

## A. Marketing Analysis:

### 1) Loyal User Reward:

**Query :** select \* from users order by created\_at limit 5;

The screenshot shows a database query editor interface. At the top, there's a toolbar with various icons and a 'Limit to 1000 rows' dropdown. Below the toolbar, the query editor contains the following SQL query:

```
1  
2  
3 • select * from users order by created_at limit 5;  
4
```

Below the query editor, there's a 'Result Grid' section. It includes a 'Filter Rows' input field, an 'Edit' button, an 'Export/Import' button, a 'Wrap Cell Content' button, and a 'Fetch rows' button. The result grid displays the following data:

	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

At the bottom, there's a 'users 9' tab and 'Apply' and 'Revert' buttons.

### 2) Interactive User Engagement

**Query :** select \* from users where id not in (select user\_id from photos);

The screenshot shows a database query editor interface. At the top, there's a toolbar with various icons and a 'Limit to 1000 rows' dropdown. Below the toolbar, the query editor contains the following SQL query:

```
1  
2 • select * from users where id not in (select user_id from photos);  
3
```

Below the query editor, there's a 'Result Grid' section. It includes a 'Filter Rows' input field, an 'Edit' button, an 'Export/Import' button, a 'Wrap Cell Content' button, and a 'Fetch rows' button. The result grid displays the following data:

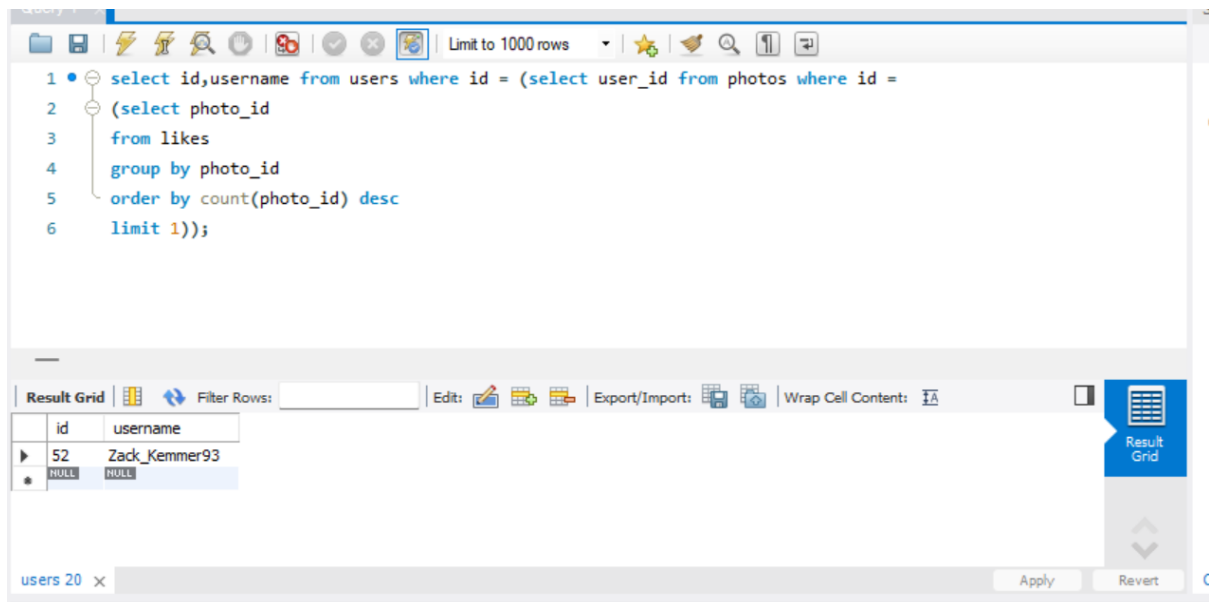
	id	username	created_at
	24	Maxwell.Halvorson	2017-04-18 02:32:44
	90	Esmeralda.Mraz57	2017-03-03 11:52:27
	53	Linnea59	2017-02-07 07:49:34
	14	Jadyn81	2017-02-06 23:29:16
	45	David.Osinski47	2017-02-05 21:23:37
	57	Julien_Schmidt	2017-02-02 23:12:48
▶	34	Hulda.Macejkovic	2017-01-25 17:17:28
	21	Rocio33	2017-01-23 11:51:15
	81	Esther.Zulauf61	2017-01-14 17:02:34
	54	Duane60	2016-12-21 04:43:38
	7	Kassandra_Homenick	2016-12-12 06:50:08
	5	Aniya_Hackett	2016-12-07 01:04:39
	68	Franco_Keebler64	2016-11-13 20:09:27
	83	Bartholome.Bernhard	2016-11-06 02:31:23
	49	Morgan.Kassulke	2016-10-30 12:42:31
	25	Tierra.Trantow	2016-10-03 12:49:21
	75	Leila67	2016-09-21 05:14:01

At the bottom, there's a 'users 14' tab and 'Apply' and 'Revert' buttons.

<b>id</b>	<b>username</b>	<b>created_at</b>
5	Aniya_Hackett	2016-12-07 01:04:39
7	Kasandra_Homenick	2016-12-12 06:50:08
14	Jaclyn81	2017-02-06 23:29:16
21	Rocio33	2017-01-23 11:51:15
24	Maxwell.Halvorson	2017-04-18 02:32:44
25	Tierra.Trantow	2016-10-03 12:49:21
34	Pearl7	2016-07-08 21:42:01
36	Ollie_Ledner37	2016-08-04 15:42:20
41	Mckenna17	2016-07-17 17:25:45
45	David.Osinski47	2017-02-05 21:23:37
49	Morgan.Kassulke	2016-10-30 12:42:31
53	Linnea59	2017-02-07 07:49:34
54	Duane60	2016-12-21 04:43:38
57	Julien_Schmidt	2017-02-02 23:12:48
66	Mike.Auer39	2016-07-01 17:36:15
68	Franco_Keebler64	2016-11-13 20:09:27
71	Nia_Haag	2016-05-14 15:38:50
74	Hulda.Macejkovic	2017-01-25 17:17:28
75	Leslie67	2016-09-21 05:14:01
76	Janelle.Nikolaus81	2016-07-21 09:26:09
80	Darby_Herzog	2016-05-06 00:14:21
81	Esther.Zulauf61	2017-01-14 17:02:34
83	Bartholome.Bernhard	2016-11-06 02:31:23
89	Jessyca_West	2016-09-14 23:47:05
90	Esmeralda.Mraz57	2017-03-03 11:52:27
91	Bethany20	2016-06-03 23:31:53

### 3)Contest winner declaration

**Query** : select id, username from users where id = (select user\_id from photos where id =  
where id=(select photo\_id  
from likes  
group by photo\_id  
order by count(photo\_id) desc  
limit 1));



The screenshot shows a database query editor interface. The query is as follows:

```
1 • select id,username from users where id = (select user_id from photos where id =  
2 (select photo_id  
3 from likes  
4 group by photo_id  
5 order by count(photo_id) desc  
6 limit 1));
```

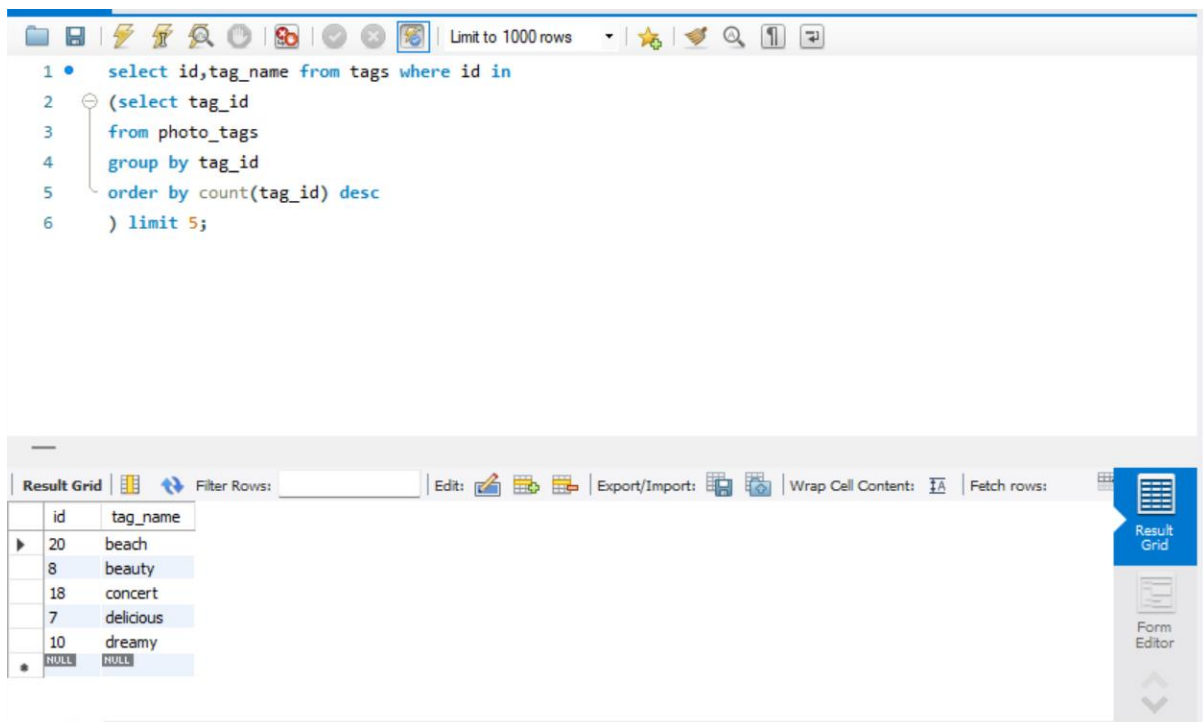
Below the query editor, the 'Result Grid' is displayed, showing the results of the query. The grid has two columns: 'id' and 'username'. The first row shows the result for the user with id 52, whose username is 'Zack\_Kemmer93'. The second row shows 'NULL' for both columns, indicating no further results.

	id	username
▶	52	Zack_Kemmer93
*	NULL	NULL

The interface also includes a toolbar with various icons for editing and viewing the query, and a status bar at the bottom showing 'users 20' and 'Apply Revert' buttons.

#### 4)Hashtag research

**Query** : select id,tag\_name from tags where id in  
(select tag\_id  
from photo\_tags  
group by tag\_id  
order by count(tag\_id) desc  
) limit 5;



The screenshot shows a database query editor interface. The top toolbar includes icons for file operations, a 'Limit to 1000 rows' dropdown, and search/execution tools. The SQL editor contains the following query:

```
1 • select id,tag_name from tags where id in
2   (select tag_id
3     from photo_tags
4     group by tag_id
5     order by count(tag_id) desc
6   ) limit 5;
```

Below the editor is the 'Result Grid' section. It has a toolbar with 'Filter Rows', 'Edit', 'Export/Import', 'Wrap Cell Content', and 'Fetch rows'. The results are displayed in a table with two columns: 'id' and 'tag\_name'.

	id	tag_name
▶	20	beach
	8	beauty
	18	concert
	7	delicious
	10	dreamy
*	NULL	NULL

On the right side of the 'Result Grid' section, there are buttons for 'Result Grid' and 'Form Editor', along with up and down arrow icons.

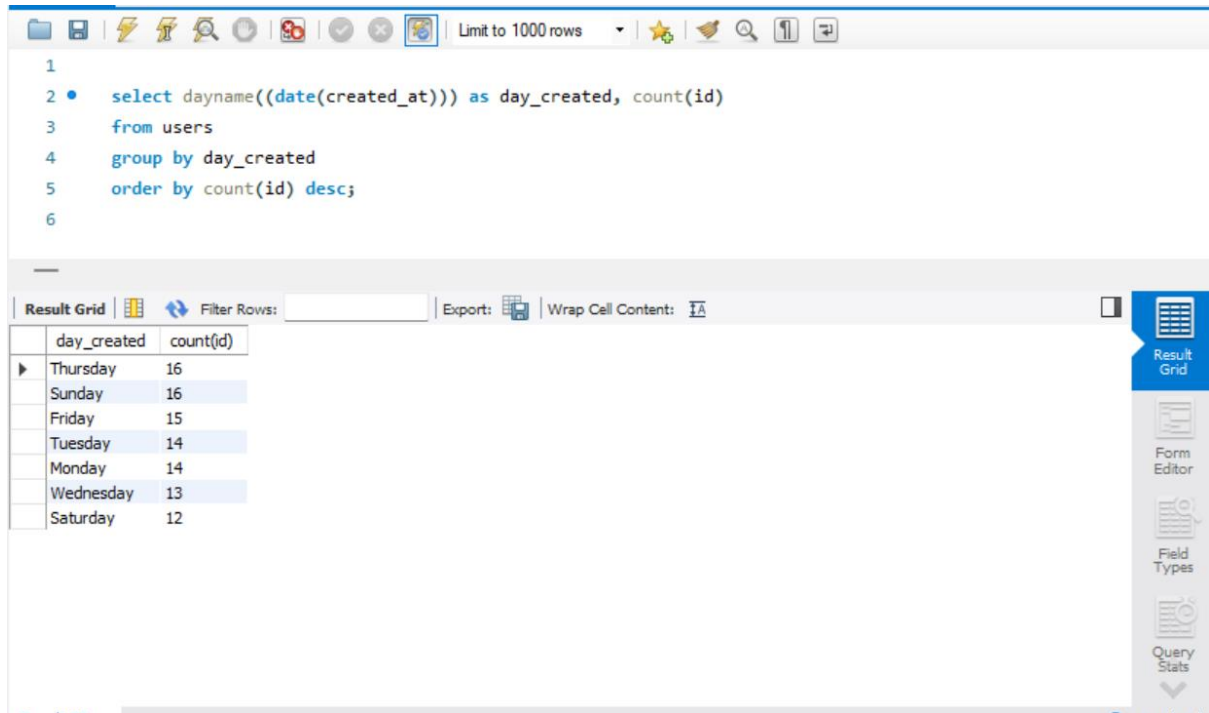
### 5)Ad campaign launch

**Query** : select dayname((date(created\_at))) as day\_created, count(id)

from users

group by day\_created

order by count(id) desc;



The screenshot shows a database query interface. The top toolbar includes icons for file operations, a search icon, and a 'Limit to 1000 rows' dropdown. The SQL query is displayed in a text area:

```
1
2 • select dayname((date(created_at))) as day_created, count(id)
3   from users
4  group by day_created
5  order by count(id) desc;
6
```

Below the query, the 'Result Grid' tab is active, showing the query results in a table. The table has two columns: 'day\_created' and 'count(id)'. The results are sorted by 'count(id)' in descending order.

day_created	count(id)
Thursday	16
Sunday	16
Friday	15
Tuesday	14
Monday	14
Wednesday	13
Saturday	12

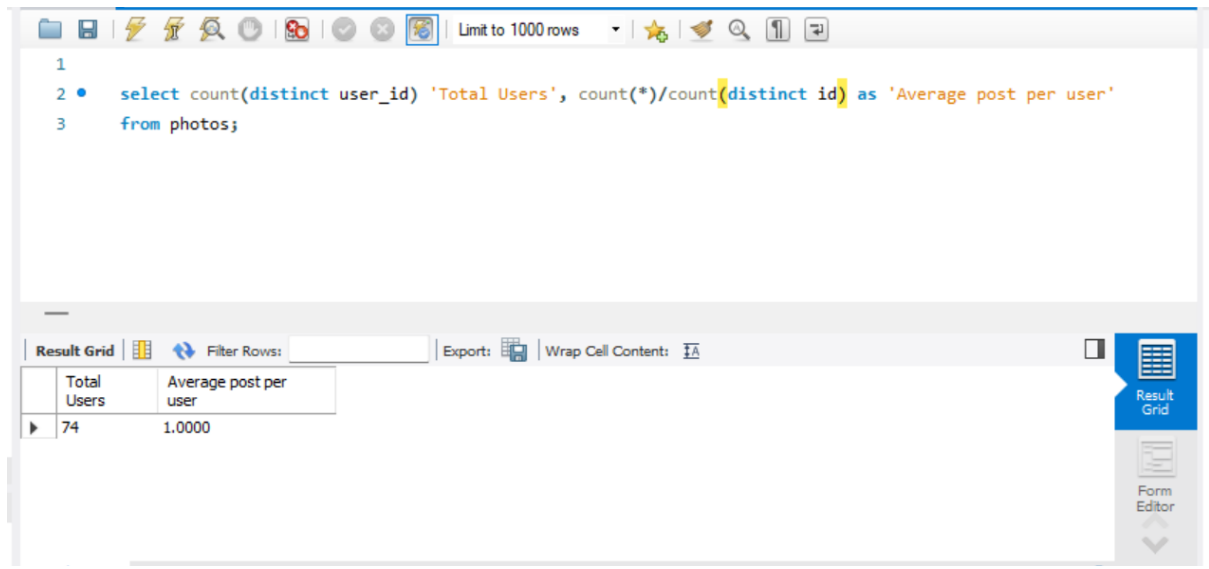
On the right side of the interface, there is a vertical toolbar with icons for 'Result Grid', 'Form Editor', 'Field Types', and 'Query Stats'.

## **B.Investor Metrics:**

### **1)User Engagement:**

**Query** : select count(distinct user\_id) 'Total Users', count(\*)/count(distinct id)  
as 'Average post per user'

from photos;



The screenshot shows a database query editor interface. The top toolbar includes icons for file operations, execution, and search, along with a 'Limit to 1000 rows' dropdown. The query editor contains the following SQL code:

```
1  
2 • select count(distinct user_id) 'Total Users', count(*)/count(distinct id) as 'Average post per user'  
3 from photos;
```

Below the query editor, the 'Result Grid' is displayed. It shows a table with two columns: 'Total Users' and 'Average post per user'. The first row of data shows '74' for 'Total Users' and '1.0000' for 'Average post per user'. The interface also includes an 'Export' button, a 'Wrap Cell Content' checkbox, and a 'Form Editor' button on the right side.

	Total Users	Average post per user
▶	74	1.0000

## 2)Bots & Fake accounts

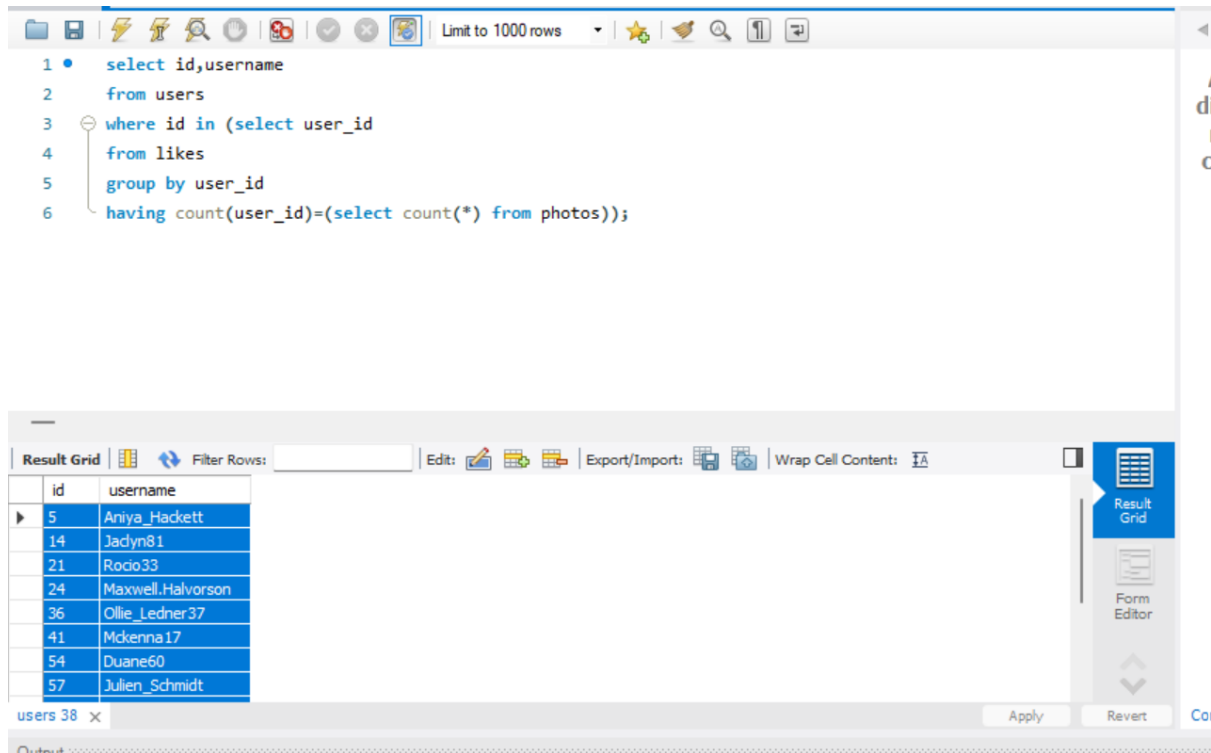
**Query** : select id,username from users

where id in (select user\_id

from likes

group by user\_id

having count(user\_id)=(select count(\*) from photos));



The screenshot shows a SQL query editor with the following query:

```
1 • select id,username
2   from users
3   where id in (select user_id
4                 from likes
5                 group by user_id
6                 having count(user_id)=(select count(*) from photos));
```

Below the query editor, the results are displayed in a table:

id	username
5	Aniya_Hackett
14	Jaclyn81
21	Rocio33
24	Maxwell.Halvorson
36	Ollie_Ledner37
41	Mckenna17
54	Duane60
57	Julien_Schmidt

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

### **Project description :**

The purpose of this project is to provide the insights about the Instagram users, tags, photos, to identify fake accounts and so on.

### **Approach:**

Steps I followed –

- I marked the integrity constraints such as primary key, foreign key
- I drew the relation between the entities given
- Break down the given problem statement into sub modules where multiple tables are involved (subquery)

### **Tech Stack used :** MySQL Workbench

I have used MySQL workbench. Reason why I have selected the mentioned tech stack is it is cross platform compatible open source RDBMS.

### **Insights :**

I was able to analyze the data to get the required results like which day would be the best to launch the campaign, how to find bots/fake accounts and so on.

### **Result :**

I have applied the knowledge to get the required findings from the given data. It was challenging at some point later I referred the documentation to resolve it.

### **Drive Link :**

This project is very helpful, it has given real time example to gain the knowledge in much more understanding way.