

Instagram User Analytics

Project Description:

This project is about how users are engaging the Instagram platform and how the data is collected and analysed by queries.

In this task we are going to find marketing metrics and investor metrics.

Approach:

- 1) Create data from sql queries.
- 2) Used mysql workbench to execute the queries.

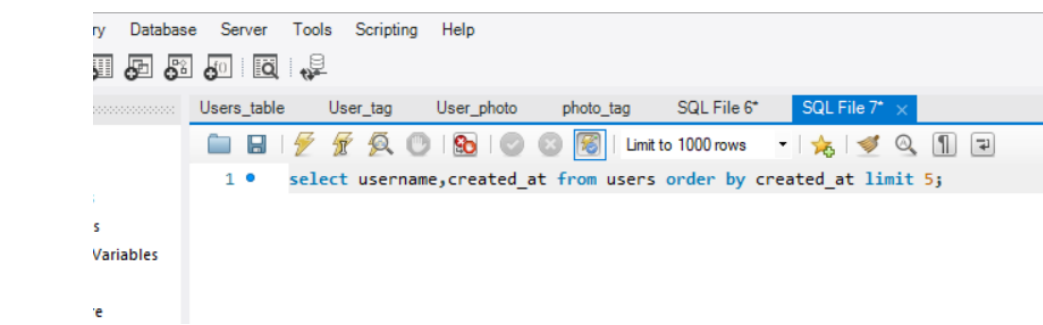
Tech Stack:

We worked on mysql workbench.

Sql Tasks:

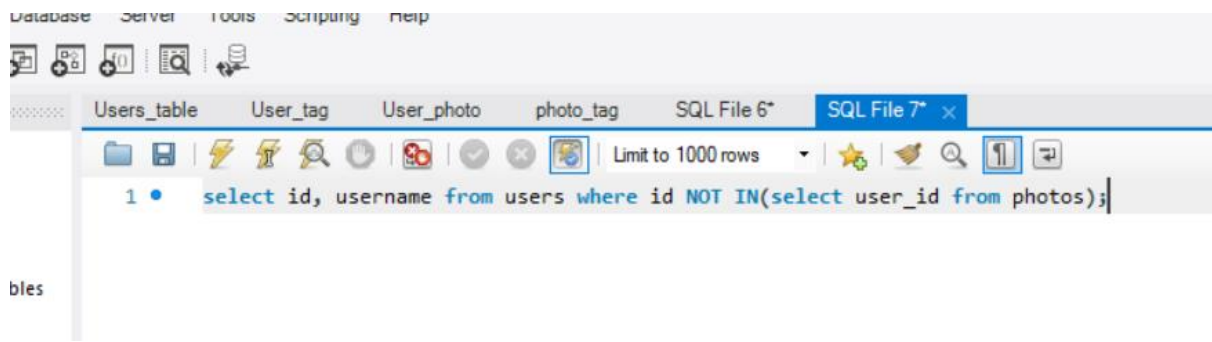
1) Marketing Analysis:

Most Loyal users:



Result: List of 5 oldest data is retrieved.

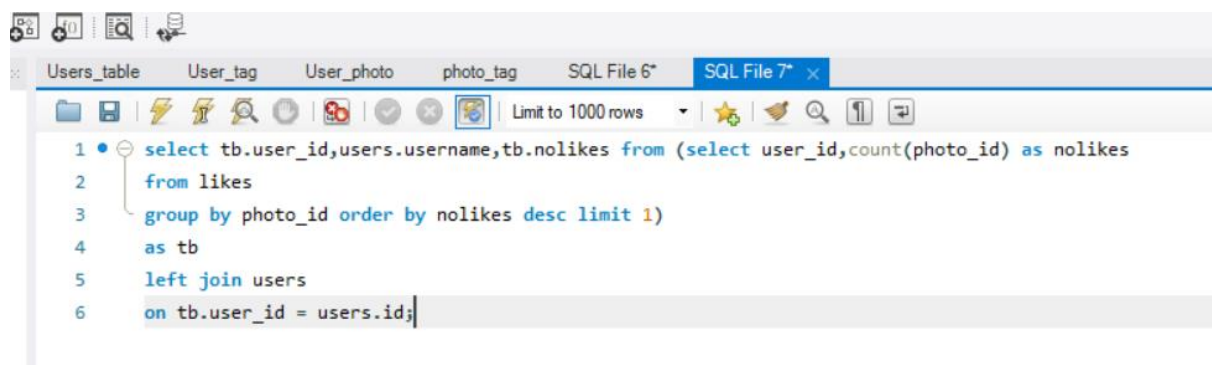
Inactive users:



A screenshot of a SQL IDE interface. The top menu bar includes 'Database', 'Server', 'Tools', 'Scripting', and 'Help'. Below the menu is a toolbar with various icons. The main window has several tabs: 'Users_table', 'User_tag', 'User_photo', 'photo_tag', 'SQL File 6*', and 'SQL File 7*'. The 'SQL File 7*' tab is active, showing a SQL query: `1 • select id, username from users where id NOT IN(select user_id from photos);`. The query is highlighted in blue. The status bar at the bottom shows 'Limit to 1000 rows'.

Result: Retrive total no of users inactive.

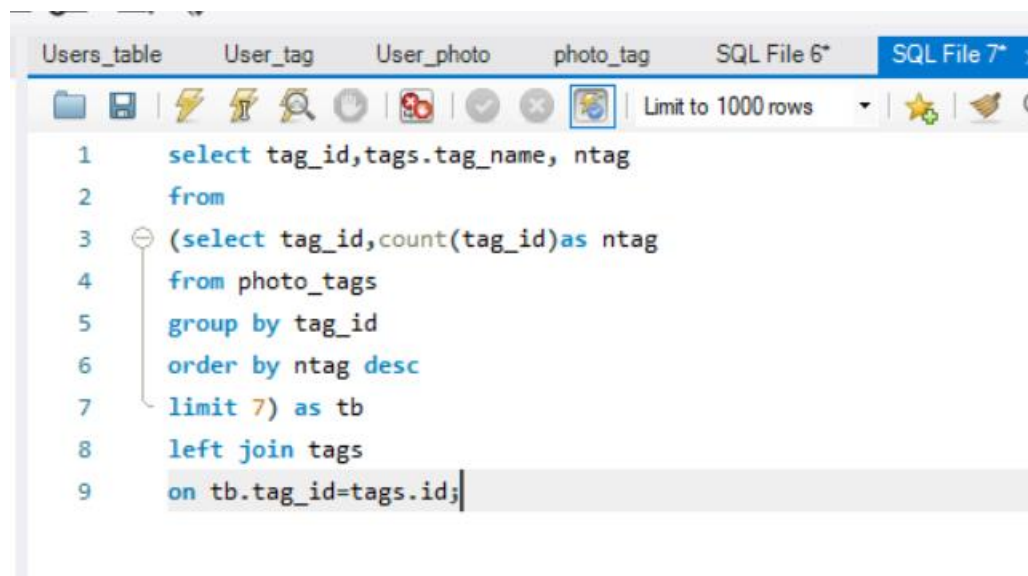
Contest Winner:



A screenshot of a SQL IDE interface. The top menu bar includes 'Database', 'Server', 'Tools', 'Scripting', and 'Help'. Below the menu is a toolbar with various icons. The main window has several tabs: 'Users_table', 'User_tag', 'User_photo', 'photo_tag', 'SQL File 6*', and 'SQL File 7*'. The 'SQL File 7*' tab is active, showing a SQL query: `1 • select tb.user_id,users.username,tb.nolikes from (select user_id,count(photo_id) as nolikes
2 from likes
3 group by photo_id order by nolikes desc limit 1)
4 as tb
5 left join users
6 on tb.user_id = users.id;`. The query is highlighted in blue. The status bar at the bottom shows 'Limit to 1000 rows'.

Result: retrieve the data who got more likes.

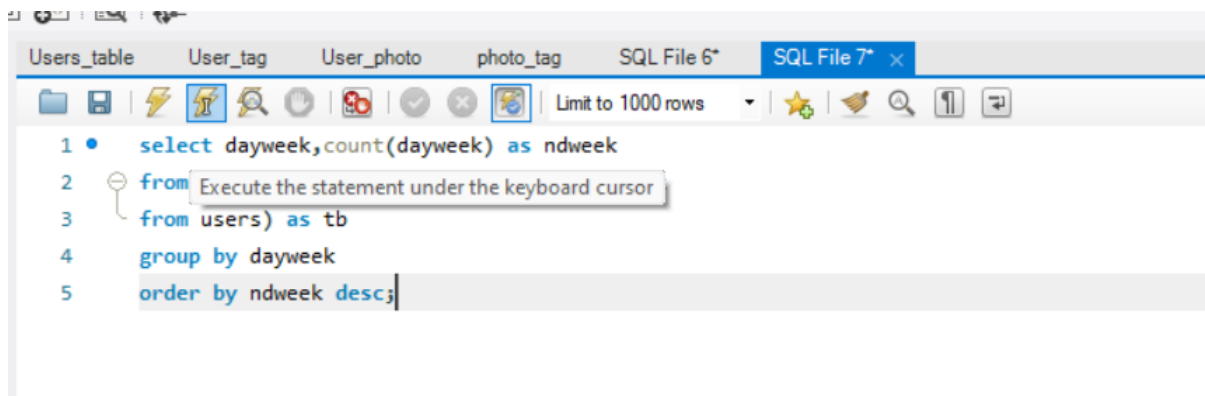
Popular Hashtags:



A screenshot of a SQL IDE interface. The top menu bar includes 'Database', 'Server', 'Tools', 'Scripting', and 'Help'. Below the menu is a toolbar with various icons. The main window has several tabs: 'Users_table', 'User_tag', 'User_photo', 'photo_tag', 'SQL File 6*', and 'SQL File 7*'. The 'SQL File 7*' tab is active, showing a SQL query: `1 select tag_id,tags.tag_name, ntag
2 from
3 (select tag_id,count(tag_id)as ntag
4 from photo_tags
5 group by tag_id
6 order by ntag desc
7 limit 7) as tb
8 left join tags
9 on tb.tag_id=tags.id;`. The query is highlighted in blue. The status bar at the bottom shows 'Limit to 1000 rows'.

Result: retrieve most used hashtags from user

Best day of the week:



The screenshot shows a SQL IDE with a query editor. The query is as follows:

```
1 • select dayweek, count(dayweek) as ndweek
2   from
3   from users) as tb
4   group by dayweek
5   order by ndweek desc;
```

A tooltip is visible over the 'from' keyword on line 2, stating: "Execute the statement under the keyboard cursor". The IDE interface includes a toolbar with various icons and a tab labeled "SQL File 7* x".

Result: Retrive the days with most user register.

2) Investor Analysis:

User Engagement:



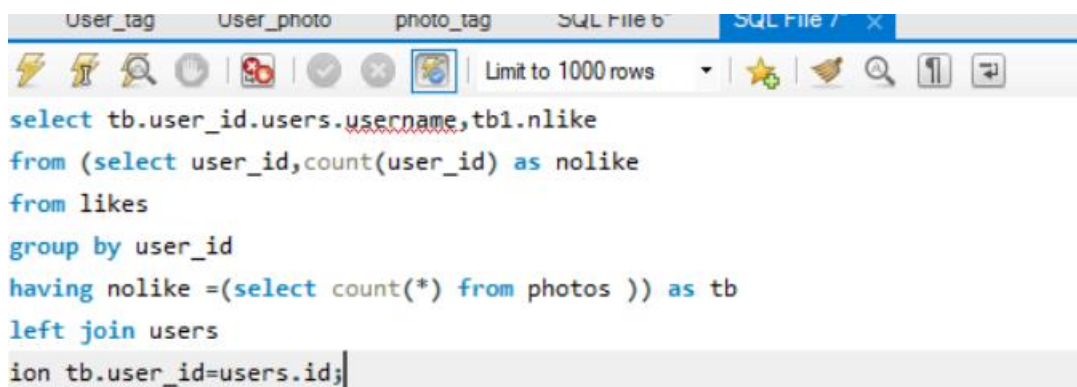
The screenshot shows a SQL IDE with a query editor. The query is as follows:

```
select (select count(*) from photos/(select count(*) from users) as navg);
```

The IDE interface includes a toolbar with various icons and a tab labeled "SQL File 7* x".

Result: On an average posted count.

Bots&FakeAccounts:



The screenshot shows a SQL IDE with a query editor. The query is as follows:

```
select tb.user_id, users.username, tb1.nlike
from (select user_id, count(user_id) as nlike
from likes
group by user_id
having nlike =(select count(*) from photos )) as tb
left join users
on tb.user_id=users.id;
```

The IDE interface includes a toolbar with various icons and a tab labeled "SQL File 7* x".

Result: retrieve the fake Account

