PROJECT - 3

Operation Analytics and Investigating Metric Spike

Advanced SQL

Case Study 1: Job Data Analysis

Task:

1. **Jobs Reviewed Over Time:**
   Objective: Calculate the number of jobs reviewed per hour for each day in November 2020.

   Task: **Write an SQL query to calculate the number of jobs reviewed per hour for each day in November 2020.**

   Query:

   ```
   select avg(t) as 'avg jobs reviewed per day per hour',
   avg(p) as 'avg jobs reviewed per day per second'
   from
   (select ds,((count(job_id)*3600)/sum(time_spent)) as t,
   ((count(job_id))/sum(time_spent)) as p from job_data
   where month(ds)=11 group by ds) a;
   ```

   Output:

```
avg jobs reviewed per day per hour        avg jobs reviewed per day
                                                     Per second
      126.18048333                                   0.03505000
```

2. **Throughput Analysis:**
   Objective: Calculate the 7-day rolling average of throughput (number of events per second).

   Task:Write an SQL query to calculate the 7-day rolling average of throughput. Additionally, explain whether you prefer using the daily metric or the 7-day rolling average for throughput, and why.

   Query: select round(count(event)/sum(time_spent),2)as 'weekly throughput' from job_data;

   select ds as Dates,round(count(event)/sum(time_spent),2) as 'Daily throughput' from job_data
   group by ds order by ds;

   Output:

Dates	Daily throughput
2020-11-25	**0.02**
2020-11-26	**0.02**
2020-11-27	**0.01**
2020-11-28	**0.06**
2020-11-29	**0.05**
2020-11-30	**0.05**

3. Language Share Analysis:

Objective: Calculate the percentage share of each language in the last 30 days.

Task: Write an SQL query to calculate the percentage share of each language over the last 30 days.

○

Query:

```
select language as
Languages,round(100*count(*)/total,2)AS
Percentage,sub.total
from job_data
cross join(select count(*)as total from job_data)as sub
group by language,sub.total;
```

Output:

Languages	Percentage	total
English	**12.50**	**8**
Arabic	**12.50**	**8**
Persian	**37.50**	**8**
Hindi	**12.50**	**8**
French	**12.50**	**8**
italian	**12.50**	**8**

4. Duplicate Rows Detection:

Objective: Identify duplicate rows in the data.

Task: Write an SQL query to display duplicate rows from the job_data table.

Query: select actor_id,count(*) as Duplicates from job_data group by actor_id having count(*)>1;

Output:

actor_id	Duplicates
1003	**2**

Case Study 2: Investigating Metric Spike

1.Weekly User Engagement:
Objective: Measure the activeness of users on a weekly basis.

Task: Write an SQL query to calculate the weekly user engagement.

Query:

```
select extract(week from occurred_at) as week_number,
count(distinct user_id)as active_user
from events
where event_type='engagement'
group by week_number
order by week_number;
```

Output:

Week number	active_user
17	663
18	1068
19	1113
20	1154
21	1121
22	1186
23	1232
24	1275
25	1264
26	1302
27	1372
28	1365
29	1376
30	1467
31	1299

32	1225
33	1225
34	1204
35	104

2.User Growth Analysis:

Objective: Analyze the growth of users over time for a product.

Task: Write an SQL query to calculate the user growth for the product.

Query:

```
SELECT extract(MONTH from created_at) as
MonthNum, COUNT(DISTINCT user_id) AS total_users
 FROM users
 WHERE activated_at NOT IN ("")
 group by MonthNum;
 SELECT extract(MONTH from created_at) as MonthNum,
COUNT(DISTINCT user_id) as Users,
ROUND(((COUNT(DISTINCT
user_id)/LAG(COUNT(DISTINCT user_id), 1)
  OVER (ORDER BY  extract(MONTH from created_at)) -
1)*100),2)
 as 'Growth %'FROM users WHERE activated_at NOT IN ("")
group by MonthNum;
```

Output:

Monthnum	users	Growth%
1	712	
2	685	-3.79
3	765	11.68

4	907	18.56
5	993	9.48
6	1086	9.37
7	1281	17.96
8	1347	5.15
9	330	-75.50
10	390	18.18
11	399	2.31
12	486	21.80

3.Weekly Retention Analysis:

Objective: Analyze the retention of users on a weekly basis after signing up for a product.

Task: Write an SQL query to calculate the weekly retention of users based on their sign-up cohort.

Query:

```
select extract(week from occurred_at) as sign_up_week,count(distinct user_id) as 'count of users sign up'
from events
 where event_type = 'signup_flow'
 and event_name = 'complete_signup'
 group by sign_up_week;
```

Output:

sign_up_week	count of user signup
17	72
18	163
19	185
20	176

21	**183**
22	**196**
23	**196**
24	**229**
25	**207**
26	**201**
27	**222**
28	**215**
29	**221**
30	**238**
31	**193**
32	**245**
33	**261**
34	**259**
35	**18**

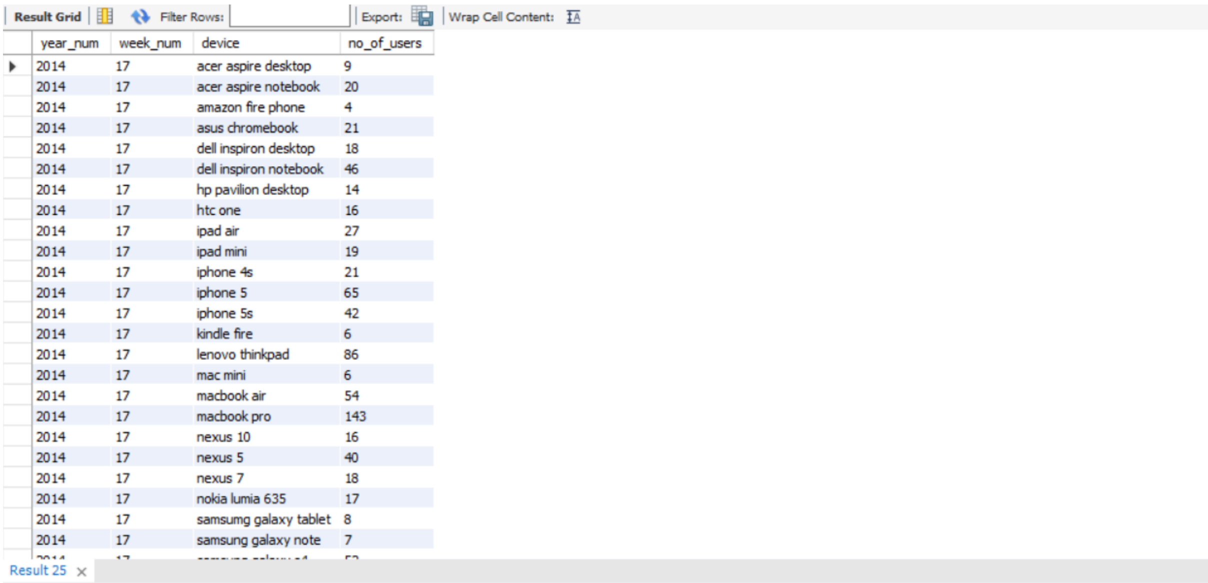
4.Weekly Engagement Per Device:

Objective: Measure the activeness of users on a weekly basis per device.

Task: Write an SQL query to calculate the weekly engagement per device.

Query:
```
select
extract(year from occurred_at) as year_num,
extract(week from occurred_at) as week_num,
device,
count(distinct user_id) as no_of_users
from events
where event_type = 'engagement'
group by 1,2,3
order by 1,2,3;
```

Output:



year_num	week_num	device	no_of_users
2014	17	acer aspire desktop	9
2014	17	acer aspire notebook	20
2014	17	amazon fire phone	4
2014	17	asus chromebook	21
2014	17	dell inspiron desktop	18
2014	17	dell inspiron notebook	46
2014	17	hp pavilion desktop	14
2014	17	htc one	16
2014	17	ipad air	27
2014	17	ipad mini	19
2014	17	iphone 4s	21
2014	17	iphone 5	65
2014	17	iphone 5s	42
2014	17	kindle fire	6
2014	17	lenovo thinkpad	86
2014	17	mac mini	6
2014	17	macbook air	54
2014	17	macbook pro	143
2014	17	nexus 10	16
2014	17	nexus 5	40
2014	17	nexus 7	18
2014	17	nokia lumia 635	17
2014	17	samsumg galaxy tablet	8
2014	17	samsung galaxy note	7

continue.....(total 491 rows)

5.Email Engagement Analysis:

Objective: Analyze how users are engaging with the email service.

Task: Write an SQL query to calculate the email engagement metrics.

Query: select

```
100.0 * sum(case when email_cat = 'email_opened' then 1
else 0 end)
        /sum(case when email_cat = 'email_sent' then 1 else 0
end)
as email_opening_rate,
100.0 * sum(case when email_cat = 'email_clicked' then 1
else 0 end)
        /sum(case when email_cat = 'email_sent' then 1 else 0
end)
as email_clicking_rate
```

```
from
(
select *,
case when action in ('sent_weekly_digest',
'sent_reengagement_email')
    then 'email_sent'
    when action in ('email_open')
    then 'email_opened'
    when action in ('email_clickthrough')
    then 'email_clicked'
end as email_cat
from events
)a;
```

Output:

```
email_opening_rate          email_clicking_rate
    31.1921                       10.4745
```