INSTAGRAM USER ANALYTICS

Ankit Gaikar

INTRODUCTION

In this project we have analyzed Instagram user activities based on two basic requirements which are

- 1. Marketing
 - Most loyal users
 - Inactive users
 - Contest winners
 - Hashtag report
 - Weekly uses report
- 2. Invester metrics
 - User engagement
 - Fake/bot accounts

Approach

As per standard procedure we discussed/ the requirements in detail with marketing teams and investors. And worked on them with SQL.

Software used

• MySQL Workbench 8.0

Most loyal users:

```
• SQL Query:
select
from users
order by created_at
limit 5
```

• Result:

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

Inactive users:

There are total of 26 inactive users who haven't posted anything on Instagram.

```
SQL Query:

SELECT

users.id as 'user id',

users.username

FROM

users

WHERE

users.id NOT IN

(SELECT

photos.user_id

FROM

photos)
```

user_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

Contest Winner:

'Zack_Kemmer93'

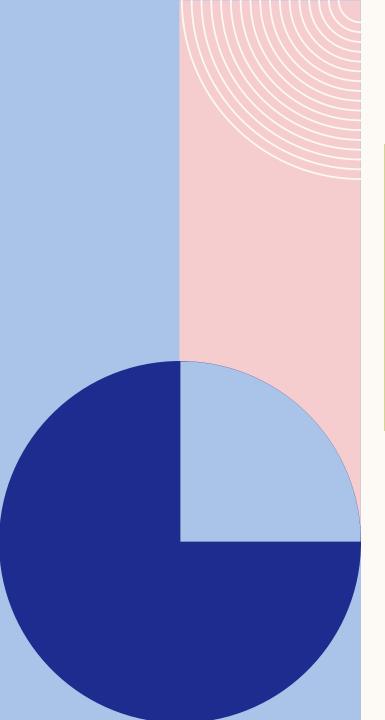
```
SQL Query:

SELECT

photos.user_id,
users.username,
likes.photo_id,
COUNT(likes.photo_id) AS likes_count
FROM
likes
JOIN
photos ON likes.photo_id = photos.id
JOIN
users ON photos.user_id = users.id
GROUP BY photo_id
ORDER BY likes_count DESC
LIMIT 1
```

• Result:

user_id	username	photo_id	likes_count
52	Zack_Kem mer93	145	48



Hashtag report:

```
SQL Query:

SELECT

tags.id AS tag_id,

tags.tag_name,

COUNT(photo_tags.tag_id) AS uses_count

FROM

tags

JOIN

photo_tags ON photo_tags.tag_id = tags.id

GROUP BY photo_tags.tag_id

ORDER BY uses_count DESC

LIMIT 5
```

tag_id	tag_name	uses_count
21	smile	59
20	beach	42
17	party	39
13	fun	38
18	concert	24

Weekly uses report:

SQL Query:

SELECT
DAYNAME(created_at) AS
day_name,
COUNT(created_at) AS
registration_count
FROM
users
GROUP BY day_name

day_name	registration_count
Thursday	16
Sunday	16
Friday	15
Tuesday	14
Monday	14
Wednesday	13
Saturday	12

User Engagement:

```
SQL Query:

SELECT

AVG(post_count) AS 'avrage post count',

SUM(post_count) AS 'total photos',

COUNT(ID) AS 'total users'

FROM

(SELECT

users.id AS ID, COUNT(photos.id) AS post_count

FROM

photos

RIGHT JOIN users ON photos.user_id = users.id

GROUP BY users.id) post_per_user
```

avrage post count	total photos	total users
2.5700	257	100

Fake/ Bot Accounts:

```
SQL Query:
         SELECT
           a.user_id, users.username
         FROM
           (SELECT
             user_id, COUNT(photo_id) AS like_count
           FROM
             likes
           GROUP BY user_id) a
             JOIN
           users ON a.user_id = users.id
         WHERE
           like_count = (SELECT
               COUNT(id)
             FROM
               photos)
```

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

Presentation title 11

SUMMARY

Through this project we learned uses of various Aggregate and Sorting functions, various joins, date functions and windows functions.

THANK YOU