

# Instagram User Analytics

SQL Fundamentals

# Project Description

- ▶ User analytics is the process of tracking how users interact with digital products to derive business insights for marketing, product and development teams.
- ▶ These insights are used by teams across the organization to launch new marketing campaigns, determine the features to develop for the app, measure user engagement to track app success, and provide an overview while supporting business growth. improve your personal experience.
- ▶ Here we are working with Instagram's product team, and the product manager asks you to provide insight on a question from the executive team.

# Tech-Stack Used

- For solving the tasks I have used MySQL Workbench 8.0 CE



MySQL Workbench is the official graphical user interface [GUI] tool for MySQL. It allows you to design, create and browse your database schemas, work with database objects and insert data as well as design and run SQL queries to work with stored data. You can also migrate schemas and data from other database vendors to your MySQL database.

# Marketing

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



**Task:** Find the 5 oldest users of the Instagram from the database provided.

**Approach:** Selecting all the information about users by using Wildcard Character(\*) from 'users' table and order it in ascending order by using ASC keyword, and hence limit our result to 5 as required by using LIMIT clause.

## ➤ Query Written

```
# MARKETING
-- 1) Rewarding Most Loyal Users: Find 5 oldest users
SELECT
    *
FROM
    users
ORDER BY created_at ASC
LIMIT 5;
```

## ✓ Output Obtained

Result Grid   Filter Rows: <input type="text"/> Ed			
	id	username	created_at
▶	80	Darby_Herzog	2016-05-06 00:14:21
	67	Emilio_Bernier52	2016-05-06 13:04:30
	63	Elenor88	2016-05-08 01:30:41
	95	Nicole71	2016-05-09 17:30:22
	38	Jordyn.Jacobson2	2016-05-14 07:56:26
*	NULL	NULL	NULL

# Marketing

- 2) **Remind Inactive Users to Start Posting:** By sending them promotional emails to post their 1st photo.

**Task:** Find the users who have never posted a single photo on Instagram

**Approach:** To find the users with no post, I firstly selected the username and user id from the users table , joining the users table to photos table on attribute user id to create a link between users and photos, hence finally finding the users with no posts using **IS NULL** clause which returns the records with null values, in our case users with no posts or photos.

## ➤ Query Written

```
-- 2) Remind Inactive Users to Start Posting: User with 0 posts.
```

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

## ✓ Output Obtained

▶ Aniya_Hackett	5	Mike.Auer39	66
Kasandra_Homenick	7	Franco_Keebler64	68
Jadyn81	14	Nia_Haag	71
Rocio33	21	Hulda.Macejkovic	74
Maxwell.Halvorson	24	Leslie67	75
Tierra.Trantow	25	Janelle.Nikolaus81	76
Pearl7	34	Darby_Herzog	80
Ollie_Ledner37	36	Esther.Zulauf61	81
Mckenna17	41	Bartholome.Bernhard	83
David.Osinski47	45	Jessyca_West	89
Morgan.Kassulke	49	Esmeralda.Mraz57	90
Linnea59	53	Bethany20	91
Duane60	54		
Julien_Schmidt	57		

# Marketing

- 3) **Declaring Contest Winner:** The team started a contest and the user who gets the most likes on a single photo will win the contest now they wish to declare the winner.

**Task:** Identify the winner of the contest and provide their details to the team

**Approach:** In order to find user with most liked photo we need to combine three tables i.e., users, likes, photos on user\_id attribute, to print or get the name and details about the user I used **SELECT** command to get the id, username, photo id and the URL of photo, and the number of likes on photos, the number of photos we counted by using **COUNT** aggregate function, at last I have limited the result to 1 by using the **LIMIT** clause to find the top person.

## ➤ Query Written

```
-- 3) Declaring Contest Winner: Most liked photo
SELECT
    users.id,
    users.username,
    photos.id,
    photos.image_url,
    COUNT(*) AS Total_Likes
FROM photos
INNER JOIN likes on likes.photo_id = photos.id
INNER JOIN users on photos.user_id = users.id
GROUP BY photos.id
ORDER BY Total_Likes DESC
LIMIT 1;
```

## ✓ Output Obtained

	id	username	id	image_url	Total_Likes
▶	52	Zack_Kemmer93	145	https://jarret.name	48

# Marketing

- 4) **Hashtag Researching:** A partner brand wants to know, which hashtags to use in the post to reach the most people on the platform.

**Task:** Identify and suggest the top 5 most commonly used hashtags on the platform

**Approach:** To fetch the desired output we need to first get the name of the tag and the number of times the tag have been used for that I have used **COUNT** aggregate function, after which we need to join the tags table to photo tag table to create a link between the tables so that we could get the desired hashtags.

Finishing off the limiting the values to 5 as to find the top 5 hashtags by using **LIMIT** function.

## ➤ Query Written

```
-- 4) Hashtag Researching
SELECT
    tags.tag_name, COUNT(tag_name) AS No_of_Time_Tag_Used
FROM
    tags
JOIN photo_tags ON tags.id = photo_tags.tag_id
GROUP BY tags.tag_name
ORDER BY No_of_Time_Tag_Used DESC
LIMIT 5;
```

## ✓ Output Obtained

	tag_name	No_of_Time_Tag_Used
▶	smile	59
	beach	42
	party	39
	fun	38
	concert	24

# Marketing

5) **Launch AD Campaign:** The team wants to know, which day would be the best day to launch ADs..

**Task:** What day of the week do most users register on? Provide insights on when to schedule an ad campaign

**Approach:** In this task, we first need to get the Day from the 'created at' column which was obtained by using DAYNAME function and counting the occurrence of the Day by using COUNT aggregate function. Then afterwards grouping our result by Day and ordering them by their occurrence in descending order.

## ➤ Query Written

```
-- 5) Launch AD Campaign
SELECT
    DAYNAME(created_at) AS Day_of_Week,
    COUNT(created_at) AS Most_registered_Day
FROM
    users
GROUP BY Day_of_Week
ORDER BY Most_registered_Day DESC;
```

## ✓ Output Obtained

	Day_of_Week	Most_registered_Day
▶	Thursday	16
	Sunday	16
	Friday	15
	Tuesday	14
	Monday	14
	Wednesday	13
	Saturday	12



# Investor Metrics

- 1) **User Engagement:** Are users still as active and post on Instagram or they are making fewer posts

**Task:** Provide how many times does average user posts on Instagram.

Also, provide the total number of photos on Instagram/total number of users

**Approach:** To find the average user post we first need to calculate the number of users and the total number of photos, then using the formula

$$\frac{\text{Total number of Photos}}{\text{Total number of Users}}$$

We can find the average number of times a user post on Instagram.

- ❖ PART 1: Calculating the total number of photos and total number of users

I have used following Query, using COUNT aggregate function.

```
SELECT
    COUNT(id) AS `Total Photos`
FROM
    photos ;

SELECT
    COUNT(id) AS `Total Users`
FROM
    users;
```

✓ Output Obtained

Total Photos
257

Total Users
100

- ❖ PART 2 : Calculating the Average number of times a user posts on Instagram. To achieve the task I have used the following Query, here we have used the Sub-Query feature of MySQL Workbench.

```
-- AVERAGE Post Made = Total Photos /Total Users
SELECT
  ((SELECT
    COUNT(id) AS `Total Photos`
  FROM
    photos) / (SELECT
    COUNT(id) AS `Total Users`
  FROM
    users))
  AS `Average Posts`;
```

✓ Output Obtained

	Average Posts
▶	2.5700

# Investor Metrics

- 2) **Bots & Fake Accounts:** The investors want to know if the platform is crowded with fake and dummy accounts.

**Task:** Provide data on users (bots) who have liked every single photo on the site (since any normal user would not be able to do this).

**Approach:** Firstly, I selected the id, username, and the count of likes on photos by using COUNT aggregate function and naming it 'Likes Per User/Person' by using AS clause , afterwards I joined the two tables 'users' and 'likes' to create link between them as to enable the count function, at last I have applied a condition where output will be having the only values where the only users who have liked all the photos will appear using a Sub-query in HAVING clause.

## ➤ Query Written

```
-- 2) Bots And Fake Accouts
SELECT
    u.id, u.username, COUNT(*) AS `Likes Per User/Person`
FROM
    users u
    INNER JOIN
    likes l ON u.id = l.user_id
GROUP BY l.user_id
HAVING `Likes Per User/Person` = (SELECT
    COUNT(*)
FROM
    photos);
```

## ✓ Output Obtained

	id	username	Likes Per User/Person
▶	5	Aniya_Hackett	257
	14	Jadyn81	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

# Result

- ▶ Hence, all the question/tasks given were solved successfully, in these tasks all the basic as well as intermediate concepts related to SQL in Data Analytics were used .