Instagram User Analytics

- Rukmini Annadata
- rukmini2k2@gmail.com

SQL Tasks :

A) Marketing Analysis:

1.Loyal User Reward:

```
74  select username, created_at
75  from users
76  order by created_at ASC
77  limit 5;
78
```

Result Grid | Filter Rows: | Export: | Wrap Cell Content: | Fetch rows:

username	created_at
Darby_Herzog	2016-05-06 00:14:21
Emilio_Bernier52	2016-05-06 13:04:30
Elenor88	2016-05-08 01:30:41
Nicole71	2016-05-09 17:30:22
Jordyn.Jacobson2	2016-05-14 07:56:26

To identify the most loyal, i.e., the top 5 oldest users of Instagram, follow these steps:

1. Retrieve data from the "users" table by selecting the "username" and "created_at" columns.

2. Sort the data in ascending order based on the "created_at" column using the "ORDER BY" function.

3. Use the "LIMIT" function to display only the top 5 oldest Instagram users in the output.

To identify the top 5 most loyal users of Instagram, we will retrieve the usernames and creation dates from the users table. The data will be sorted in ascending order based on the creation dates using the ORDER BY function. This way, we can find the users who have been using the platform for the longest time.

2.Inactive User Engagement

To identify the most inactive users on Instagram, those who have never posted a single photo, we will follow these steps:

1. Select the "username" column from the "users" table.

2. Perform a LEFT JOIN between the "users" table and the "photos" table, matching "users.id" with "photos.user_id" since they share common data.

3. Filter the rows from the "users" table where there is no matching "photos.id" (i.e., where the "photos" table does not have any records for that user). This will give us the users who have never posted a photo on Instagram.

```
79  -- Program/Query:
80  select username, users.id as user_id
81  from users
82  left join photos
83  on users.id = photos.user_id
84  where photos.id IS NULL
85  order by users.id;
```

Result Grid | Filter Rows: | Export: | Wra

username	user_id
Aniya_Hackett	5
Kasandra_Homenick	7
Jaclyn81	14
Rocio33	21
Maxwell.Halvorson	24
Tierra.Trantow	25
Pearl7	34
Ollie_Ledner37	36
Mckenna17	41
David.Osinski47	45
Morgan.Kassulke	49
Linnea59	53
Duane60	54
Julien_Schmidt	57

```
85  order by users.id;
```

Result Grid | Filter Rows:

username	user_id
Duane60	54
Julien_Schmidt	57
Mike.Auer39	66
Franco_Keebler64	68
Nia_Haag	71
Hulda.Macejkovic	74
Leslie67	75
Janelle.Nikolaus81	76
Darby_Herzog	80
Esther.Zulauf61	81
Bartholome.Bernhard	83
Jessyca_West	89
Esmeralda.Mraz57	90
Bethany20	91

Result 2

3.Contest Winner Declaration:

To find the username, photo_id, image_url, and total_number_of_likes of the most-liked photo on Instagram, we will perform the following steps:

1. Select the "users.username," "photos.id," "photos.image_url," and "COUNT(*) as total" from the respective tables.

2. Perform an inner join among the "photos," "likes," and "users" tables using the conditions "likes.photo_id = photos.id" and "photos.user_id = users.id."

3. Group the output based on "photos.id" using the GROUP BY function.

4. Sort the data in descending order based on the "total" using the ORDER BY function.

5. Retrieve only the information of the top-liked photo using the LIMIT function.

```
86
87
88 •  select users.id as user_id, users.username, photos.id as photo_id,
89    photos.image_url, count(*) as total
90    from photos
91    inner join likes
92    on likes.photo_id = photos.id
93    inner join users
94    on photos.user_id = users.id
95    group by photos.id
96    order by total DESC
97    limit 1;
```

Result Grid | Filter Rows: | Export: | Wrap Cell Content: | Fetch rows:

user_id	username	photo_id	image_url	total
52	Zack_Kemmer93	145	https://jarret.name	48

4.Hashtag Research:

To determine the top 5 most commonly used hashtags on Instagram, follow these steps:

1. Select the "tag_name" column from the "tags" table and use the "COUNT(*) as total" function to count the number of times each hashtag is used individually.

2. Perform an inner join between the "tags" table and the "photo_tags" table on the condition "tags.id = photo_tags.tag_id" as they both contain related information, i.e., "tag_id."

3. Group the resulting output based on "tags.tag_name" using the GROUP BY function.

4. Sort the grouped data in descending order based on the "total" (the total number of occurrences for each tag_name) using the ORDER BY function.

5. Retrieve only the top 5 most used tag names by applying the LIMIT 5 function.

```
99
100 •  select tags.tag_name, count(*) as total_number_of_times_tag_used_individually
101    from tags
102    join photo_tags
103    on tags.id = photo_tags.tag_id
104    group by tags.tag_name
105    order by total_number_of_times_tag_used_individually DESC
106    limit 5;
```

up

Result Grid | Filter Rows: | Export: | Wrap Cell Content: | Fetch rows:

tag_name	total_number_of_times_tag_used_individually
smile	59
beach	42
party	39
fun	38
concert	24

5.Ad Campaign Launch:

To determine the day of the week on which most users register on Instagram, follow these steps:

1. Select the "dayname(created_at) as day_of_week" and "count(*) as total_number_of_users_registered" columns from the "users" table to create the desired output table.

2. Group the output table based on the "day_of_week" using the GROUP BY function, which will group the user registration data according to the day of the week.

3. Sort the output table in descending order based on the "total_number_of_users_registered" using the ORDER BY function. This will help us identify the day with the highest number of user registrations.

```
107
108 •  select dayname(created_at) as day_of_week,
109    count(*) as total_number_of_users_registered
110    from users
111    group by day_of_week
112    order by total_number_of_users_registered DESC;
113
```

Result Grid | Filter Rows: | Export

day_of_week	total_number_of_users_registered
Thursday	16
Sunday	16
Friday	15
Tuesday	14
Monday	14
Wednesday	13
Saturday	12

B) Investor Metrics:

1.User Engagement:

To calculate the average number of posts on Instagram per user, follow these steps:

1. Begin by counting the total number of photos (posts) in the "photos" table using the "count(*)" function.

2. Similarly, count the total number of users in the "users" table using the "count(*)" function.

3. Divide the total number of photos by the total number of users (count from photos / count from users). This will give us the average number of posts per user.

4. To find how frequently users post on Instagram, you need to determine the total occurrences of each unique "user_id" in the "photos" table. This will help you understand the posting behavior of individual users.

```
114
115 •   select
116    (select count(*) from photos)/(select count(*) from users) as
117    total_photos_divide_total_photos;
```

Result Grid | Filter Rows: | Export: | Wrap Cell Content:

total_photos_divide_total_photos
2.5700

```
119    /*Program/Query to find the times each user posts on Instagram :*/
120 •  select user_id,count(*) as user_post_count
121    from photos
122    group by user_id
123    order by user_id;
```

Result Grid | Filter Rows:

user_id	user_post_count
1	5
2	4
3	4
4	3
6	5
8	4
9	4
10	3
11	5
12	4
13	5
15	4

Result 7 ×

Result Grid | Filter Rows:

user_id	user_post_count
19	2
20	1
22	1
23	12
26	5
27	1
28	4
29	8
30	2
31	1
32	4
33	5

Result Grid | Filter Rows:

user_id	user_post_count
43	5
44	4
46	4
47	5
48	1
50	3
51	5
52	5
55	1
56	1
58	8
59	10

Result 7 ×

Result Grid | Filter Rows:

user_id	user_post_count
86	9
87	4
88	11
92	3
93	2
94	1
95	2
96	3
97	2
98	1
99	3
100	2

Result 7 ×

2.Bots & Fake Accounts:

To identify potential bots and fake accounts on Instagram, follow these steps:

1. Select the "user_id" column from the "photos" table to track users who have received likes on their photos.

2. Additionally, select the "username" column from the "users" table to display the usernames of these users.

3. Use the "count(*)" function to calculate the total number of likes each user has received from the "likes" table.

4. Perform an inner join between the "users" and "likes" tables based on the common field "users.id" and "likes.user_id" using the "on" clause.

5. Group the output table using the "group by" function, which will group the results based on "likes.user_id" to consolidate the likes for each user.

6. Finally, search for users whose total likes (count from photos) match the total likes received (total_likes_per_user). These users might be potential bots or accounts engaging in suspicious behavior.

```
125
126 •  select user_id, username, count(*) as total_likes_per_user
127    from users
128    inner join likes
129    on users.id = likes.user_id
130    group by likes.user_id
131    having total_likes_per_user = (select count(*) from photos);
```

Result Grid | Filter Rows: | Export: | Wrap Cell Content:

user_id	username	total_likes_per_user
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 8

```
43        FOREIGN KEY (follower_id) REFERENCES users(id),
44        FOREIGN KEY (followee_id) REFERENCES users(id),
45        PRIMARY KEY(follower_id,followee_id)
46    );
47
48    /*Tags*/
49 •  CREATE TABLE tags(
50        id INTEGER AUTO_INCREMENT PRIMARY KEY,
51        tag_name VARCHAR(255) UNIQUE NOT NULL,
52        created_at TIMESTAMP DEFAULT NOW()
53    );
54
55    /*junction table: Photos - Tags*/
56 •  CREATE TABLE photo_tags(
57        photo_id INT NOT NULL,
58        tag_id INT NOT NULL,
59        FOREIGN KEY(photo_id) REFERENCES photos(id),
60        FOREIGN KEY(tag_id) REFERENCES tags(id),
61        PRIMARY KEY(photo_id,tag_id)
62    );
63 •  INSERT INTO users (username, created_at) VALUES ('Kenton_Kirlin', '2017-02-16 18:22:10.846'), ('Andre_Purdy85', '2017-04-02 17:11:21.4
64 •  INSERT INTO photos(image_url, user_id) VALUES ('http://elijah.biz', 1), ('https://shanon.org', 1), ('http://vicky.biz', 1), ('http://o
65 •  INSERT INTO follows(follower_id, followee_id) VALUES (2, 1), (2, 3), (2, 4), (2, 5), (2, 6), (2, 7), (2, 8), (2, 9), (2, 10), (2, 11),
66 •  INSERT INTO comments(comment_text, user_id, photo_id) VALUES ('unde at dolorem', 2, 1), ('quae ea ducimus', 3, 1), ('alias a voluptatu
67 •  INSERT INTO likes(user_id,photo_id) VALUES (2, 1), (5, 1), (9, 1), (10, 1), (11, 1), (14, 1), (19, 1), (21, 1), (24, 1), (35, 1), (36,
68
69 •  INSERT INTO tags(tag_name) VALUES ('sunset'), ('photography'), ('sunrise'), ('landscape'), ('food'), ('foodie'), ('delicious'), ('beau
70 •  INSERT INTO photo_tags(photo_id, tag_id) VALUES (1, 18), (1, 17), (1, 21), (1, 13), (1, 19), (2, 4), (2, 3), (2, 20), (2, 2), (3, 8),
71
```