metrics**

### Q1

**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.

**For this I need to calculate the total number of users and total number of posts and then find the average of that**

## Project 2 - Instagram User Analytics

Project by – Alokk Joshi


```
select
count(p.id) as TotalPosts,
count(distinct u.id) as TotalUsers,
(count(p.id) / count(distinct u.id)) as Avg_Post_Per_User
from users u
left join photos p on p.user_id = u.id;
```



| Result Grid  |            |            |                   |
|--------------|------------|------------|-------------------|
| Filter Rows: |            |            |                   |
|              | TotalPosts | TotalUsers | Avg_Post_Per_User |
| ▶            | 257        | 100        | 2.5700            |

Left join provides also the users who have not posted anything and this brings our avg down, whereas 'inner join' provide only the matching values and that is more accurate

```
select
count(p.id) as TotalPosts,
count(distinct u.id) as TotalUsers,
(count(p.id) / count(distinct u.id)) as Avg_Post_Per_User
from users u
join photos p on p.user_id = u.id;
```



| Result Grid  |            |            |                   |
|--------------|------------|------------|-------------------|
| Filter Rows: |            |            |                   |
|              | TotalPosts | TotalUsers | Avg_Post_Per_User |
| ▶            | 257        | 74         | 3.4730            |



Inactive users have been confirmed by the below query

```
select
count(u.id) as InactiveUsers
from users u
left join photos p on u.id = p.user_id where p.id is null;
```

|   | InactiveUsers |
|---|---------------|
| ▶ | 26            |

With left join

Result Grid | Filter Rows:

|   | TotalPosts | TotalUsers | Avg_Post_Per_User |
|---|------------|------------|-------------------|
| ▶ | 257        | 100        | 2.5700            |

With inner join

Result Grid | Filter Rows:

|   | TotalPosts | TotalUsers | Avg_Post_Per_User |
|---|------------|------------|-------------------|
| ▶ | 257        | 74         | 3.4730            |

Inactive users

Result Grid | Filter Rows:

|   | InactiveUsers |
|---|---------------|
| ▶ | 26            |

### Q2

**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.

Here we need to find the user id from likes table and username from users table linking them with inner join

Using “Having” will make sure that the later condition is met  
The later condition is – how many photos(posts) are there and they all have been liked

(the likes count of such users should match the no of photos)

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

| Result Grid | Filter Rows:      |
|-------------|-------------------|
| user_id     | username          |
| 5           | Aniya_Hackett     |
| 14          | Jaclyn81          |
| 21          | Rocio33           |
| 24          | Maxwell.Halvorson |
| 36          | Ollie_Ledner37    |
| 41          | Mckenna17         |

| Result Grid | Filter Rows:       |
|-------------|--------------------|
| user_id     | username           |
| 54          | Duane60            |
| 57          | Julien_Schmidt     |
| 66          | Mike.Auer39        |
| 71          | Nia_Haag           |
| 75          | Leslie67           |
| 76          | Janelle.Nikolaus81 |
| 91          | Bethany20          |

The total count of such fake users = 13

Which is further confirmed with  
subquery in the next slide

The Subquery for confirming the count of fake users

```
select count(*) as FakeUsers from
(
 select l.user_id, u.username
 from likes l
 left join users u on l.user_id = u.id
 group by l.user_id, u.username
 having count(l.user_id) = (select count(*) from photos))
As FakeUsers;
```

| Result Grid |           | Filter Rows: |
|-------------|-----------|--------------|
|             | FakeUsers |              |
| ▶           | 13        |              |

**Here are my 5 major learning from this project**

## **1. Data Can Tell Unexpected Stories**

When analyzing user engagement, I expected the oldest users to be the most active, but that wasn't always the case. Some early adopters had stopped posting altogether, while newer users were highly engaged.

## **2. Identifying Fake Activity Requires a Different Perspective**

Finding users who liked every single post seemed straightforward at first, but it made me realize how bots operate differently from normal users. Unlike organic engagement, which is random and varied, bot behavior is systematic and repetitive.

### **3. Engagement Peaks Can Influence Business Strategy**

Discovering the best days for user registrations and activity helped me understand how Instagram could optimize ad campaigns. Instead of just looking at raw numbers, I learned to think strategically and data driven.

### **4. SQL Can Be More Powerful Than Expected**

Initially, I relied on simple queries, but as I progressed, I saw how subqueries, joins, and aggregate functions could extract deep, meaningful insights from complex data. A single query could reveal behavioral trends, detect anomalies, and drive decision-making, making SQL a crucial skill for any data-driven role. I never thought that it would be that strong and imperative for data analytics.

### 5. Small Details Can Lead to Big Business Decisions

The hashtag research was interesting. knowing which hashtags are most popular can dramatically impact reach, engagement, and brand partnerships. It reinforced that in data analytics, even seemingly minor insights can have major business implications when used correctly.