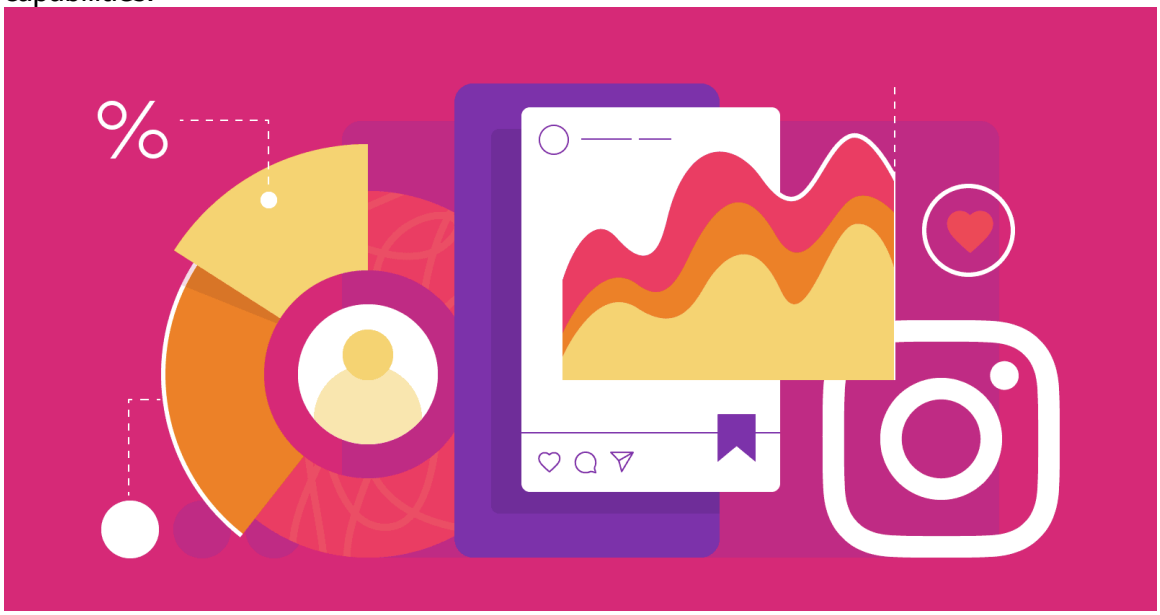

**Project -2**

# INSTAGRAM USER ANALYTICS

Done By:
**B.Havilah**

# I. Project Description

The Instagram User Analytics project aims to analyze user interactions and engagement on the Instagram app through the use of SQL and MySQL Workbench. By extracting valuable insights, this initiative intends to empower diverse business teams, including marketing, product, and development, with data-driven decision-making capabilities.



The derived insights will provide a comprehensive understanding of user behavior, allowing teams to optimize strategies, refine targeting approaches, and enhance features to ultimately improve the Instagram app's user experience and drive overall business growth.

# II. Approach

## A. Creating database

Created the database called **ig_clone**, and executed all the necessary commands to create a full stacked Instagram user database by using the provided database document.

## B. Contents of the database

- Users Table in which we insert the values of username and the timestamp when the user was registered into Instagram.
- Photos Table in which we insert values of photo id, image URL and timestamp.
- Comments Table in which insert values of id, user id, comment text and photo id.
- Likes Table in which we insert the user id, photo id and created at column which is the time stamp.

- Followers Table in which we insert the followers of the users, i.e. follower id and the accounts that user follow, i.e. follow id.
- Tags Table in which we insert id, hashtag name and the timestamp when the tag is created.
- Photo tags Table in which we insert photo id and tag id.

All The SQL commands start by selecting the database **ig_clone** using the **USE** statement, indicating the subsequent queries will focus on this particular database.

## A) Marketing Analysis:

### 1.Loyal User Reward

Following the database selection, the **SELECT** query retrieves specific information from the **users** table. The requested columns are **id**, **username**, and **created_at**. The results are then arranged in ascending order based on the **created_at** timestamp, facilitated by the **ORDER BY created_at ASC** clause.

```
USE ig_clone;
SELECT id, username, created_at
FROM users
ORDER BY created_at ASC
LIMIT 5;
```

To streamline the output, the **LIMIT 5** clause is included, restricting the result set to the first five records.

### 2. Inactive User Engagement

Following the database selection, a **SELECT** query retrieves information from the **user's** table. The specified columns are **id** and **username**. However, the results are filtered using a **WHERE** clause. The condition is structured to exclude users whose IDs are found in the result of a subquery **(SELECT DISTINCT user_id FROM photos)**.

```
USE ig_clone;
SELECT id, username
FROM users
WHERE id NOT IN (SELECT DISTINCT user_id FROM photos);
```

### 3.Contest Winner Declaration

Starting with the "users" table, it utilizes left joins to incorporate information from both the "photos" and "likes" tables, ensuring that all users are included in the analysis, even if they haven't posted photos or received likes. The selected columns include the user ID (**user_id**), the username of the top user (**winner_username**), and the total count of likes received by that user (**total_like_count**).

```
USE ig_clone;
SELECT u.id AS user_id, u.username AS winner_username, COUNT(l.user_id) AS total_like_count
FROM users u
LEFT JOIN photos p ON u.id = p.user_id
LEFT JOIN likes l ON p.id = l.photo_id
GROUP BY u.id, u.username
ORDER BY total_like_count DESC
LIMIT 1;
```

By grouping the results based on user ID and username, the query aggregates the like counts for each user. The final output is ordered in descending order according to the total like count, and the "LIMIT 1" clause ensures that only the user with the highest like count is presented.

### 4. Hashtag Research

Implemented by selecting the tag ID (**t.id**) and tag name (**t.tag_name**) from the "tags" table. The script then joins this information with the "photo_tags" table on the condition that the tag ID matches the one in the "photo_tags" table. The COUNT function is utilized to determine the number of occurrences of each tag in the "photo_tags" table, effectively representing the count of photos associated with each tag.

```
USE ig_clone;
SELECT t.id, t.tag_name, COUNT(pt.photo_id) AS count
FROM tags t
JOIN photo_tags pt ON t.id = pt.tag_id
GROUP BY t.id, t.tag_name
ORDER BY count DESC
LIMIT 5;
```

The results are grouped by both the tag ID and tag name, ensuring that each unique tag is considered individually. The GROUP BY clause facilitates this aggregation. Subsequently, the results are ordered in descending order based on the tag count, and the LIMIT 5 clause is applied to retrieve only the top 5 tags with the highest photo counts.

### 5. Ad Campaign Launch

Implemented by selecting the day of the week ("registration_day") and the corresponding count of users registered on that day from the "users" table. The results are grouped by registration day, and the query then orders the results in descending order based on the user count.

```
USE ig_clone;
SELECT DAYNAME(created_at) AS registration_day, COUNT(*) AS user_count
FROM users
GROUP BY registration_day
ORDER BY user_count DESC
LIMIT 1;
```

Finally, the LIMIT 1 clause ensures that only the top result, representing the day with the highest user registrations, is returned.

## B) Investor Metrics:

### 1.User Engagement

The script counts the average number of posts per user in the "ig_clone" database by initially grouping photos based on user IDs. This is achieved through the "user_photos"

subquery, resulting in a table with user IDs and their corresponding post counts. The outer query then utilizes this intermediate result to calculate the average posts per user using the AVG function, named "avg_posts_per_user

```
USE ig_clone;
SELECT AVG(posts_per_user) AS avg_posts_per_user,
       (SELECT COUNT(*) FROM photos) / (SELECT COUNT(*) FROM users) AS avg_photos_per_user
FROM (SELECT user_id, COUNT(*) AS posts_per_user FROM photos GROUP BY user_id) AS user_photos;
```

." Additionally, it determines the average number of photos per user by dividing the total counts of photos and users obtained through separate subqueries, assigning it the alias "avg_photos_per_user."

### 2.Fake/Bot accounts:

This SQL script in the "ig_clone" database retrieves user information based on specific criteria related to photo activity and likes. It selects user ID (**id**) and username from the "users" table.

The WHERE clause includes two conditions:

1. Users not found in the subquery result selecting user IDs from the "photos" table, indicating users who haven't posted any photos.
2. Users found in a subquery result involving the "likes" table, counting distinct photo IDs liked by each user and comparing it to the total distinct photo IDs in the "photos" table. This identifies users who have liked every distinct photo on the platform.

```
USE ig_clone;
SELECT id, username
FROM users
WHERE id NOT IN (SELECT user_id FROM photos)
  OR id IN (SELECT user_id FROM likes GROUP BY user_id HAVING COUNT(DISTINCT photo_id) = (SELECT COUNT(DISTINCT id) FROM photos));
```

The query effectively identifies users who haven't posted photos or have liked every distinct photo, providing insights into user activity and engagement patterns in the "ig_clone" database.

## III. Tech Stack used

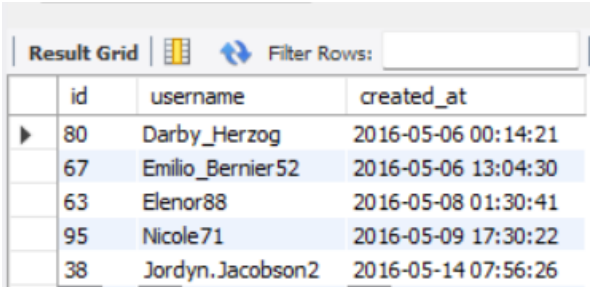
- MYSQL Workbench 8.0 CE
- SQL Server Management Studio

## IV. Insights

These are outputs from the commands executed.

### 1. Loyal user Reward:

Q) Identify the five oldest users on Instagram from the provided database.



Result Grid | Filter Rows:

| 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 |

Ans) The 5 users Darby_Herzog,Emilio_Bernier52,Elenor88,Nicole71,Jordyn.Jacobson2 are the oldest ,i.e. they are the first registered ones in the Instagram app.

## 2.Inactive users Management:

Q) Identify users who have never posted a single photo on Instagram.

(Continuation in next page)

| 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 |

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

| id | username |
|---|---|
| 91 | Bethany20 |
| 101 | Kenton_Kirlin |
| 102 | Andre_Purdy85 |
| 103 | Harley_Lind18 |
| 104 | Arely_Bogan63 |
| 105 | Aniya_Hackett |
| 106 | Travon.Waters |
| 107 | Kasandra_Homenick |
| 108 | Tabitha_Schamber... |
| 109 | Gus93 |
| 110 | Presley_McClure |
| 111 | Justina.Gaylord27 |
| 112 | Dereck65 |
| 113 | Alexandro35 |
| 114 | Jaclyn81 |
| 115 | Billy52 |
| 116 | Annalise.McKenzie16 |
| 117 | Norbert_Carroll35 |
| 118 | Odessa2 |
| 119 | Hailee26 |
| 120 | Delpha.Kihn |
| 121 | Rocio33 |
| 122 | Kenneth64 |
| 123 | Eveline95 |
| 124 | Maxwell.Halvorson |

| id | username |
|---|---|
| 148 | Granville_Kutch |
| 149 | Morgan.Kassulke |
| 150 | Gerard79 |
| 151 | Mariano_Koch3 |
| 152 | Zack_Kemmer93 |
| 153 | Linnea59 |
| 154 | Duane60 |
| 155 | Meggie_Doyle |
| 156 | Peter.Stehr0 |
| 157 | Julien_Schmidt |
| 158 | Aurelie71 |
| 159 | Cesar93 |
| 160 | Sam52 |
| 161 | Jayson65 |
| 162 | Ressie_Stanton46 |
| 163 | Elenor88 |
| 164 | Florence99 |
| 165 | Adelle96 |
| 166 | Mike.Auer39 |
| 167 | Emilio_Bernier52 |
| 168 | Franco_Keebler64 |
| 169 | Karley_Bosco |
| 170 | Erick5 |
| 171 | Nia_Haag |
| 172 | Kathryn80 |

| id | username |
|---|---|
| 220 | Delpha.Kihn |
| 221 | Rocio33 |
| 222 | Kenneth64 |
| 223 | Eveline95 |
| 224 | Maxwell.Halvorson |
| 225 | Tierra.Trantow |
| 226 | Josianne.Friesen |
| 227 | Darwin29 |
| 228 | Dario77 |
| 229 | Jaime53 |
| 230 | Kaley9 |
| 231 | Aiyana_Hoeger |
| 232 | Irwin.Larson |
| 233 | Yvette.Gottlieb91 |
| 234 | Pearl7 |
| 235 | Lennie_Hartmann40 |
| 236 | Ollie_Ledner37 |
| 237 | Yazmin_Mills95 |
| 238 | Jordyn.Jacobson2 |
| 239 | Kelsi26 |
| 240 | Rafael.Hickle2 |
| 241 | Mckenna17 |
| 242 | Maya.Farrell |
| 243 | Janet.Armstrong |
| 244 | Seth46 |

| id | username |
|---|---|
| 245 | David.Osinski47 |
| 246 | Malinda_Streich |
| 247 | Harrison.Beatty50 |
| 248 | Granville_Kutch |
| 249 | Morgan.Kassulke |
| 250 | Gerard79 |
| 251 | Mariano_Koch3 |
| 252 | Zack_Kemmer93 |
| 253 | Linnea59 |
| 254 | Duane60 |
| 255 | Meggie_Doyle |
| 256 | Peter.Stehr0 |
| 257 | Julien_Schmidt |
| 258 | Aurelie71 |
| 259 | Cesar93 |
| 260 | Sam52 |
| 261 | Jayson65 |
| 262 | Ressie_Stanton46 |
| 263 | Elenor88 |
| 264 | [illegible]ence99 |
| 265 | [illegible]elle96 |
| 266 | Mike.Auer39 |
| 267 | Emilio_Bernier52 |
| 268 | Franco_Keebler64 |
| 269 | Karley_Bosco |

| id | username |
|---|---|
| 343 | Janet.Armstrong |
| 344 | Seth46 |
| 345 | David.Osinski47 |
| 346 | Malinda_Streich |
| 347 | Harrison.Beatty50 |
| 348 | Granville_Kutch |
| 349 | Morgan.Kassulke |
| 350 | Gerard79 |
| 351 | Mariano_Koch3 |
| 352 | Zack_Kemmer93 |
| 353 | Linnea59 |
| 354 | Duane60 |
| 355 | Meggie_Doyle |
| 356 | Peter.Stehr0 |
| 357 | Julien_Schmidt |
| 358 | Aurelie71 |
| 359 | Cesar93 |
| 360 | Sam52 |
| 361 | Jayson65 |
| 362 | Ressie_Stanton46 |
| 363 | Elenor88 |
| 364 | Florence99 |
| 365 | Adelle96 |
| 366 | Mike.Auer39 |
| 367 | Emilio_Bernier52 |

| id | username |
|---|---|
| 368 | Franco_Keebler64 |
| 369 | Karley_Bosco |
| 370 | Erick5 |
| 371 | Nia_Haag |
| 372 | Kathryn80 |
| 373 | Jaylan.Lakin |
| 374 | Hulda.Macejkovic |
| 375 | Leslie67 |
| 376 | Janelle.Nikolaus81 |
| 377 | Donald.Fritsch |
| 378 | Colten.Harris76 |
| 379 | Katarina.Dibbert |
| 380 | Darby_Herzog |
| 381 | Esther.Zulauf61 |
| 382 | Aracely.Johnston98 |
| 383 | Bartholome.Bernhard |
| 384 | Alysa22 |
| 385 | Milford_Gleichner42 |
| 386 | Delfina_VonRuede... |
| 387 | Rick29 |
| 388 | Clint27 |
| 389 | Jessyca_West |
| 390 | Esmeralda.Mraz57 |
| 391 | Bethany20 |
| 392 | Frederik_Rice |

| id | username |
|---|---|
| 386 | Delfina_VonRuede... |
| 387 | Rick29 |
| 388 | Clint27 |
| 389 | Jessyca_West |
| 390 | Esmeralda.Mraz57 |
| 391 | Bethany20 |
| 392 | Frederik_Rice |
| 393 | Willie_Leuschke |
| 394 | Damon35 |
| 395 | Nicole71 |
| 396 | Keenan.Schamberg... |
| 397 | Tomas.Beatty93 |
| 398 | Imani_Nicolas17 |
| 399 | Alek_Watsica |
| 400 | Javonte83 |
| 401 | Kenton_Kirlin |
| 402 | Andre_Purdy85 |
| 403 | Harley_Lind18 |
| 404 | Arely_Bogan63 |
| 405 | Aniya_Hackett |
| 406 | Travon.Waters |
| 407 | Kasandra_Homenick |
| 408 | Tabitha_Schamber... |
| 409 | Gus93 |
| 410 | Presley_McClure |

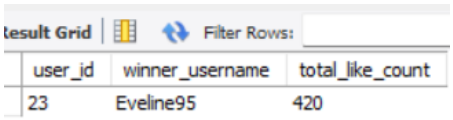
| id | username |
|---|---|
| 448 | Granville_Kutch |
| 449 | Morgan.Kassulke |
| 450 | Gerard79 |
| 451 | Mariano_Koch3 |
| 452 | Zack_Kemmer93 |
| 453 | Linnea59 |
| 454 | Duane60 |
| 455 | Meggie_Doyle |
| 456 | Peter.Stehr0 |
| 457 | Julien_Schmidt |
| 458 | Aurelie71 |
| 459 | Cesar93 |
| 460 | Sam52 |
| 461 | Jayson65 |
| 462 | Ressie_Stanton46 |
| 463 | Elenor88 |
| 464 | Florence99 |
| 465 | Adelle96 |
| 466 | Mike.Auer39 |
| 467 | Emilio_Bernier52 |
| 468 | Franco_Keebler64 |
| 469 | Karley_Bosco |
| 470 | Erick5 |
| 471 | Nia_Haag |
| 472 | Kathryn80 |

| id | username |
|---|---|
| 795 | Nicole71 |
| 796 | Keenan.Schamberg... |
| 797 | Tomas.Beatty93 |
| 798 | Imani_Nicolas17 |
| 799 | Alek_Watsica |
| 800 | Javonte83 |
| NULL | NULL |

Ans) There are many users who are inactive, so I actually haven't included all the snapshots of the output, but there were over 800 users.

## 3. Contest Winner Declaration:

Q) Determine the winner of the contest and provide their details to the team.

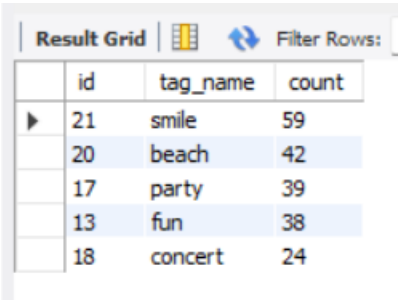


| user_id | winner_username | total_like_count |
|---|---|---|
| 23 | Eveline95 | 420 |

Ans)Eveline95 is the user with most liked photo with count of 420.

## 4.Hashtag Research:

Q) Identify and suggest the top five most commonly used hashtags on the platform.

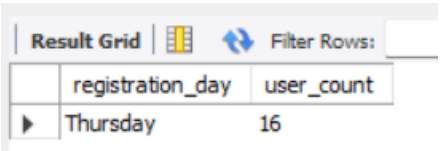


| id | tag_name | count |
|---|---|---|
| 21 | smile | 59 |
| 20 | beach | 42 |
| 17 | party | 39 |
| 13 | fun | 38 |
| 18 | concert | 24 |

Ans)The top most used hashtags are smile,beach ,party, fun and Concert.

## 5.Ad Campaign Launch:

Q) Determine the day of the week when most users register on Instagram. Provide insights on when to schedule an ad campaign.

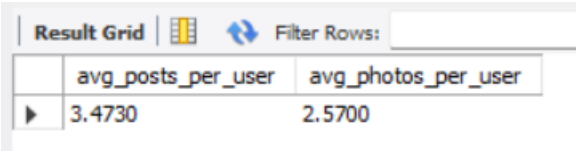


| registration_day | user_count |
|---|---|
| Thursday | 16 |

Ans) The campaign could be scheduled on Thursday, because most of the registrations occurred on Thursday with user count of 16.

## 6.User Engagement:

Q) 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.

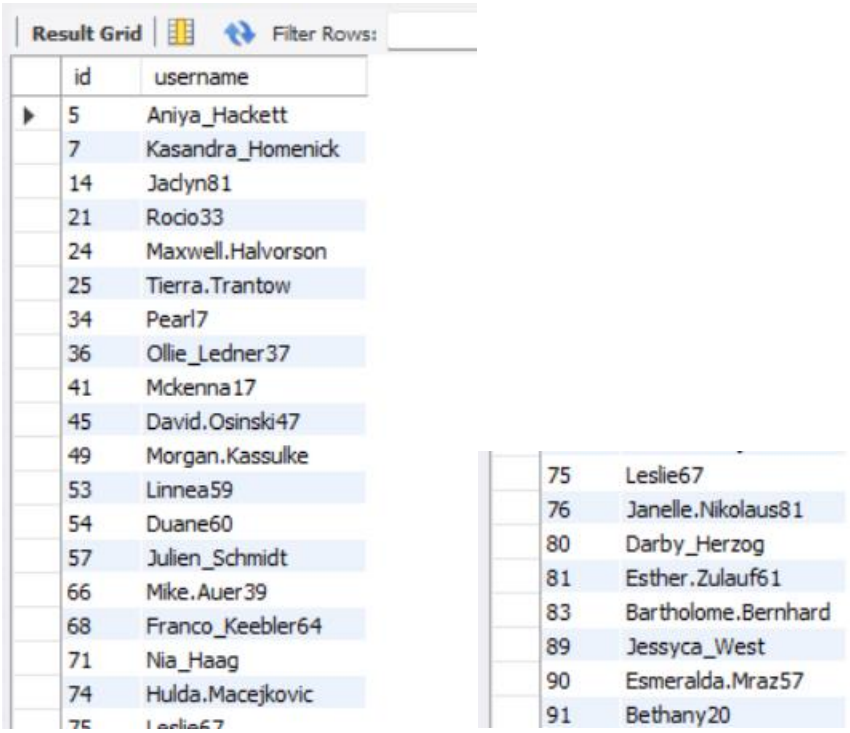


| avg_posts_per_user | avg_photos_per_user |
|---|---|
| 3.4730 | 2.5700 |

Ans) The average posts per user is 3.4730
The average photos per user is 2.5700

### 7.Fake/Bot Accounts:

Q) Identify users (potential bots) who have liked every single photo on the site, as this is not typically possible for a normal user.



Result Grid | Filter Rows:

| 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 |

Ans) There are total 26 Fake/Bot accounts in given database.

**Observation:**
Here, the output of Inactive users includes the output of Fake/Bot accounts because: Bot users do not upload single photo from their account but likes every other post on the Instagram. As per the given question, we have to look at the photos are uploaded from the account or not, so This is the reason the fake or bot accounts are also counted as Inactive users.

## V. Result

- gaining insights into the patterns of user behavior.
- These insights provided me with a comprehensive understanding of marketing strategies, including the optimization of Ad Campaigns and initiatives to encourage user activity on the platform. These efforts contribute to diversity in content and enhancing overall user engagement.
- Rewarding loyal users and identifying the most used hashtags of the app contribute to businesses' efforts in enhancing their brand visibility, fostering customer loyalty, and expanding their reach to a broader audience.
- Monitoring the average posts and photos per user helps to understand interaction with the app and the overall health of the platform from an investor perspective.
- Identifying the fake and bot accounts is crucial for maintaining the integrity of user engagement metrics and investor confidence.

## VI. Conclusion

This project helped me to get an overview of MySQL workbench and how it works. Also, I got a grip on SQL commands, I really find the project work interesting and the insights derived help me to understand how the analytics work and what businesses need from this data analytics in order to increase the growth.

## VII. Drive Link

**instagram_user_analytics**