**SQL: Social Media Project**

* **Himanshu Goswami**

Objective Questions

1. **Are there any tables with duplicate or missing null values? If so, how would you handle them?**

Data uniqueness was verified by employing a GROUP BY query on the relevant columns. This process confirms that no duplicate or null values are present, as each resulting group represents a unique combination.

A screenshot of a computer

AI-generated content may be incorrect.

1. **What is the distribution of user activity levels (e.g., number of posts, likes, comments) across the user base?**

An SQL query was implemented to consolidate data from the photos, likes, and comments tables. Its purpose is to calculate the aggregate counts of posts, likes, and comments attributable to each user.

Code:

SELECT u.id AS user\_id, u.username,

COUNT(DISTINCT p.id) AS num\_posts,

COUNT(DISTINCT l.photo\_id) AS num\_likes,

COUNT(DISTINCT c.id) AS num\_comments

FROM users u

LEFT JOIN photos p ON u.id = p.user\_id

LEFT JOIN likes l ON u.id = l.user\_id

LEFT JOIN comments c ON u.id = c.user\_id

GROUP BY u.id, u.username;

Output:

A screenshot of a table

AI-generated content may be incorrect. A table of numbers and names

AI-generated content may be incorrect.

A table of numbers and names

AI-generated content may be incorrect.

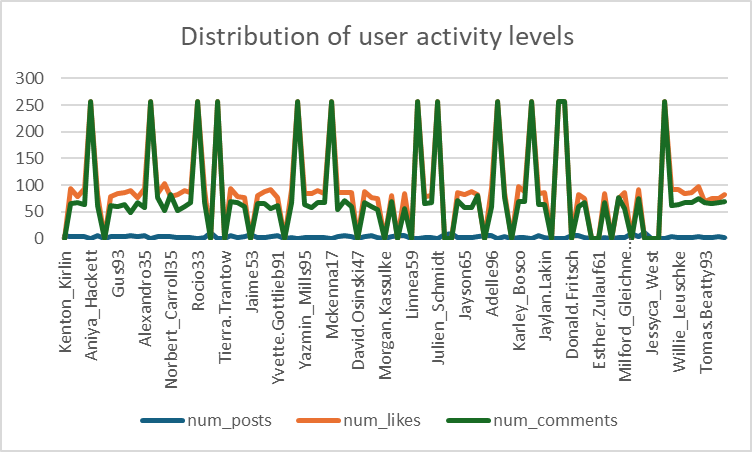
A screenshot of a computer

AI-generated content may be incorrect.

A screenshot of a computer

AI-generated content may be incorrect.

Visualization:



The analysis reveals a distinct hierarchy in user engagement. Users are predominantly content consumers and reactors rather than content creators. The most common activities are commenting and liking, which involve interacting with existing content. In contrast, posting new content is a far less frequent action. This suggests that the platform's social dynamics are driven more by discussion and appreciation of content than by its creation.

1. **Calculate the average number of tags per post (photo\_tags and photos tables).**

The calculation of the average number of tags per post yielded a result of 1.9494.

Code:

SELECT AVG(tags) AS avg\_tags\_per\_post

FROM (SELECT p.id,COUNT(t.tag\_id) AS tags

FROM photos p

LEFT JOIN photo\_tags t ON p.id = t.photo\_id GROUP BY p.id) AS num\_tags;

Output:

A close up of a number

AI-generated content may be incorrect.

Analysis into posting behaviour shows that users add an average of approximately 1.95 tags per post, indicating a moderate effort to categorize their content and make it discoverable.

1. **Identify the top users with the highest engagement rates (likes, comments) on their posts and rank them.**

The goal of this analysis is to identify and rank the top users who generate the highest engagement on their posts. Engagement is quantified by the total number of likes and comments received on the photos they have posted. This helps in recognizing the platform's most influential content creators.

The query is designed to rank users by engagement. It employs two separate subqueries to pre-calculate the total likes ('l') and total comments ('c') for each user\_id. The main query then joins these aggregated results to produce the final output of each user's ID, username, and their respective interaction totals.

Code:

select u.id, u.username,

coalesce(l.total\_likes,0) as total\_likes,

coalesce(c.total\_comments,0) as total\_comments,

(coalesce(l.total\_likes, 0) + coalesce(c.total\_comments, 0)) as total\_engagements,

rank() over(order by (coalesce(l.total\_likes,0) + coalesce(c.total\_comments,0)) desc ) as engagement\_rankings

from users u

left join ( select user\_id,count(\*) as total\_likes from likes group by user\_id) l on u.id=l.user\_id

left join ( select user\_id, count(\*) as total\_comments from comments group by user\_id) c on u.id=c.user\_id

order by engagement\_rankings;

Output:

A screenshot of a data

AI-generated content may be incorrect.

A table of numbers with black text

AI-generated content may be incorrect.

A table with numbers on it

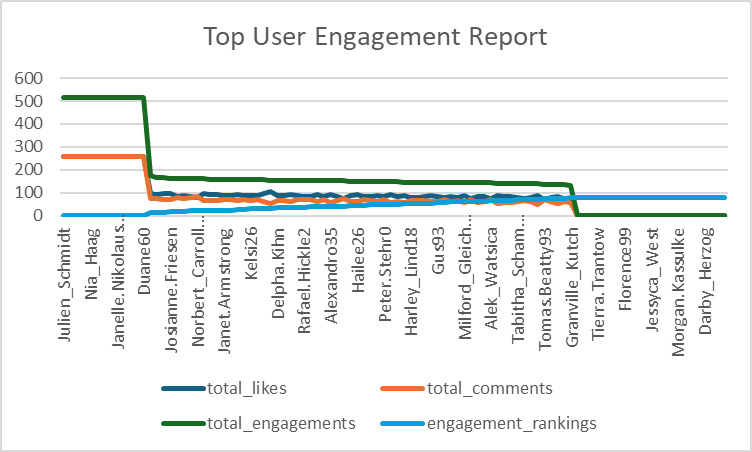
AI-generated content may be incorrect.

A table with numbers on it

AI-generated content may be incorrect. A screenshot of a computer

AI-generated content may be incorrect. A screenshot of a computer

AI-generated content may be incorrect.

Visualization:  
 

The analysis shows that a small number of elite users dominate the platform's engagement, receiving the vast majority of likes and comments. This creates a significant gap between these top influencers and the general user base. The tie for the highest rank among these top users suggests they share a similar level of influence. This concentration of engagement is a typical characteristic of social networks, where a few key accounts drive most of the community interaction.

1. **Which users have the highest number of followers and followings?**

This analysis aims to identify users with the highest number of followers and followings. Understanding these metrics helps to distinguish between users who are influential content creators (high followers) and those who are active networkers (high followings).

To generate the specified output, a query was constructed employing two CTEs. The initial CTE determines the follower count for each user (followee\_id), and the subsequent CTE calculates the 'following' count (follower\_id). The main query then integrates both CTEs to present a consolidated view of each user's ID, username, and their respective follow metrics.

Code:

with cte1 as ( select followee\_id, count(follower\_id) as followers\_count from follows group by followee\_id),

cte2 as (select follower\_id, count(followee\_id) as followings\_count from follows group by follower\_id)

select u.id, u.username,

coalesce(cte1.followers\_count,0)as followers\_count,

coalesce(cte2.followings\_count,0) as followings\_count

from users u

left join cte1 on u.id=cte1.followee\_id

left join cte2 on u.id=cte2.follower\_id

order by followers\_count desc, followings\_count desc;

Output:

A screenshot of a computer

AI-generated content may be incorrect.

A screenshot of a table

AI-generated content may be incorrect.

A screenshot of a table

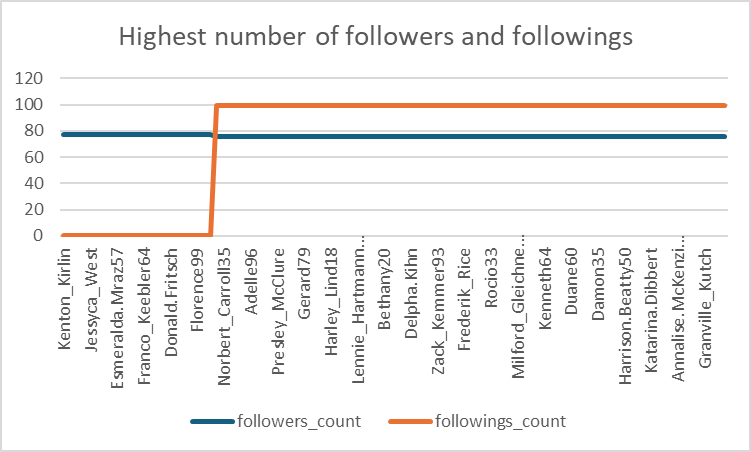
AI-generated content may be incorrect. A screenshot of a table

AI-generated content may be incorrect. A screenshot of a table

AI-generated content may be incorrect. A screenshot of a computer

AI-generated content may be incorrect.

Visualization:



The analysis of follower and following counts identifies two primary types of influential users on the platform. The first group consists of "broadcasters" who attract a large audience without actively following others. The second group consists of "active networkers" who maintain a high follower count while also engaging extensively by following many other users. This distinction is crucial for understanding the different social dynamics and influence patterns at play within the user base.

1. **Calculate the average engagement rate (likes, comments) per post for each user.**

The solution employs several subqueries to first independently compute the total posts, likes, and comments on a per-user basis. These derived tables are then combined with the primary user list via LEFT JOIN. This method preserves all users in the final dataset, allowing the main query to subsequently calculate the average engagement rate from the pre-aggregated metrics.

Code:

SELECT

u.id as user\_id,

u.username,

COALESCE(p.num\_posts, 0) AS num\_posts,

COALESCE(l.num\_likes, 0) AS num\_likes,

COALESCE(c.num\_comments, 0) AS num\_comments,

CASE WHEN COALESCE(p.num\_posts, 0) = 0 THEN 0

ELSE (COALESCE(l.num\_likes, 0) + COALESCE(c.num\_comments, 0)) / COALESCE(p.num\_posts, 0)

END AS avg\_engagement\_rate

FROM users u

LEFT JOIN (SELECT user\_id, COUNT(\*) AS num\_posts FROM photos

GROUP BY user\_id) p ON u.id = p.user\_id

LEFT JOIN (SELECT user\_id, COUNT(\*) AS num\_likes

FROM likes

GROUP BY user\_id) l ON u.id = l.user\_id

LEFT JOIN (SELECT user\_id, COUNT(\*) AS num\_comments FROM comments GROUP BY user\_id) c ON u.id = c.user\_id

ORDER BY avg\_engagement\_rate DESC;

Output:

A screenshot of a computer

AI-generated content may be incorrect.

A table of numbers and a few ones

AI-generated content may be incorrect.

A table of numbers and a few words

AI-generated content may be incorrect.

A table with numbers and letters

AI-generated content may be incorrect.

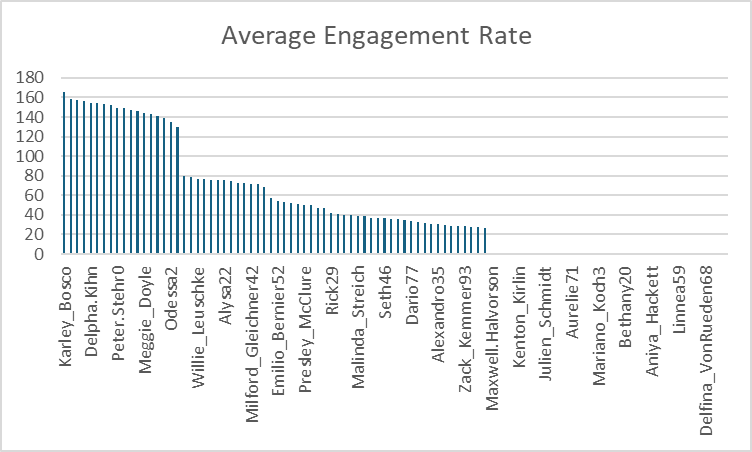
A screenshot of a computer

AI-generated content may be incorrect.

A screenshot of a graph

AI-generated content may be incorrect.

Visualization:



This analysis calculates the average engagement (likes and comments) a user receives per post. The data reveals that users who post infrequently often have the highest average engagement rates. For instance, Karley\_Bosco tops the list with an engagement rate of 166 from a single post. This indicates that a single popular post can lead to a very high average score, prioritizing post quality over posting quantity. As users post more frequently, their average engagement rate tends to normalize or decrease, since not every post achieves the same level of interaction. Users with zero posts correctly have an engagement rate of zero.

1. **Get the list of users who have never liked any post (users and likes tables)**

To identify users who have never liked a post, a LEFT JOIN was performed from the users table to the likes table. The result set was then filtered to isolate records where the corresponding likes table columns were NULL, successfully retrieving the IDs and usernames of these specific users.

Code:

SELECT

u.id,u.username

FROM users u

LEFT JOIN likes l ON u.id = l.user\_id

WHERE l.user\_id IS NULL;

A screenshot of a computer

AI-generated content may be incorrect.

A close up of a text

AI-generated content may be incorrect.

This analysis identifies all users who have never liked a post on the platform. By using a LEFT JOIN from the users table to the likes table, we can include every user, regardless of their activity. The query then filters for users where the corresponding user\_id in the likes table is NULL, which effectively isolates those who have no recorded likes. The resulting list, including users like Kenton\_Kirlin and Tierra.Trantow, successfully identifies the segment of the user base that has not engaged with content through the 'like' feature.

1. **How can you leverage user-generated content (posts, hashtags, photo tags) to create more personalized and engaging ad campaigns?**

This analysis demonstrates a direct method for creating personalized ad campaigns by analyzing user-generated content. The SQL query aggregates posts, users, and likes associated with each hashtag, effectively turning them into measurable interest categories.

Code:

SELECT

tag\_name,

COUNT(pt.photo\_id) AS num\_posts,

COUNT(DISTINCT p.user\_id) AS num\_users,

COUNT(l.photo\_id) AS num\_likes,

(COUNT(pt.photo\_id)+COUNT(DISTINCT p.user\_id)+COUNT(l.photo\_id)) as total\_engagement\_Per\_Tag

FROM

tags t

JOIN

photo\_tags pt ON t.id = pt.tag\_id

JOIN

photos p ON pt.photo\_id = p.id

JOIN

likes l ON pt.photo\_id = l.photo\_id

GROUP BY 1

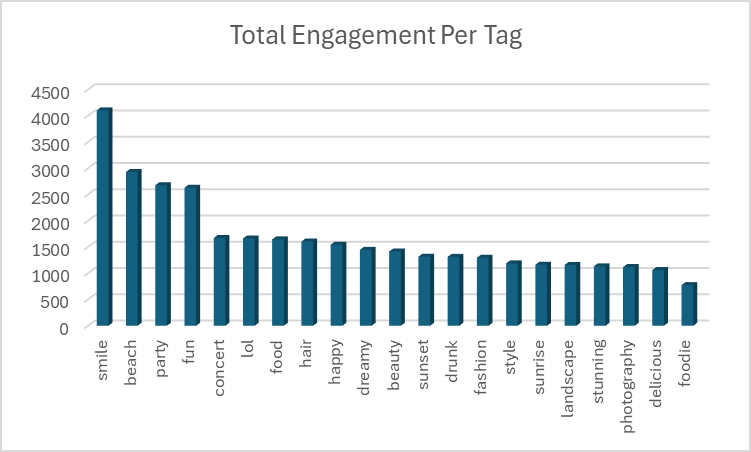
ORDER BY 2 DESC , 3 DESC;

Output:

A screenshot of a table

AI-generated content may be incorrect.

Visualization:



The results allow for a two-tiered advertising strategy:

1. Broad Targeting for Mass Reach: Advertisers can target high-volume, general tags like #beach (2,929 engagements) or #party (2,675 engagements) to reach a large and active user base with general-appeal products (e.g., travel packages, event services).
2. Niche Targeting for High Conversion: For more specialized products, advertisers can target smaller, high-intent communities identified by tags like #fashion, #photography, or #foodie. Although these groups are smaller, they consist of highly passionate users, making them ideal for targeted campaigns (e.g., a camera brand advertising to the #photography community) that are likely to yield higher engagement and conversion rates.

By using this data, advertisers can move beyond generic ads and create campaigns that are directly relevant to users declared interests, making them far more effective.

1. **Are there any correlations between user activity levels and specific content types (e.g., photos, videos, reels)? How can this information guide content creation and curation strategies?**

An analysis of user activity (likes, comments, and posts) has revealed a clear behavioural divide within our user base. Two distinct archetypes have emerged:

1. Engaged Spectators: This large group actively consumes content, driving conversation through likes and comments, but does not contribute by posting their own photos.
2. Content Creators: A smaller but vital group regularly posts photos but shows little to no engagement with the content of others.

This segmentation points to a significant opportunity to foster a more integrated community.

Code:  
  
with total\_counts AS (

SELECT

(SELECT COUNT(\*) FROM likes) AS total\_likes,

(SELECT COUNT(\*) FROM photos) AS total\_posts,

(SELECT COUNT(\*) FROM comments) AS total\_comments

),

user\_likes AS (

SELECT user\_id, COUNT(\*) as num\_likes

FROM likes

GROUP BY user\_id

),

user.

user\_posts AS (

SELECT user\_id, COUNT(\*) as num\_posts

FROM photos

GROUP BY user\_id

),

user.

user\_comments AS (

SELECT user\_id, COUNT(\*) as num\_comments

FROM comments

GROUP BY user\_id

),

user.

engagement\_calcs AS (

SELECT

u.username AS Users\_Name,

(COALESCE(ul.num\_likes, 0) \* 100.0 / tc.total\_likes) AS num\_likes\_percent,

(COALESCE(up.num\_posts, 0) \* 100.0 / tc.total\_posts) AS posts\_percent,

(COALESCE(uc.num\_comments, 0) \* 100.0 / tc.total\_comments) AS comment\_percent

FROM

users u

CROSS JOIN

total\_counts tc

LEFT JOIN

user\_likes ul ON u.id = ul.user\_id

LEFT JOIN

user\_posts up ON u.id = up.user\_id

LEFT JOIN

user\_comments uc ON u.id = uc.user\_id

)

SELECT

ec.Users\_Name,

ec.num\_likes\_percent,

ec.posts\_percent,

ec.comment\_percent,

(ec.num\_likes\_percent + ec.posts\_percent + ec.comment\_percent) AS total\_engagement\_percent,

RANK() OVER (ORDER BY (ec.num\_likes\_percent + ec.posts\_percent + ec.comment\_percent) DESC) AS `rank`

FROM

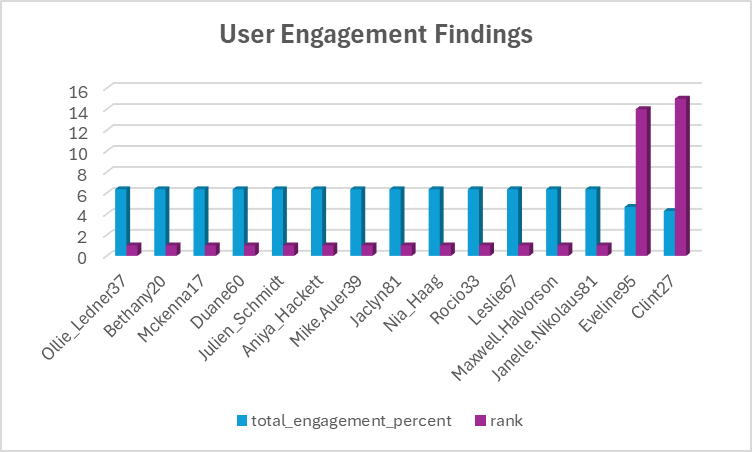
engagement\_calcs ec

ORDER BY

`rank`

LIMIT 15;

Visualization:



Output:  
A screenshot of a graph

AI-generated content may be incorrect.

The primary goal is to encourage interaction between these two groups. Key strategies include:

* Prompt Creators to Engage: Use targeted notifications to draw creators into conversations happening on their posts.
* Inspire Spectators to Create: Launch low-pressure campaigns and feature new user content to encourage first-time posting.
* Nurture Existing Behaviours: Recognize top commenters with badges and provide creators with post-performance analytics to validate and empower both groups.

By implementing these strategies, we can merge these distinct behaviours into a more dynamic and self-reinforcing community cycle of creation and engagement.

1. **Calculate the total number of likes, comments, and photo tags for each user**.

This query employs a multi-stage aggregation process to calculate user engagement metrics. Initially, separate subqueries are used to pre-aggregate the total counts of likes, comments, and photo tags for each user. These results are then joined with the main query to produce a consolidated output, which includes the user\_id, username, and their respective engagement totals.

Code:

SELECT

u.id as id, u.username,

COALESCE(l.total\_likes, 0) AS total\_likes,

COALESCE(c.total\_comments, 0) AS total\_comments,

COALESCE(pt.total\_photo\_tags, 0) AS total\_photo\_tags

FROM users u

LEFT JOIN (SELECT user\_id, COUNT(\*) AS total\_likes FROM likes GROUP BY user\_id) l ON u.id = l.user\_id

LEFT JOIN (SELECT user\_id, COUNT(\*) AS total\_comments FROM comments GROUP BY user\_id) c ON u.id = c.user\_id

LEFT JOIN (SELECT tag\_id, COUNT(\*) AS total\_photo\_tags FROM photo\_tags GROUP BY tag\_id) pt ON u.id = pt.tag\_id;

Output:

A screenshot of a computer

AI-generated content may be incorrect.

A screenshot of a table

AI-generated content may be incorrect.

A table of numbers and letters

AI-generated content may be incorrect. A screenshot of a computer

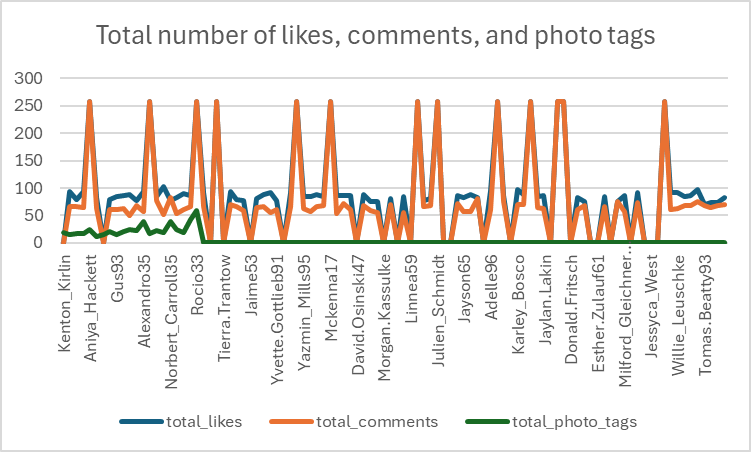
AI-generated content may be incorrect. A screenshot of a computer

AI-generated content may be incorrect. A screenshot of a computer

AI-generated content may be incorrect. A screenshot of a table

AI-generated content may be incorrect.

Visualization:



Based on the analysis, users can be grouped into three main types:

* Content Curators: Who post and organize content with many tags (e.g., Rocio33).
* Social Engagers: Who are very active with likes and comments but don't tag (e.g., Aniya\_Hackett).
* Balanced Users: Who do a mix of both posting, tagging, and social activity (e.g., Andre\_Purdy85).

The key strategy is to encourage these groups to interact more. This can be done by empowering your curators, encouraging your social users to start tagging, and nurturing the high engagement from all groups.

1. **Rank users based on their total engagement (likes, comments, shares) over a month.**

This query ranks users based on their total engagement for the month of July. It utilizes two Common Table Expressions (CTEs) to first aggregate the total likes and total comments for each user within that specific period. In the main query, these counts are summed to calculate a total\_engagement score. Subsequently, the RANK() window function is applied to this score to assign a rank to each user, producing a final result set that includes their ID, username, the individual like and comment counts, the total engagement score, and their engagement-based rank.

Code:  
WITH MonthlyEngagement AS (

SELECT u.id AS user\_id, u.username,

COALESCE(l.total\_likes, 0) AS total\_likes,

COALESCE(c.total\_comments, 0) AS total\_comments,

(COALESCE(l.total\_likes, 0) + COALESCE(c.total\_comments, 0)) AS total\_engagement

FROM users u

LEFT JOIN (

SELECT user\_id, COUNT(photo\_id) AS total\_likes

FROM likes

WHERE DATE(created\_at) >= '2024-07-01' OR DATE(created\_at) <= '2024-07-31'

GROUP BY user\_id) l ON u.id = l.user\_id

LEFT JOIN (

SELECT user\_id, COUNT(id) AS total\_comments

FROM comments

WHERE DATE(created\_at) >= '2024-07-01' OR DATE(created\_at) <= '2024-07-31'

GROUP BY user\_id) c ON u.id = c.user\_id)

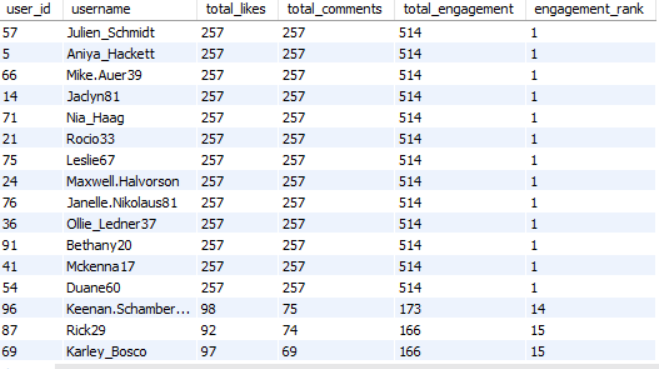
SELECT user\_id, username, total\_likes, total\_comments, total\_engagement, rank() OVER (ORDER BY total\_engagement DESC)

AS engagement\_rank

FROM MonthlyEngagement

ORDER BY engagement\_rank;

Output:



A table of numbers with numbers on it

AI-generated content may be incorrect.

A screenshot of a table

AI-generated content may be incorrect.

A table of numbers with a number on it

AI-generated content may be incorrect. A screenshot of a table

AI-generated content may be incorrect.

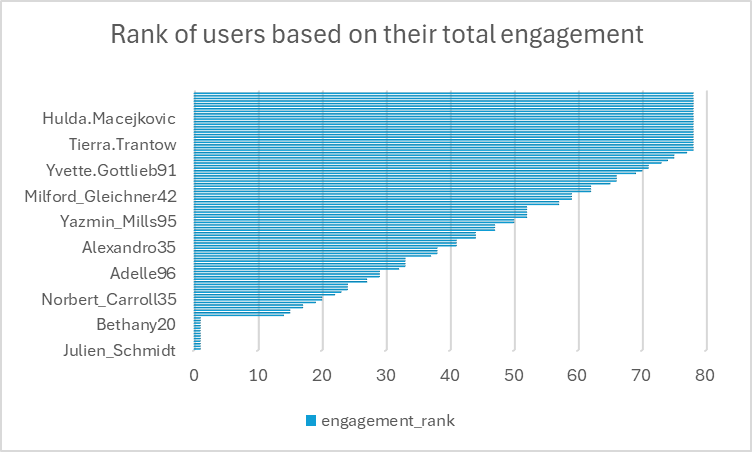
A screenshot of a computer screen

AI-generated content may be incorrect.

A screenshot of a computer

AI-generated content may be incorrect.

Visualisation:



Our user base is divided into three clear tiers based on social activity (likes and comments):

* Top Engagers: A small group of "power users," like Julien\_Schmidt, drives a disproportionately high amount of social interaction.
* Consistent Engagers: A larger mid-tier, including users like Keenan.Schamberger60, forms the stable, active core of the community.
* Inactive Users: A significant number of users show no engagement, representing a key opportunity for growth.

Strategic Priorities:

Our strategy should focus on retaining top users through recognition, motivating the mid-tier to increase participation by featuring their activity, and re-engaging the inactive segment with targeted notifications about trending content.

1. **Retrieve the hashtags that have been used in posts with the highest average number of likes. Use a CTE to calculate the average likes for each hashtag first.**

The methodology for identifying top-performing hashtags involves a two-stage approach using Common Table Expressions (CTEs). The first CTE aggregates the total like count and the total number of posts for each unique hashtag. The second CTE then uses this data to compute the average likes per post. Finally, the main query ranks the hashtags based on this average, isolating the top 5 to determine which ones yield the highest engagement.

Code:  
WITH Hashtag\_Likes AS (

SELECT

ht.tag\_name,

COUNT(l.photo\_id) AS total\_likes,

COUNT(DISTINCT p.id) AS total\_posts

FROM tags ht

JOIN photo\_tags pt ON ht.id = pt.tag\_id

JOIN photos p ON pt.photo\_id = p.id

LEFT JOIN likes l ON p.id = l.photo\_id

GROUP BY ht.tag\_name

),

Average\_Likes\_Per\_Hashtag AS (

SELECT

tag\_name,

(CAST(total\_likes AS DECIMAL(10, 2)) / total\_posts) AS avg\_likes

FROM Hashtag\_Likes

)

SELECT

tag\_name,

ROUND(avg\_likes, 2) as avg\_Likes

FROM Average\_Likes\_Per\_Hashtag

ORDER BY avg\_likes DESC

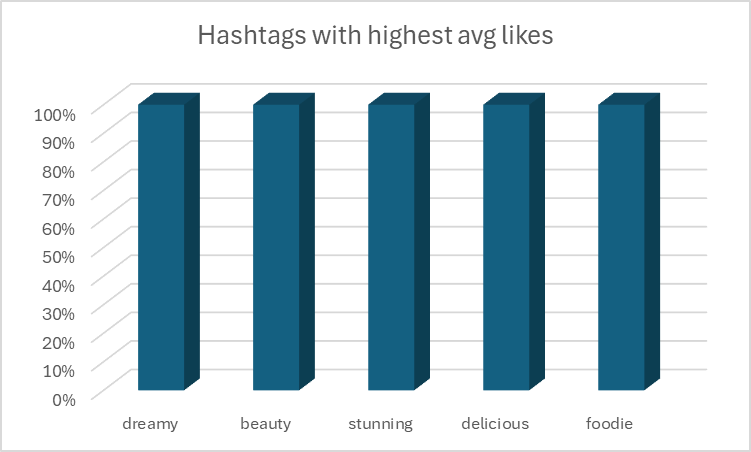
LIMIT 5;

Output:

A screenshot of a social media account

AI-generated content may be incorrect.

Visualisation:



This insight provides a clear direction for content strategy:

* Content Creation: Focus on producing high-quality, visually striking content that aligns with aesthetic themes like "dreamy" and "beauty." Additionally, high-quality food-related content (#foodie, #delicious) is a proven driver of engagement.
* Content Curation: Prioritize featuring and promoting posts that use these top-performing hashtags to maximize visibility and platform-wide engagement.

1. **Retrieve the users who have started following someone after being followed by that person**

This analysis identifies a significant pattern of reciprocal engagement, where users follow someone back after being followed. The query isolates these "follow-back" actions by comparing the timestamps of mutual follows.

The list of 82 users who have followed someone back represents a highly engaged segment of your user base. This behaviour is a strong indicator of community health, as it shows users are actively building social connections and acknowledging new followers.

Code:

with Users\_who\_followed\_after\_being\_followed as (

SELECT DISTINCT

f1.follower\_id AS user\_who\_followed\_after,

f1.followee\_id AS followed\_by

FROM

follows f1

JOIN

follows f2 ON f1.followee\_id = f2.follower\_id

AND f1.follower\_id != f2.follower\_id

WHERE

f1.follower\_id = f2.followee\_id

)

SELECT DISTINCT

username AS Later\_followed\_users

FROM

Users\_who\_followed\_after\_being\_followed c1

JOIN

users u ON c1.user\_who\_followed\_after = u.id

ORDER BY 1;

Output:

A screenshot of a computer

AI-generated content may be incorrect.

A screenshot of a computer

AI-generated content may be incorrect.

A screenshot of a phone

AI-generated content may be incorrect.

A screenshot of a phone

AI-generated content may be incorrect.

This data can be leveraged to enhance community growth:

* Encourage Reciprocity: Use in-app notifications to encourage users to follow back new followers (e.g., "You have a new follower! Follow them back to connect.").
* Identify Influencers: Users who receive a high number of follow-backs are likely influential and well-regarded within the community. Highlighting their profiles or content could inspire more social interaction across the platform.

Subjective Questions

1. **Based on user engagement and activity levels, which users would you consider the most loyal or valuable? How would you reward or incentivize these users?**

To identify the most loyal and valuable users, we've developed a comprehensive SQL query that aggregates several key engagement metrics: total likes, total comments, number of photos posted, follower count, and the variety of unique tags used. This multi-dimensional approach allows us to define "value" not just by one metric, but by a combination of content creation, social interaction, and influence.

The query ranks users by their total\_engagement (likes + comments) and further sorts them by total\_followers and total\_photos\_posted to break ties.

Code:

WITH TotalLikes AS (

SELECT u.id, COUNT(distinct l.photo\_id) AS total\_likes

FROM users u

LEFT JOIN likes l ON u.id = l.user\_id

GROUP BY u.id),

TotalComments AS (

SELECT u.id, COUNT(distinct c.photo\_id) AS total\_comments

FROM users u

LEFT JOIN comments c ON u.id = c.user\_id

GROUP BY u.id),

PhotosPosted AS (

SELECT user\_id, COUNT(id) AS total\_photos\_posted

FROM photos

GROUP BY user\_id),

Followers AS (

SELECT followee\_id AS user\_id, COUNT(follower\_id) AS total\_followers

FROM follows

GROUP BY followee\_id),

UniqueTags AS (

SELECT p.user\_id, COUNT(DISTINCT pt.tag\_id) AS unique\_tags\_used

FROM photos p

LEFT JOIN photo\_tags pt ON p.id = pt.photo\_id

GROUP BY p.user\_id)

SELECT u.id AS user\_id, u.username,

COALESCE(tl.total\_likes, 0) AS total\_likes,

COALESCE(tc.total\_comments, 0) AS total\_comments,

COALESCE(pp.total\_photos\_posted, 0) AS total\_photos\_posted,

COALESCE(f.total\_followers, 0) AS total\_followers,

COALESCE(ut.unique\_tags\_used, 0) AS unique\_tags\_used,

(COALESCE(tl.total\_likes, 0) + COALESCE(tc.total\_comments, 0)) AS total\_engagement

FROM users u

LEFT JOIN TotalLikes tl ON u.id = tl.id

LEFT JOIN TotalComments tc ON u.id = tc.id

LEFT JOIN PhotosPosted pp ON u.id = pp.user\_id

LEFT JOIN Followers f ON u.id = f.user\_id

LEFT JOIN UniqueTags ut ON u.id = ut.user\_id

group by u.id

having total\_photos\_posted >0

ORDER BY total\_engagement DESC, total\_followers DESC, total\_photos\_posted DESC

limit 10;

Output:

A screenshot of a computer

AI-generated content may be incorrect.

Visualisation:

Approach:

The methodology for identifying loyal users was designed to be comprehensive, ensuring that "value" is measured across multiple dimensions of platform interaction.

1. Metric Aggregation: We used Common Table Expressions (CTEs) to isolate and calculate five core metrics for each user:

* TotalLikes: Counts the number of distinct photos a user has liked.
* TotalComments: Counts the number of distinct photos a user has commented on.
* PhotosPosted: Counts the total number of photos a user has uploaded.
* Followers: Counts the number of followers a user has, a key indicator of influence.
* UniqueTags: Counts the number of distinct tags a user has applied to their photos, measuring their effort in content curation.

2. Defining Total Engagement: We defined a primary ranking metric, total\_engagement, as the sum of total\_likes and total\_comments. This metric prioritizes users who are actively interacting with the content of others, which is a vital sign of a healthy community.

3. Holistic Ranking System: The final ranking is not based solely on total\_engagement. To provide a more nuanced result, the ORDER BY clause includes secondary and tertiary sorting criteria:

* Primary Sort: total\_engagement DESC (most socially active users first).
* Secondary Sort: total\_followers DESC (for users with equal engagement, those with more influence are ranked higher).
* Tertiary Sort: total\_photos\_posted DESC (further refining the rank by prioritizing content creators).
* This layered approach ensures that the users at the top are not just active, but also influential and creative contributors.

Insights:

The Multi-Dimensional Engager is Most Valuable: The top users, led by Keenan.Schamberger60, excel across multiple metrics. They don't just consume content (high likes/comments); they also create it (photos posted) and build influence (high followers). This indicates that the most loyal users are those who are deeply invested in both creating and participating in the community.

Recommendation:

Launch a Tiered "Community Ambassador" Program like a formal, multi-level recognition program.

* Gold Tier (Top 10): Grant these users an exclusive "Ambassador" badge on their profile. Offer them direct access to a private feedback channel with the product development team, giving them a real stake in the platform's future. Provide early access to all new features a week before public release.
* Silver Tier (Top 50): Reward these users with a "Community Leader" badge and smaller perks like exclusive photo filters or profile themes.

1. **For inactive users, what strategies would you recommend to re-engage them and encourage them to start posting or engaging again?**

Code:

with like\_per\_user as (

SELECT

user\_id, COUNT(photo\_id) AS num\_likes

FROM

likes

GROUP BY 1

),

comments\_per\_user as (

SELECT

user\_id, COUNT(comment\_text) AS num\_coments\_per\_users

FROM

comments

GROUP BY 1

),

Posts\_per\_user as (

SELECT

user\_id, COUNT(image\_url) AS num\_posts

FROM

photos

GROUP BY 1

),

Username\_with\_zero\_Activity\_Engagement as (

SELECT

username,

SUM(num\_posts + num\_coments\_per\_users + num\_likes) AS activity\_level\_users,

SUM(num\_coments\_per\_users + num\_likes) AS Engagement\_rate

FROM

users u

LEFT JOIN

like\_per\_user c1 ON u.id = c1.user\_id

LEFT JOIN

comments\_per\_user c2 ON u.id = c2.user\_id

LEFT JOIN

Posts\_per\_user c3 ON u.id = c3.user\_id

WHERE

c1.user\_id IS NULL

GROUP BY 1

)

SELECT

username AS Inactive\_users

FROM

Username\_with\_zero\_Activity\_Engagement

ORDER BY 1;

Output:

A screenshot of a computer

AI-generated content may be incorrect.

Approach

Our approach to identifying and re-engaging inactive users is methodical, and data driven.

* Defining Inactivity: We first established a clear definition of an "inactive user": any user who has a record in the users table but has zero associated records in the likes, comments, and photos tables. This provides a precise segment of the user base that has shown no engagement whatsoever.
* User Segmentation: The SQL query uses Common Table Expressions (CTEs) to aggregate activity counts (likes, comments, posts) for each user. The final SELECT statement then uses a WHERE clause with IS NULL checks on the LEFT JOIN results to filter this list down to users for whom all these counts are zero, effectively isolating the inactive segment.
* Targeted Strategy Formulation: By creating a definitive list of inactive users, we can move from generic marketing to a targeted re-engagement campaign. The strategies outlined below are designed specifically to address the potential reasons for user dormancy, such as a lack of initial connection, content discovery issues, or a need for motivation.

Insights

The identification of a substantial list of inactive users provides several critical insights into user behaviour and platform health.

* Insight 1: A Significant Portion of the User Base is Untapped: The presence of a long list of inactive users, including names like Aurelie71 and Bartholome.Bernhard, indicates that a considerable percentage of sign-ups do not convert into active participants. This "leaky bucket" is a major area for potential growth if these users can be successfully re-activated. It points to a gap between user acquisition and user retention.
* Insight 2: Initial Onboarding May Be a Friction Point: The fact that these users have accounts, but no recorded activity suggests a potential weakness in the user onboarding process. They may have created an account but failed to find compelling content or users to connect with in their first session, leading to a quick drop-off and no subsequent visits.
* Insight 3: Lack of Social Connection is a Likely Cause of Inactivity: Social platforms thrive on connection. These inactive users likely have few to no followers and are not following anyone, meaning their feed is probably empty or populated with non-relevant generic content. Without an initial social graph, the incentive to return to the platform is extremely low.

Recommendation:

Implement a "Welcome Back" Email and Push Notification Campaign. For example, create a sequence of automated emails and push notifications triggered after a certain period of inactivity (e.g., 14-30 days).

* Email 1: "Trending on Your Feed" - Showcase the most liked and commented-on photos from the past week to highlight the vibrant and engaging content they are missing.
* Push Notification 1: "Your Friends Are Here" - If the user has granted contact access, send a notification suggesting they connect with contacts already on the platform. Social proof is a powerful motivator.
* Email 2: "Did You See This?" - Personalize this email by featuring content with tags related to the user's initial interests (if this data is collected during sign-up) or location.

1. **Which hashtags or content topics have the highest engagement rates? How can this information guide content strategy and ad campaigns?**

Code:

with likes\_per\_posts as (

SELECT

photo\_id, COUNT(user\_id) AS num\_likes

FROM

likes

GROUP BY 1

),

comments\_per\_posts as (

SELECT

photo\_id, COUNT(comment\_text) AS num\_coments\_per\_post

FROM

comments

GROUP BY 1

),

engagement\_rate\_of\_each\_tag as(

SELECT

tag\_name,

SUM(num\_coments\_per\_post + num\_likes) AS Engagement\_rate

FROM

tags t

JOIN

photo\_tags pt ON t.id = pt.tag\_id

JOIN

likes\_per\_posts c1 ON pt.photo\_id = c1.photo\_id

JOIN

comments\_per\_posts c2 ON pt.photo\_id = c2.photo\_id

GROUP BY 1

)

SELECT

\*,

dense\_rank() Over(order by Engagement\_rate desc) as tag\_ranking

FROM

engagement\_rate\_of\_each\_tag;

Output:

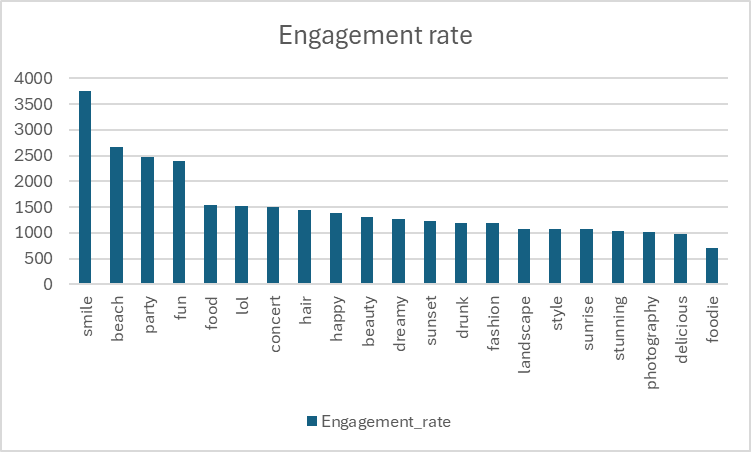
A screenshot of a computer

AI-generated content may be incorrect.

A screenshot of a table

AI-generated content may be incorrect.

Visualization:



Approach:

The approach to determining the highest-engaging hashtags involved several steps:

* Calculate Per-Post Engagement: First, two Common Table Expressions (CTEs), likes\_per\_posts and comments\_per\_posts, were created to aggregate the total number of likes and comments received by each individual photo (photo\_id).
* Aggregate Engagement by Tag: The main CTE, engagement\_rate\_of\_each\_tag, joins the tags table with the photo\_tags junction table. It then LEFT JOINs these results with the likes\_per\_posts and comments\_per\_posts CTEs. This structure links each tag to all the photos it appears on and aggregates the total likes and comments associated with those photos. The SUM() function calculates the total Engagement\_rate (likes + comments) for each tag\_name. COALESCE is used to ensure that photos with tags but zero engagement are still counted appropriately.
* Rank the Tags: The final SELECT statement retrieves all tags and their calculated Engagement\_rate, then uses the DENSE\_RANK() window function to assign a rank based on engagement in descending order. DENSE\_RANK() is chosen to handle potential ties in engagement rates gracefully (assigning the same rank).

Insights:

Social Experiences and Positive Emotions Drive Highest Engagement: The top-performing hashtags are overwhelmingly related to positive social experiences and emotions. #smile leads significantly, followed by tags like #beach, #party, #fun, #happy. This strongly indicates that content depicting enjoyable moments, social gatherings, and positive feelings resonates most powerfully with the user base, driving likes and comments.

Recommendation:

Prioritize Content Themes Around Positive Social Experiences:

* Detailed Action (Content Strategy): Encourage users (and create official platform content) centered around the top-performing themes: happiness, social events, travel (especially beaches), and fun activities. Run themed photo challenges like "#BestSmile," "#PartyMoments," or "#BeachLife" to stimulate content creation around these high-engagement topics. Feature user-generated content prominently that uses these top hashtags.
* Detailed Action (Ad Campaigns): Advertisers should align their creative content and messaging with these positive themes. Ads featuring smiling people, social settings, or aspirational travel/leisure activities are likely to perform better. Targeting users who frequently engage with these top hashtags can improve ad relevance and effectiveness.

1. **Are there any patterns or trends in user engagement based on demographics (age, location, gender) or posting times? How can these insights inform targeted marketing campaigns?**

Code:

SELECT

HOUR(p.created\_dat) AS post\_hour,

DAYOFWEEK(p.created\_dat) AS post\_day,

COUNT(DISTINCT p.id) AS total\_photos\_posted,

COUNT(DISTINCT l.photo\_id) AS total\_likes\_received,

COUNT(DISTINCT c.id) AS total\_comments\_made

FROM photos p

JOIN likes l ON p.id = l.photo\_id

JOIN comments c ON p.id = c.photo\_id

GROUP BY post\_hour, post\_day

ORDER BY post\_hour, post\_day;

Output:



Approach:

The methodology for analysing engagement timing involved these steps:

* Extract Time Units: The HOUR() and DAYOFWEEK() functions were used on the created\_dat column of the photos table to extract the specific hour (0-23) and day of the week (1=Sunday, 7=Saturday) each photo was posted.
* Join Engagement Data: The photos table was joined with the likes and comments tables using the photo\_id. Critically, the use of INNER JOIN (implied by JOIN) filters the results to include only those photos that have received both likes and comments.
* Aggregate Metrics: The COUNT(DISTINCT ...) function was used to count the unique number of photos posted, the unique photos receiving likes, and the unique comments made within each specific post\_hour and post\_day combination.
* Group and Order: The GROUP BY clause aggregates these counts for each unique hour/day combination, and ORDER BY sorts the results chronologically.

Insights:

Peak Engagement Identified on Saturday Afternoons: The query output shows a single, highly concentrated period of engagement. Activity significantly peaks at Hour 16 (4:00 PM - 4:59 PM) on Day 7 (Saturday). During this specific hour, 257 photos were posted that received both likes and comments, accumulating 257 likes (indicating each photo received at least one like, as filtered by the JOIN) and a remarkably high 7488 comments.

Recommendation:

Optimize Posting Schedule for Peak Time

* Detailed Action (Content Strategy): Schedule key content releases, announcements, and user prompts (e.g., asking questions in captions) for Saturdays around 4:00 PM. This maximizes the potential for visibility, likes, and especially comment interaction, given the exceptionally high comment volume observed. Run weekend-specific campaigns or features that encourage posting during this high-traffic window.
* Detailed Action (Ad Campaigns): Concentrate ad spend and campaign flights during this peak period (Saturday late afternoon). Ads shown at this time are likely to reach the most active and engaged segment of the user base. Tailor ad creatives to weekend themes or activities that resonate with users active on a Saturday afternoon.

1. **Based on follower counts and engagement rates, which users would be ideal candidates for influencer marketing campaigns? How would you approach and collaborate with these influencers?**

Code:

WITH TotalLikes AS (

SELECT u.id, COUNT(distinct l.photo\_id) AS total\_likes

FROM users u

LEFT JOIN likes l ON u.id = l.user\_id

GROUP BY u.id),

TotalComments AS (

SELECT u.id, COUNT(distinct c.photo\_id) AS total\_comments

FROM users u

LEFT JOIN comments c ON u.id = c.user\_id

GROUP BY u.id),

PhotosPosted AS (

SELECT user\_id, COUNT(id) AS total\_photos\_posted

FROM photos

GROUP BY user\_id),

Followers AS (

SELECT followee\_id AS user\_id, COUNT(follower\_id) AS total\_followers

FROM follows

GROUP BY followee\_id)

SELECT u.id AS user\_id, u.username,

COALESCE(tl.total\_likes, 0) AS total\_likes,

COALESCE(tc.total\_comments, 0) AS total\_comments,

COALESCE(pp.total\_photos\_posted, 0) AS total\_photos\_posted,

COALESCE(f.total\_followers, 0) AS total\_followers,

((COALESCE(tl.total\_likes, 0) + COALESCE(tc.total\_comments, 0))/(COALESCE(pp.total\_photos\_posted, 0))) as engagement\_rate

FROM users u

JOIN TotalLikes tl ON u.id = tl.id

JOIN TotalComments tc ON u.id = tc.id

JOIN PhotosPosted pp ON u.id = pp.user\_id

JOIN Followers f ON u.id = f.user\_id

group by u.id

having total\_photos\_posted >0

ORDER BY engagement\_rate desc, total\_followers desc,total\_photos\_posted desc

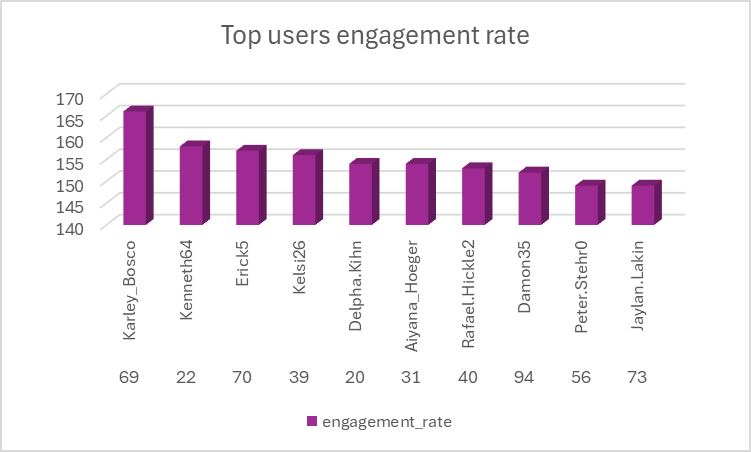
limit 10;

Output:

A screenshot of a computer

AI-generated content may be incorrect.

Visualization:



Approach:

The methodology used to identify potential influencers focuses on a calculated engagement rate relative to the number of posts, combined with follower count as a measure of reach.

* Aggregate Core Metrics: CTEs were used to calculate total\_likes, total\_comments, total\_photos\_posted, and total\_followers for each user independently.
* Join User Data: The main SELECT statement joins the users table with these aggregated metrics using LEFT JOINs to ensure all users are considered initially.
* Calculate Engagement Rate: An engagement\_rate is calculated per user using the formula: (total\_likes + total\_comments) / total\_photos\_posted. This metric represents the average number of interactions (likes + comments) received per photo posted by the user. A CASE statement ensures safe division, although the subsequent filter makes it somewhat redundant here.
* Filter for Active Posters: A WHERE clause (COALESCE(pp.total\_photos\_posted, 0) > 0) is applied to filter out users who have not posted any photos, as the engagement rate calculation is meaningless without posts.
* Rank Potential Influencers: The results are ordered primarily by engagement\_rate (descending) to prioritize users whose content generates the most interaction per post. total\_followers (descending) and total\_photos\_posted (descending) are used as secondary and tertiary sorting criteria to differentiate users with similar engagement rates, prioritizing those with larger reach and higher content volume. The LIMIT 10 clause restricts the output to the top candidates.

Insights:

Extremely High Engagement Rates Driven by Low Post Counts: The top users, such as Karley\_Bosco (166.00 rate), Kenneth64 (158.00 rate), and Erick5 (157.00 rate), exhibit exceptionally high engagement rates. However, the output table reveals that all top 10 users have only posted one photo (total\_photos\_posted = 1). This significantly inflates the engagement rate calculation, as their entire like/comment history is attributed to that single post. While this single post was clearly very engaging, this metric might not reliably predict future performance or sustained influence.

Recommendation:

Identify "High-Potential" Candidates (with Caveats)

* Detailed Action: The top 10 users (Karley\_Bosco, Kenneth64, Erick5, etc.) identified by the query should be considered "high-potential" candidates due to the demonstrated high engagement on their single post, but not yet confirmed influencers. Their high engagement rate indicates their content resonated strongly, and their follower count (76) provides a decent initial reach.
* Next Steps: Before outreach, manually review their profile and single post. Assess the content quality, the nature of the comments (are they genuine interactions?), and the user's profile presentation.

1. **Based on user behaviour and engagement data, how would you segment the user base for targeted marketing campaigns or personalized recommendations?**

Code:

SELECT

u.id AS user\_id,

u.username,

COALESCE(p.total\_posts, 0) AS total\_posts,

COALESCE(l.total\_likes, 0) + COALESCE(c.total\_comments, 0) AS total\_engagement,

CASE

WHEN COALESCE(l.total\_likes, 0) + COALESCE(c.total\_comments, 0) > 150 THEN 'Highly Engaged'

WHEN COALESCE(l.total\_likes, 0) + COALESCE(c.total\_comments, 0) BETWEEN 100 AND 150 THEN 'Moderately Engaged'

ELSE 'Less Engaged' END AS user\_category,

CASE

WHEN YEAR(u.created\_at) >= 2017 THEN 'New\_User'

ELSE 'Old\_User' END AS user\_join\_status

FROM users u

LEFT JOIN (

SELECT user\_id, COUNT(\*) AS total\_likes

FROM ig\_clone.likes

GROUP BY user\_id) l ON u.id = l.user\_id

LEFT JOIN (

SELECT user\_id, COUNT(\*) AS total\_comments

FROM comments

GROUP BY user\_id) c ON u.id = c.user\_id

LEFT JOIN (

SELECT user\_id, COUNT(\*) AS total\_posts

FROM photos

GROUP BY user\_id) p ON u.id = p.user\_id

GROUP BY u.id

HAVING total\_posts > 0

ORDER BY total\_engagement DESC, user\_category;

Output:

A screenshot of a computer screen

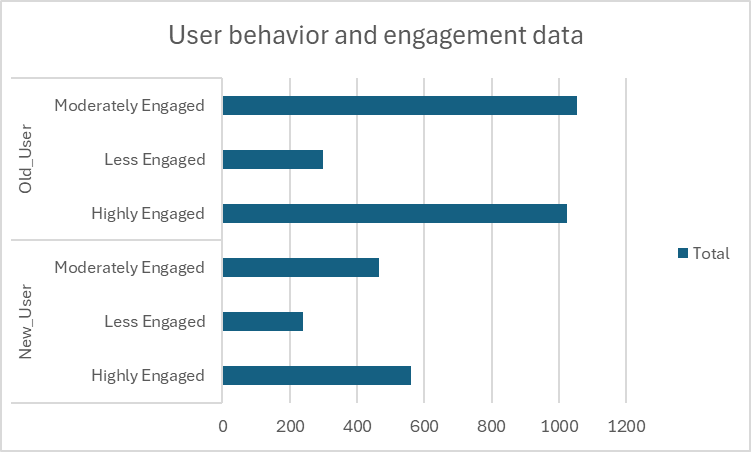
AI-generated content may be incorrect. A screenshot of a computer

AI-generated content may be incorrect. A screenshot of a computer screen

AI-generated content may be incorrect. A screenshot of a data

AI-generated content may be incorrect.

Visualization:



Approach:

* Calculate Total Engagement: The query first aggregates the total number of likes and comments made by each user using CTEs (though implemented here via subqueries in LEFT JOINs) and LEFT JOINs to the users table. The sum of these two metrics (total\_engagement) serves as the primary measure of user activity.
* Define Engagement Tiers: A CASE statement (user\_category) categorizes users into three distinct engagement tiers based on their total\_engagement score:
* Highly Engaged: > 150 total interactions (likes + comments)
* Moderately Engaged: 100 to 150 total interactions
* Less Engaged: < 100 total interactions
* Define User Tenure: A second CASE statement (user\_join\_status) categorizes users based on the year they joined (created\_at field in the users table):
* New\_User: Joined in 2017 or later
* Old\_User: Joined before 2017
* Filter for Active Posters: A HAVING clause ensures that only users who have posted at least one photo (total\_posts > 0) are included in the final segmentation, focusing the analysis on users who contribute content.
* Combine Segments: By combining user\_category and user\_join\_status, we create six distinct user segments (e.g., Highly Engaged Old User, Moderately Engaged New User).

Insights:

Old Users Dominate High Engagement Tiers: The data clearly shows that 'Old\_User' segments contain significantly more users in both the 'Highly Engaged' (1025 users) and 'Moderately Engaged' (1054 users) categories compared to 'New\_User' segments (561 and 465 users, respectively). This suggests that users who have been on the platform longer tend to be more active, or that retaining high engagement over time is crucial.

Recommendation:

Target Highly Engaged Old Users (Loyal Core)

Marketing: Focus on loyalty programs, exclusive feature previews, referral bonuses, and brand ambassador opportunities. Acknowledge their long-term commitment.

Recommendations: Recommend niche or advanced content related to their established interests. Suggest new users with similar interests for them to mentor or connect with. Highlight opportunities for them to lead community initiatives.

1. **If data on ad campaigns (impressions, clicks, conversions) is available, how would you measure their effectiveness and optimize future campaigns?**

Note: The answer is based on hypothetical situation as ad campaign data is not available, therefore there will be no code, visualization and output.

Approach

Approach:

The approach involves systematically tracking, calculating, and analysing key performance indicators (KPIs) across various dimensions of the ad campaigns.

* Data Collection & Integration: Ensure reliable tracking of impressions, clicks, spend, conversions, and conversion values (if applicable) across all ad platforms and campaigns. Integrate this data into a centralized database or analytics platform.
* Define Key Metrics: Establish the core metrics for measuring success based on campaign objectives (e.g., awareness objective might focus on impressions/reach, while a sales objective focuses on CPA/ROAS). Key metrics include:
* Impressions: Number of times ads were displayed.
* Clicks: Number of times ads were clicked.
* CTR (Click-Through Rate): (Clicks / Impressions) \* 100%. Measures ad relevance and creative effectiveness.
* CPC (Cost Per Click): Spend / Clicks. Measures cost efficiency of driving traffic.
* Conversions: Desired actions taken after clicking (e.g., sign-up, purchase, lead).
* CVR (Conversion Rate): (Conversions / Clicks) \* 100%. Measures effectiveness of landing page and offer post-click.
* CPA (Cost Per Acquisition/Conversion): Spend / Conversions. Measures cost efficiency of achieving desired outcomes.
* ROAS (Return on Ad Spend): (Conversion Value / Spend). Measures overall profitability (requires tracking revenue/value).
* A/B Testing: Implement controlled experiments to test variations in ad creative, copy, targeting, landing pages, or bidding strategies to isolate factors driving performance.
* Reporting & Iteration: Regularly report on findings and use the insights gained to make data-driven decisions for optimizing ongoing and future campaigns.

Insights:

Identify Top/Bottom Performing Elements: Clearly see which campaigns, ad sets, specific ads, or audience segments are driving the best results (e.g., lowest CPA, highest ROAS) and which are underperforming and wasting budget. For example, Campaign A might have a CPA of $5 while Campaign B has a CPA of $50 for the same conversion type.

Recommendation:

Systematically shift budget away from underperforming campaigns, ad sets, audiences, or placements towards those consistently delivering results against the defined KPIs (e.g., lower CPA, higher ROAS). If Ad Set X has a CPA 3x higher than Ad Set Y for the same goal, gradually reduce X's budget and increase Y's. Don't cut poor performers immediately; investigate why they underperform first – perhaps a simple targeting or creative tweak is needed. Analyze trends over a sufficient period (e.g., 7-14 days) before making drastic changes.

1. **How can you use user activity data to identify potential brand ambassadors or advocates who could help promote Instagram's initiatives or events?**

Code:

WITH Total Likes AS (

SELECT u.id, COUNT(distinct l.photo\_id) AS total\_likes

FROM users u

LEFT JOIN likes l ON u.id = l.user\_id

GROUP BY u.id

),

TotalComments AS (

SELECT u.id, COUNT(distinct c.photo\_id) AS total\_comments

FROM users u

LEFT JOIN comments c ON u.id = c.user\_id

GROUP BY u.id

),

PhotosPosted AS (

SELECT user\_id, COUNT(id) AS total\_photos\_posted

FROM photos

GROUP BY user\_id

),

Followers AS (

SELECT followee\_id AS user\_id, COUNT(follower\_id) AS total\_followers

FROM follows

GROUP BY followee\_id

)

SELECT u.id AS user\_id, u.username,

COALESCE(tl.total\_likes, 0) AS total\_likes,

COALESCE(tc.total\_comments, 0) AS total\_comments,

COALESCE(pp.total\_photos\_posted, 0) AS total\_photos\_posted,

COALESCE(f.total\_followers, 0) AS total\_followers,

-- Calculate Engagement Rate: (Total Likes + Total Comments) / Total Photos Posted

CASE

WHEN COALESCE(pp.total\_photos\_posted, 0) > 0 THEN

((COALESCE(tl.total\_likes, 0) + COALESCE(tc.total\_comments, 0)) \* 1.0 / COALESCE(pp.total\_photos\_posted, 0))

ELSE 0

END AS engagement\_rate

FROM users u

LEFT JOIN TotalLikes tl ON u.id = tl.id

LEFT JOIN TotalComments tc ON u.id = tc.id

LEFT JOIN PhotosPosted pp ON u.id = pp.user\_id

LEFT JOIN Followers f ON u.id = f.user\_id

WHERE COALESCE(pp.total\_photos\_posted, 0) > 0 -- Filter for users who have posted

GROUP BY u.id, u.username, tl.total\_likes, tc.total\_comments, pp.total\_photos\_posted, f.total\_followers

ORDER BY engagement\_rate DESC, total\_followers DESC, total\_photos\_posted DESC

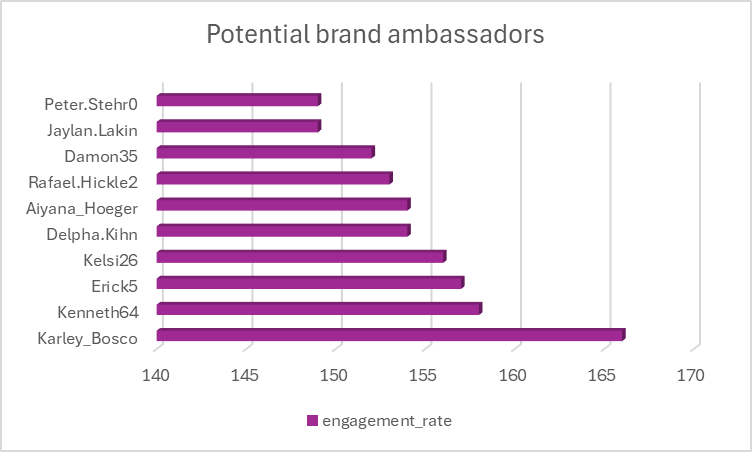
LIMIT 10;

Output:

A screenshot of a computer

AI-generated content may be incorrect.

Visualization:



Approach:

The methodology used to identify potential influencers focuses on a calculated engagement rate relative to the number of posts, combined with follower count as a measure of reach.

* Aggregate Core Metrics: CTEs were used to calculate total\_likes, total\_comments, total\_photos\_posted, and total\_followers for each user independently.
* Join User Data: The main SELECT statement joins the users table with these aggregated metrics using LEFT JOINs to ensure all users are considered initially.
* Calculate Engagement Rate: An engagement\_rate is calculated per user using the formula: (total\_likes + total\_comments) / total\_photos\_posted. This metric represents the average number of interactions (likes + comments) received per photo posted by the user. A CASE statement ensures safe division.
* Filter for Active Posters: A WHERE clause (COALESCE(pp.total\_photos\_posted, 0) > 0) is applied to filter out users who have not posted any photos, as the engagement rate calculation is meaningless without posts.
* Rank Potential Influencers: The results are ordered primarily by engagement\_rate (descending) to prioritize users whose content generates the most interaction per post. total\_followers (descending) and total\_photos\_posted (descending) are used as secondary and tertiary sorting criteria to differentiate users with similar engagement rates, prioritizing those with larger reach and higher content volume. The LIMIT 10 clause restricts the output to the top candidates.

Insights:

Extremely High Engagement Rates Driven by Low Post Counts: The top users, such as Karley\_Bosco (166.00 rate), Kenneth64 (158.00 rate), and Erick5 (157.00 rate), exhibit exceptionally high engagement rates. However, the output table reveals that all top 10 users have only posted one photo (total\_photos\_posted = 1). This significantly inflates the engagement rate calculation, as their entire like/comment history is attributed to that single post. While this single post was clearly very engaging, this metric might not reliably predict future performance or sustained influence.

Recommendation:

Identify "High-Potential" Candidates

* Detailed Action: The top 10 users (Karley\_Bosco, Kenneth64, Erick5, etc.) identified by the query should be considered "high-potential" candidates due to the demonstrated high engagement on their single post, but not yet confirmed influencers. Their high engagement rate indicates their content resonated strongly, and their follower count (76) provides a decent initial reach.
* Next Steps: Before outreach, manually review their profile and single post. Assess the content quality, the nature of the comments (are they genuine interactions?), and the user's profile presentation.

1. **How would you approach this problem, if the objective and subjective questions weren't given?**

Note: The answer is based on hypothetical situation if the objective and subjective questions weren’t given, therefore there will be no code, visualization and output.

Approach:

A comprehensive Exploratory Data Analysis (EDA) without pre-defined questions involves the following systematic steps:

* Data Familiarization:

Examine the contents of each table (users, photos, likes, comments, follows, tags, photo\_tags) individually to understand the raw data points available.

* Data Quality Assessment:

Check for NULL values in key columns across all tables to identify missing data.

Check for duplicate entries, particularly in tables like users (duplicate usernames) or junction tables where primary key constraints should prevent them (e.g., likes, follows, photo\_tags).

* Performance Optimization (Optional but Recommended):

Consider creating database views (CREATE VIEW) for frequently used joins or aggregations (e.g., a view combining user info with their post/like/comment counts) to simplify and potentially speed up subsequent queries.

* Initial User Activity Profiling:

Analyze follower counts (follows table) and overall activity levels (counts from likes, comments, photos) per user.

Identify potentially inactive users (those with zero posts, likes, comments, and possibly zero followers/following).

Consider creating a summary table or view categorizing users (e.g., 'Active', 'Inactive') based on these initial findings for easier querying later.

* Bot/Anomaly Detection (Preliminary):

Examine users with unusually high follower counts but disproportionately low engagement (likes/comments on their posts, if applicable) or low posting activity. This might indicate fake followers, although further analysis is needed for confirmation. Exclude confirmed bots if necessary.

* User Segmentation & Ranking:

Based on combined activity/engagement metrics (posts, likes, comments), segment the verified active users into tiers (e.g., 'Most Active', 'Mid-Active', 'Low Active') using ranking functions (RANK(), NTILE()) or defined thresholds.

* Content and Tag Analysis:

Determine which tags (tags.tag\_name) are most frequently used overall.

Analyze which tags correlate with the highest engagement (average likes/comments on photos using those tags).

Identify which categories of users (e.g., 'Most Active', 'New Users') are driving the usage of top tags or generating the most engagement.

Insights (Potential Findings from EDA):

An open-ended EDA on this dataset could uncover various insights crucial for platform strategy:

* User Activity Segmentation: Confirmation of distinct user groups: highly active "power users" (small percentage driving most activity), moderately active users, casual browsers/likers, and a potentially large segment of inactive users. Understanding the relative size and behavior of these groups is key.
* Content Performance Drivers: Identification of factors correlating with high engagement (likes/comments) on photos. This might include specific content themes (via tags like #food, #sunset), posting time, number of tags used, or the poster's follower count.
* Network Dynamics: Understanding the structure of the follow graph. Are there distinct communities? How prevalent is reciprocal following? Identifying users who act as hubs (followed by many) versus those who are more peripheral.
* Temporal Engagement Patterns: Clear evidence of peak activity times (e.g., evenings, weekends) versus lulls, providing insight into user habits and optimal times for platform interaction or content promotion.
* Tagging Behavior & Impact: Discovering how users utilize tags. Are there popular tag combinations? Do posts with more tags receive more engagement, up to a certain point? Which specific tags are associated with the highest average likes or comments?

Recommendation (General Strategic Directions from EDA)

The insights generated from EDA can inform various strategic initiatives:

* Develop Segment-Specific Strategies: Based on the identified user activity segments (power users, casual users, inactive users), tailor features, communication, and marketing efforts. For example, create loyalty programs for power users and re-engagement campaigns for inactive ones.
* Optimize Content Strategy & Recommendations: Promote content types and topics (identified via tag analysis) that resonate most with users. Refine content recommendation algorithms to surface high-engagement content and connect users with relevant creators or topics more effectively.
* Enhance Community Building Features: If network analysis reveals opportunities, introduce features that encourage connections, group formation, or interaction between users with shared interests or follower networks. Highlight users with high follow-back rates as potential connections.
* Guide Feature Development: Insights into user behavior can highlight unmet needs or friction points. For example, if users frequently use similar tag combinations, a feature suggesting tag bundles might be valuable. If engagement drops after a certain tenure, investigate potential reasons and develop features to maintain long-term interest.
* Inform Marketing & Communication Timing: Schedule marketing campaigns, push notifications, and major feature announcements during identified peak user activity times to maximize reach and impact.

1. **Assuming there's a "User\_Interactions" table tracking user engagements, how can you update the "Engagement\_Type" column to change all instances of "Like" to "Heart" to align with Instagram's terminology?**

Code:

CREATE TABLE User\_Interactions(

id INT AUTO\_INCREMENT UNIQUE PRIMARY KEY,

username VARCHAR(250) NOT NULL,

Engagement\_Type varchar(250) not null);

INSERT INTO User\_Interactions (username,Engagement\_Type ) VALUES

('Kenton\_Kirlin', 'Like'),

('Andre\_Purdy85', 'Comments'),

('Harley\_Lind18', 'Comments'),

('Arely\_Bogan63', 'Comments'),

('Aniya\_Hackett', 'Like'),

('Travon.Waters', 'Like'),

('Kasandra\_Homenick', 'Comments'),

('Tabitha\_Schamberger11', 'Comments'),

('Gus93', 'Like'),

('Presley\_McClure', 'Comments'),

('Justina.Gaylord27', 'Like'),

('Dereck65', 'Comments'),

('Alexandro35', 'Comments'),

('Jaclyn81', 'Comments'),

('Billy52', 'Like'),

('Annalise.McKenzie16', 'Comments'),

('Norbert\_Carroll35', 'Like'),

('Odessa2', 'Comments'),

('Hailee26', 'Comments'),

('Delpha.Kihn', 'Like'),

('Rocio33', 'Like'),

('Kenneth64', 'Like'),

('Eveline95', 'Like'),

('Maxwell.Halvorson', 'Like'),

('Tierra.Trantow', 'Like'),

('Josianne.Friesen', 'Like'),

('Darwin29', 'Like'),

('Dario77', 'Like'),

('Jaime53', 'Comments'),

('Kaley9', 'Comments'),

('Aiyana\_Hoeger', 'Like'),

('Irwin.Larson', 'Like'),

('Yvette.Gottlieb91', 'Comments'),

('Pearl7', 'Like'),

('Lennie\_Hartmann40', 'Comments'),

('Ollie\_Ledner37', 'Like'),

('Yazmin\_Mills95', 'Comments'),

('Jordyn.Jacobson2', 'Like'),

('Kelsi26', 'Like'),

('Rafael.Hickle2', 'Comments'),

('Mckenna17', 'Like'),

('Maya.Farrell', 'Comments'),

('Janet.Armstrong', 'Like'),

('Seth46', 'Comments'),

('David.Osinski47', 'Like'),

('Malinda\_Streich', 'Comments'),

('Harrison.Beatty50', 'Like'),

('Granville\_Kutch', 'Comments'),

('Morgan.Kassulke', 'Like'),

('Gerard79', 'Comments'),

('Mariano\_Koch3', 'Comments'),

('Zack\_Kemmer93', 'Like'),

('Linnea59', 'Comments'),

('Duane60', 'Comments'),

('Meggie\_Doyle', 'Like'),

('Peter.Stehr0', 'Comments'),

('Julien\_Schmidt', 'Like'),

('Aurelie71', 'Comments'),

('Cesar93', 'Comments'),

('Sam52', 'Like'),

('Jayson65', 'Comments'),

('Ressie\_Stanton46', 'Like'),

('Elenor88', 'Comments'),

('Florence99', 'Like'),

('Adelle96', 'Comments'),

('Mike.Auer39', 'Comments'),

('Emilio\_Bernier52', 'Like'),

('Franco\_Keebler64', 'Comments'),

('Karley\_Bosco', 'Like'),

('Erick5', 'Comments'),

('Nia\_Haag', 'Like'),

('Kathryn80', 'Comments'),

('Jaylan.Lakin', 'Like'),

('Hulda.Macejkovic', 'Comments'),

('Leslie67', 'Comments'),

('Janelle.Nikolaus81', 'Like'),

('Donald.Fritsch', 'Comments'),

('Colten.Harris76', 'Like'),

('Katarina.Dibbert', 'Comments'),

('Darby\_Herzog', 'Comments'),

('Esther.Zulauf61', 'Like'),

('Aracely.Johnston98', 'Comments'),

('Bartholome.Bernhard', 'Comments'),

('Alysa22', 'Comments'),

('Milford\_Gleichner42', 'Like'),

('Delfina\_VonRueden68', 'Comments'),

('Rick29', 'Like'),

('Clint27', 'Comments'),

('Jessyca\_West', 'Comments'),

('Esmeralda.Mraz57', 'Like'),

('Bethany20', 'Comments'),

('Frederik\_Rice', 'Comments'),

('Willie\_Leuschke', 'Like'),

('Damon35', 'Comments'),

('Nicole71', 'Comments'),

('Keenan.Schamberger60', 'Like'),

('Tomas.Beatty93', 'Comments'),

('Imani\_Nicolas17', 'Like'),

('Alek\_Watsica', 'Comments'),

('Javonte83', 'Like');

SET SQL\_SAFE\_UPDATES = 0;

update User\_Interactions

set Engagement\_Type = "Heart"

where Engagement\_Type= "Like";

select \* from User\_Interactions;

Output:



A screenshot of a computer

AI-generated content may be incorrect.

A screenshot of a computer

AI-generated content may be incorrect.

A screenshot of a computer

AI-generated content may be incorrect.

A screenshot of a computer

AI-generated content may be incorrect.

A screenshot of a computer

AI-generated content may be incorrect.

Approach:

The approach taken to update the terminology involves several standard SQL operations:

* Table Definition (CREATE TABLE): A hypothetical table User\_Interactions is defined to store basic interaction data, including a unique ID, the username, and the type of engagement.
* Data Population (INSERT INTO): Sample data is inserted into the table, including various Engagement\_Type values, specifically "Like" and "Comments".
* Disabling Safe Updates (SET SQL\_SAFE\_UPDATES = 0): This crucial step temporarily disables a safety feature in some SQL environments (like MySQL Workbench) that prevents UPDATE or DELETE operations without a WHERE clause referencing a key column or a LIMIT clause. Since we intend to update all rows matching Engagement\_Type = "Like", safe updates need to be disabled.
* Data Modification (UPDATE ... SET ... WHERE): The core logic resides in the UPDATE statement.
* UPDATE User\_Interactions: Specifies the table to modify.
* SET Engagement\_Type = "Heart": Defines the new value for the target column.
* WHERE Engagement\_Type = "Like": Filters the rows to be updated, ensuring only those with the old terminology ("Like") are affected. This precisely targets the records needing change.
* Verification (SELECT \*): A final SELECT statement retrieves all data from the table, allowing visual confirmation that all instances of "Like" have been successfully changed to "Heart" while other engagement types remain untouched.
* Re-enabling Safe Updates (Recommended): Although not explicitly shown in the final SELECT step, it's best practice to re-enable safe updates (SET SQL\_SAFE\_UPDATES = 1;) after the operation to prevent accidental mass updates in the future.

Insights:

Importance of Consistent Terminology: Aligning database terminology ("Heart") with the user-facing application (e.g., Instagram's heart icon) is crucial for clarity in data analysis, reporting, and communication between technical and non-technical teams. Inconsistent terms can lead to confusion and errors.

Recommendation:

* Standardize Terminology Across Systems: Ensure that terminology used in the database (Engagement\_Type) aligns consistently with the application's UI and any related analytics or reporting platforms. Document this standard terminology.
* Manage Safe Updates: Remember to disable safe updates only when necessary and re-enable them immediately afterward.
* Review Dependent Systems: After changing data values like "Like" to "Heart", review any downstream applications, reports, or dashboards that might query or rely on the Engagement\_Type column. Update these systems to use the new terminology ("Heart") to avoid breaking functionality or displaying incorrect information. Code that hardcoded checks for "Like" will need modification.

**-------------The End---------------**