# Instagram User Analytics :

# SQL Fundamentals

By: Arpan Sharma(Data Analytics Trainee)

## Project Description:-

Instagram user analytics project is about finding insights about user behavior on social media platforms. In this project, I have used the ig_clone database provided by trainity and drawn some conclusions. I have provided insights to topics serving the marketing team and investors of instagram by answering questions asked by the management team. I have used Structured Query Language for data analytics.

## Approach:-

Firstly, I have used SQL queries to process raw data and create a relational database with various tables on mysql workbench. Then, I have used different queries to draw insights and answer all the possible questions. Finally, I combined all the results into this report.

## Tech-Stack Used:-

I have used "**MySQL Server 8.0.32**" for locally hosting the database and "**MySQL Workbench 8.0.24"** as an interface for SQL query editor. It was easy to work with databases and better visualization.

# Project Insights:-

The database included tables namely: comments, follows, likes, photos, photo_tags, tags, users.

| Table Name | No. of Rows | No. of Rows | Name of Columns |
|---|---|---|---|
| comments | 7488 | 5 | id, comment_text, user_id, photo_id, created_at |
| follows | 7623 | 3 | follower_id, followee_id, created_at |
| likes | 8782 | 3 | Photo_id, user_id, created_at |
| photos | 257 | 4 | id, image_url, user_id, created_dat |
| photo_tags | 501 | 2 | photo_id, tag_id |
| tags | 21 | 3 | ld, tag_name, created_at |
| users | 100 | 3 | ld, username, created_at |

### A) Marketing:

1. **Rewarding Most Loyal Users:** People who have been using the platform for the longest time.

   Approach - I displayed top 5 rows for all the columns from the users table and sorted them in ascending order according to created_at column.

   Query :-

```
SELECT
    *

FROM
    users

ORDER BY created_at ASC
LIMIT 5;
```

   Result Grid :-

| 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 |

   Conclusion :- The 5 oldest users of Instagram from the database provided are user ID 80, 67,63,95,38.

2. **Remind Inactive Users to Start Posting:** The users who have never posted a single photo on Instagram.

Approach - I have used left join to join all the values of users table and matching records of photos table. Due to which, users who have not posted any photo had null values in photos table records. To display those users I used a condition to display id and username columns where the image_url is null.

Query -

```
SELECT
    users.id,
    users.username
FROM
    users
        LEFT JOIN
    photos ON users.id = photos.user_id
WHERE
    photos.image_url IS NULL;
```

Result Grid :-

| id | username |
|---|---|
| 5 | Aniya_Hackett |
| 7 | Kasandra_Homenick |
| 14 | Jaclyn81 |
| 21 | Rocio33 |
| 24 | Maxwell.Halvorson |
| 25 | Tierra.Trantow |
| 34 | Pearl7 |
| 36 | Ollie_Ledner37 |
| 41 | Mckenna17 |
| 45 | David.Osinski47 |
| 49 | Morgan.Kassulke |
| 53 | Linnea59 |
| 54 | Duane60 |
| 57 | Julien_Schmidt |
| 66 | Mike.Auer39 |
| 68 | Franco_Keebler64 |
| 71 | Nia_Haag |
| 74 | Hulda.Macejkovic |
| 75 | Leslie67 |
| 76 | Janelle.Nikolaus81 |
| 80 | Darby_Herzog |
| 81 | Esther.Zulauf61 |
| 83 | Bartholome.Bernhard |
| 89 | Jessyca_West |
| 90 | Esmeralda.Mraz57 |
| 91 | Bethany20 |

Conclusion :- Users who have never posted a single photo on Instagram are user id 5,7,14,21,24,25,34,36,41,45,49,53,54,57,66,68,71,74,75,76,80,81,83,89,90,91. Promotional emails to post their 1st photo can be sent to these users.

3. **Declaring Contest Winner:** The user who gets the most likes on a single photo.
   Approach: I have used group by statement to group rows having the same photo_id and

aggregate function count to calculate the total number of likes each photo has from the likes table and also, ordering them in descending order according to no_of_likes with limiting output to 1 for photo_id with most likes. Then, I have used the inner join statement twice to get the details of the winner photo_id. Firstly, the matching photo_id from likes and photos tables are joined and then to get details of those photo ids, inner join is used to find matching users id from photos and users table.

Query :-

```
SELECT
    photos.user_id, users.username, likes.photo_id, photos.image_url,
    photos.created_dat,
    COUNT(*) AS no_of_likes
FROM
    photos
        INNER JOIN
    likes ON likes.photo_id = photos.id
        INNER JOIN
    users ON photos.user_id = users.id
GROUP BY likes.photo_id
ORDER BY no_of_likes DESC
LIMIT 1;
```

Result Grid :-

| users_id | username | photo_id | image_url | no_of_likes |
|---|---|---|---|---|
| **52** | Zack_Kemmer93 | 145 | https://jarret.name | 48 |

Conclusion :- The winner of the contest is user id 52. The user name of the winner is Zack_Kemmer93. The photo_id on which the user has most likes is 145 and the number of likes on the photo is 48. The image url of the photo is https://jarret.name.

4. **Hashtag Researching:** Top 5 hashtags to use in the post to reach the most people on the platform.

Approach: I have used group by statement to group rows having the same tag_id and aggregate function count to calculate the total number of times that tag_id is used and also, ordered them in descending order according to no_of_times_used and limited the output for the first 10 rows. To get details about tags, join is used to find matching records for photo_id from photo_tags and tags tables.

Query:

```
SELECT
    tags.id AS tag_id,
    tags.tag_name,
    COUNT(*) as no_of_times_used
FROM photo_tags
JOIN tags ON photo_tags.tag_id = tags.id
GROUP BY photo_tags.tag_id
ORDER BY no_of_times_used DESC;
LIMIT 10:
```

Result Grid :-

| tag_id | tag_name | no_of_times_used |
|---|---|---|
| **21** | smile | 59 |
| **20** | beach | 42 |
| **17** | party | 39 |
| **13** | fun | 38 |
| **18** | concert | 24 |
| **5** | food | 24 |
| **11** | lol | 24 |
| **15** | hair | 23 |
| **12** | happy | 22 |
| **8** | beauty | 20 |

Conclusion: The top 5 most commonly used hashtags on the platform are smile, beach, party, fun and for 5 th position there is tie among concert, food and lol.

5. **Launch AD Campaign:** The best day to launch ADs, that is the day of the week on which most users register. Provide insights on when to schedule an ad campaign

   Approach : Firstly, I have used the DAYNAME function which returns the name of weekday for a given date to get day_of_registration for users.Then, I have used group by statement to group rows having the same day_of_registration and used count as aggregate function to calculate total number of users registered on the given day.

   Query :-

```
SELECT
        DAYNAME(created_at) as day_of_registration,
        COUNT(*) as no_of_users_registered
FROM users
GROUP BY day_of_registration
```

   Result grid :-

| day_of_registration | no_of_users_registered |
|---|---|
| Thursday | 16 |
| Sunday | 16 |
| Tuesday | 14 |
| Saturday | 12 |
| Wednesday | 13 |
| Monday | 14 |
| Friday | 15 |

   Conclusion: The ad campaign can be scheduled on Thursday or Sunday as most users registered on these two days.

**B) Investor Metrics:**

1. **User Engagement:** How many times does an average user posts on Instagram. Also, provide the total number of photos on Instagram/total number of users

   Approach: I have used a nested select statement. I have used the count aggregate function to calculate the total number of users and total number of photos posted by all the users. Then simply divided two of them to calculate average_posts_per_user

   Query:

```
SELECT
   (SELECT
        COUNT(id)
    FROM
        photos) / (SELECT
        COUNT(id)
    FROM
        users) AS average_post_per_user
```

   Result Grid :-

| average_post_per_user |
|---|
| 2.5700 |

   Conclusion: The average number post per user is 2.570.

2. **Bots & Fake Accounts**: The users (bots) who have liked every single photo on the site.

Approach: I have used group by statement to group rows having the same user_id in likes table and aggregate using count function to calculate total no. of photos liked by that user. Then, I used another sorting function 'Having' to analyse the condition and return rows having the total number of photos that is 257 in the no_of_photo_liked column. I used the inner join statement to get all details of those user_id from the likes table from the users table.

Query:

```
SELECT
    users.id,
    users.username,
    COUNT(*) AS no_of_photo_liked
FROM
    likes
        INNER JOIN
    users ON users.id = likes.user_id
GROUP BY likes.user_id
HAVING no_of_photo_liked = (SELECT
        COUNT(*)
    FROM
        photos)
```

Result Grid :-

| user_id | username | no_of_photo_liked |
|---|---|---|
| 5 | Aniya_Hackett | 257 |
| 14 | Jaclyn81 | 257 |
| 21 | Rocio33 | 257 |
| 24 | Maxwell.Halvorson | 257 |
| 36 | Ollie_Ledner37 | 257 |
| 41 | Mckenna17 | 257 |
| 54 | Duane60 | 257 |
| 57 | Julien_Schmidt | 257 |
| 66 | Mike.Auer39 | 257 |
| 71 | Nia_Haag | 257 |
| 75 | Leslie67 | 257 |
| 76 | Janelle.Nikolaus81 | 257 |
| 91 | Bethany20 | 257 |

Conclusion : The bots(users) who have liked all the photos on instagram platform are user_id 5,14,21,24,36,41,54,57,66,71,75,76,91.

# Result:-

I have answered all the questions asked by the management team in this project and explained the result grid and conclusion under the project insights part. While doing the project I applied my learning of SQL fundamentals and use of aggregate function, sorting functions, operators and join operations.