Trainity
Task 3 : Operation Analytics and Investigating Metric Spike.

By-Syed Ali Ashraf

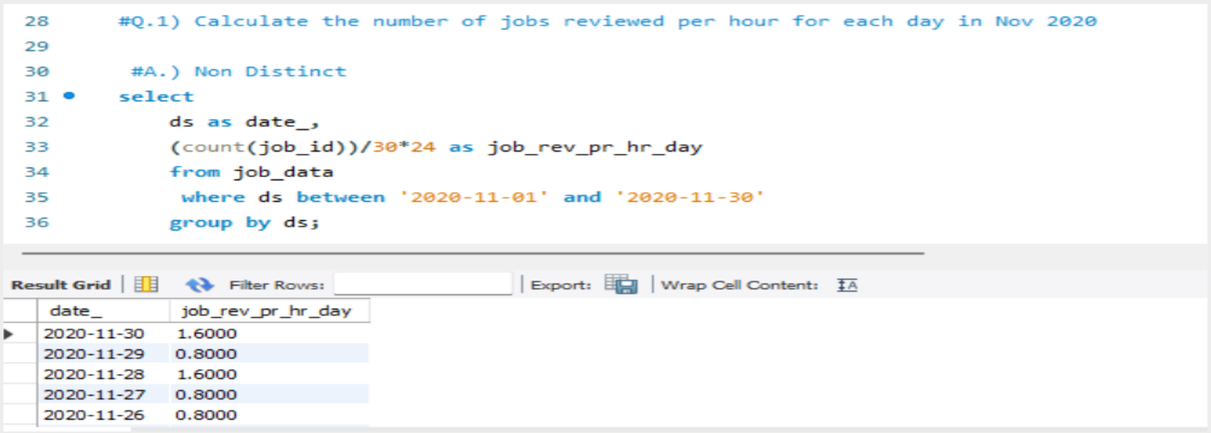
Case Study 1 : Job Data.

Case Study 2: Investigating metric spike.

Case Study 1: Job Data

Q.1) Write an SQL query to calculate the number of jobs reviewed per hour for each day in November 2020.

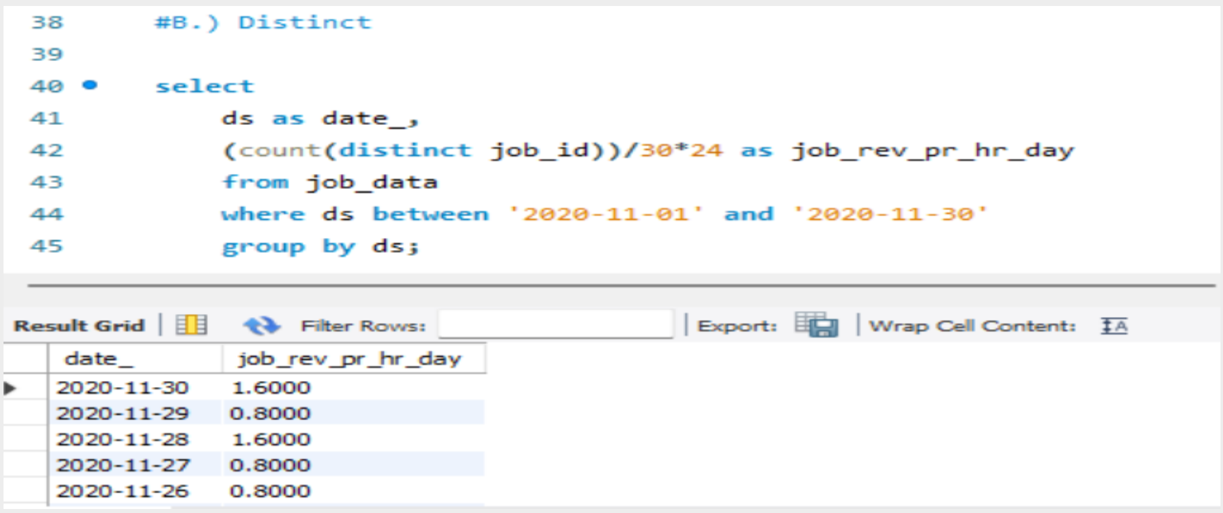
Solution: A) Non-Distinct



```
#Q.1) Calculate the number of jobs reviewed per hour for each day in Nov 2020

 #A.) Non Distinct
select
    ds as date_,
    (count(job_id))/30*24 as job_rev_pr_hr_day
    from job_data
     where ds between '2020-11-01' and '2020-11-30'
    group by ds;
```

```
Result Grid | Filter Rows: | Export: | Wrap Cell Content:
  date_        job_rev_pr_hr_day
▶ 2020-11-30   1.6000
  2020-11-29   0.8000
  2020-11-28   1.6000
  2020-11-27   0.8000
  2020-11-26   0.8000
```

B) Distinct

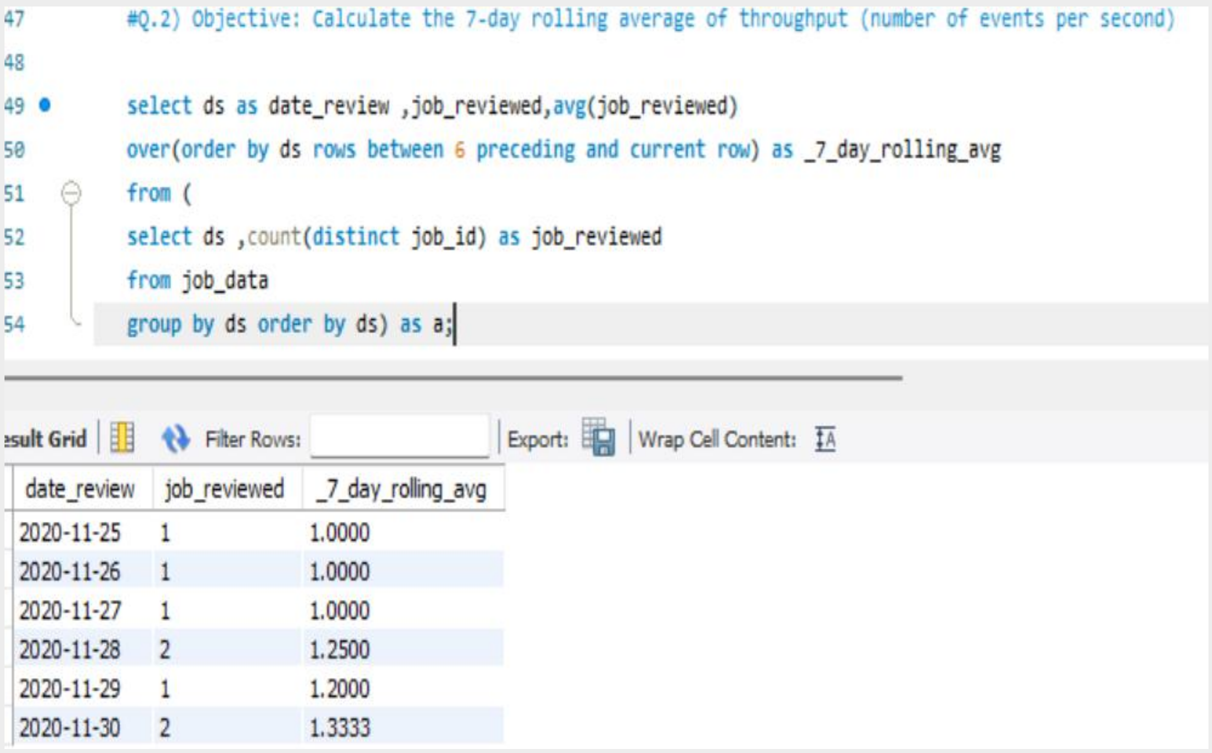


```
#B.) Distinct

select
    ds as date_,
    (count(distinct job_id))/30*24 as job_rev_pr_hr_day
    from job_data
    where ds between '2020-11-01' and '2020-11-30'
    group by ds;
```

```
Result Grid | Filter Rows: | Export: | Wrap Cell Content:
  date_        job_rev_pr_hr_day
▶ 2020-11-30   1.6000
  2020-11-29   0.8000
  2020-11-28   1.6000
  2020-11-27   0.8000
  2020-11-26   0.8000
```

Q.2) 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.

Solution:

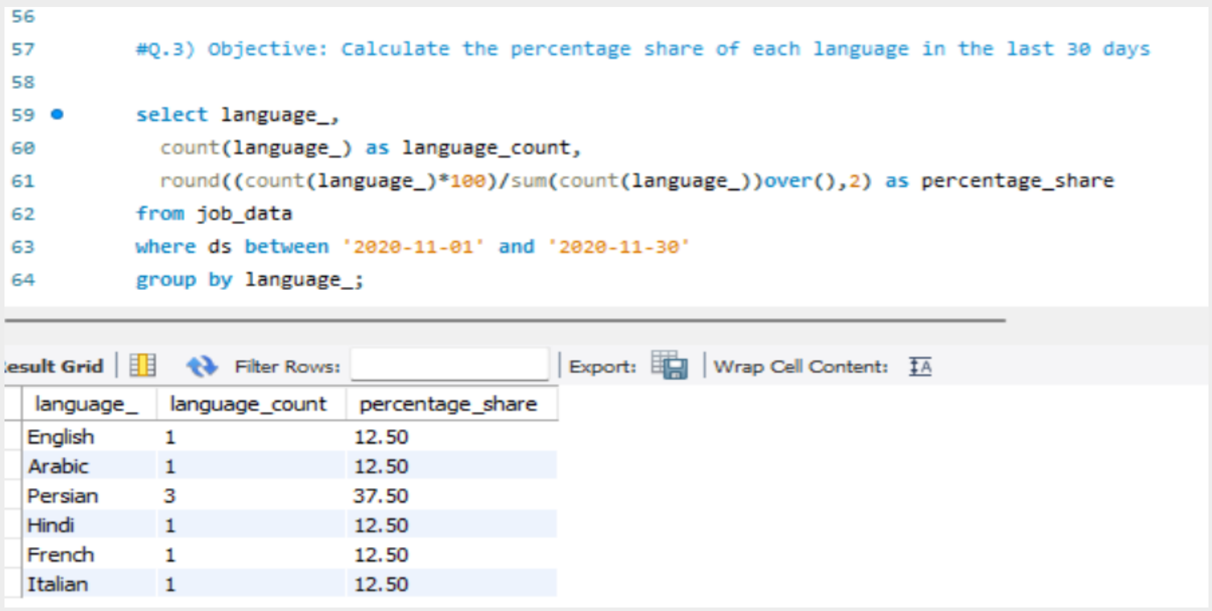


```sql
#Q.2) Objective: Calculate the 7-day rolling average of throughput (number of events per second)

select ds as date_review ,job_reviewed,avg(job_reviewed)
over(order by ds rows between 6 preceding and current row) as _7_day_rolling_avg
from (
select ds ,count(distinct job_id) as job_reviewed
from job_data
group by ds order by ds) as a;
```

date_review	job_reviewed	_7_day_rolling_avg
2020-11-25	1	1.0000
2020-11-26	1	1.0000
2020-11-27	1	1.0000
2020-11-28	2	1.2500
2020-11-29	1	1.2000
2020-11-30	2	1.3333

Q.3) Write an SQL query to calculate the percentage share of each language over the last 30 days.

Solution:

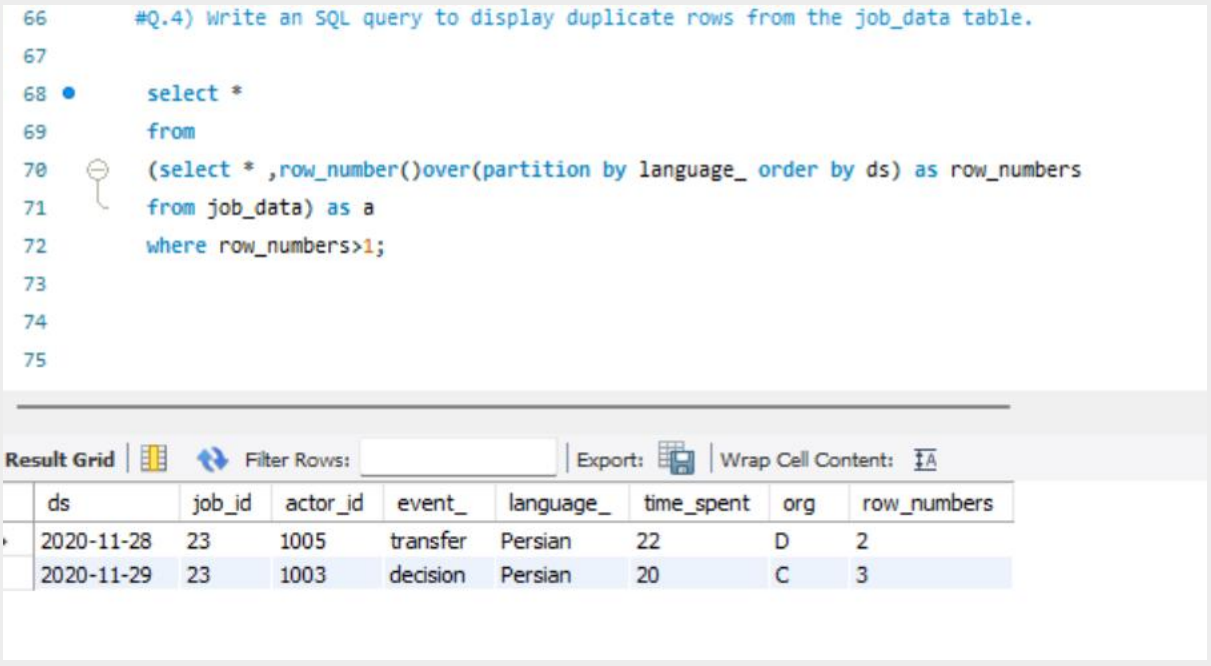


```sql
#Q.3) Objective: Calculate the percentage share of each language in the last 30 days

select language_,
  count(language_) as language_count,
  round((count(language_)*100)/sum(count(language_))over(),2) as percentage_share
from job_data
where ds between '2020-11-01' and '2020-11-30'
group by language_;
```

language_	language_count	percentage_share
English	1	12.50
Arabic	1	12.50
Persian	3	37.50
Hindi	1	12.50
French	1	12.50
Italian	1	12.50

Q.4) Write an SQL query to display duplicate rows from the job_data table.

Solution:



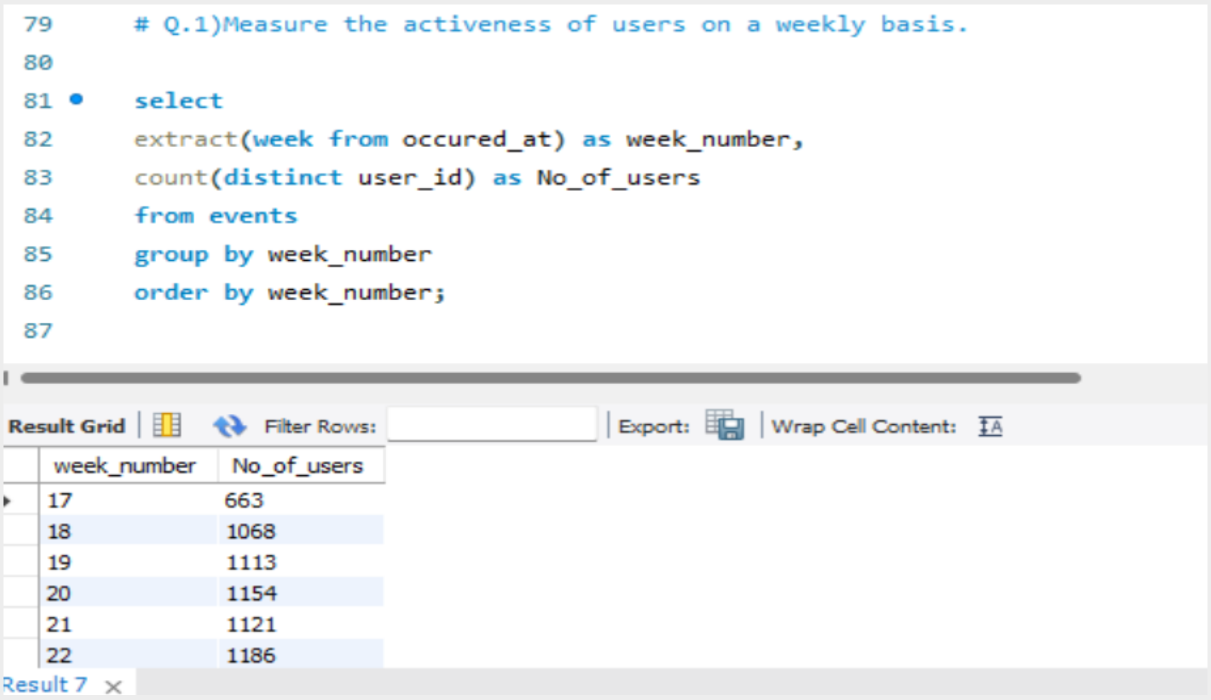
```
#Q.4) Write an SQL query to display duplicate rows from the job_data table.

 select *
 from
 (select * ,row_number()over(partition by language_ order by ds) as row_numbers
 from job_data) as a
 where row_numbers>1;
```

```
ds          job_id  actor_id  event_    language_  time_spent  org  row_numbers
2020-11-28  23      1005      transfer  Persian    22          D    2
2020-11-29  23      1003      decision  Persian    20          C    3
```

Case Study 2: Investigating Metric Spike

Q.1) Write an SQL query to calculate the weekly user engagement.

Solution:

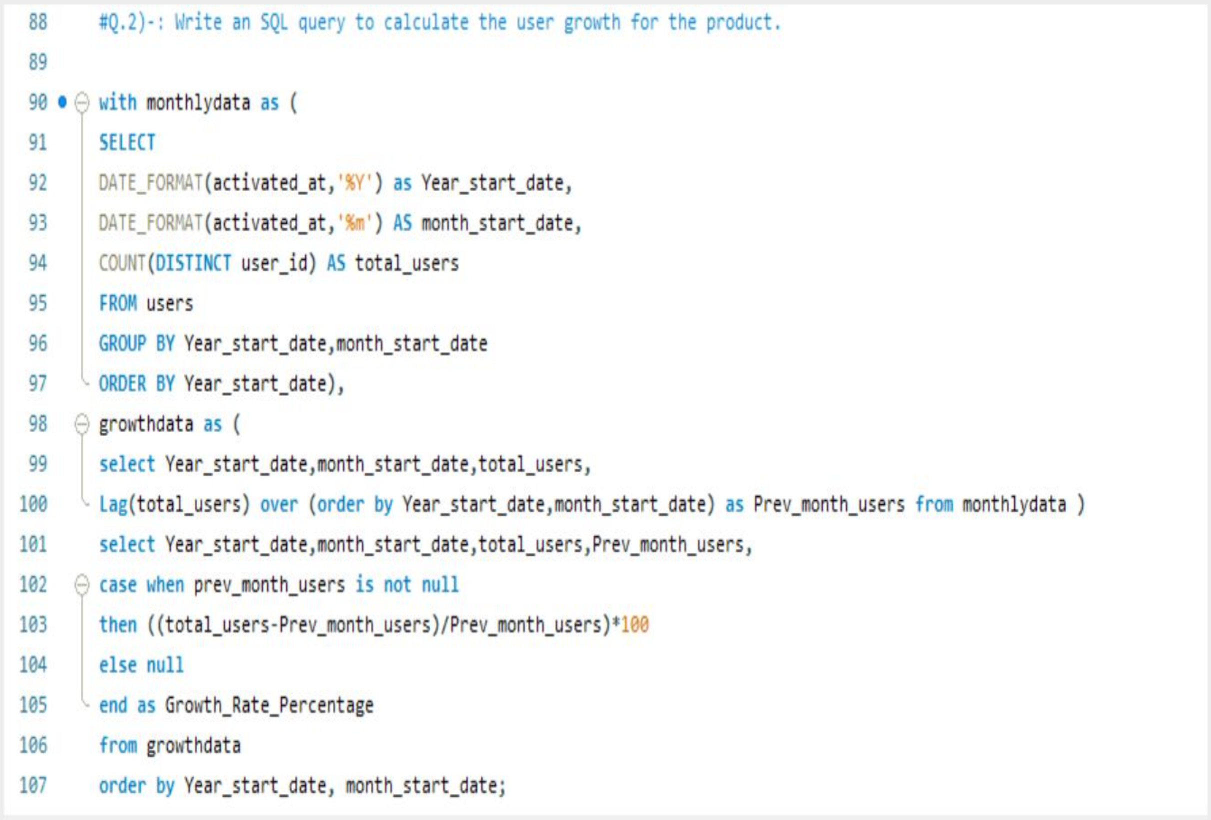


```
# Q.1)Measure the activeness of users on a weekly basis.

select
extract(week from occured_at) as week_number,
count(distinct user_id) as No_of_users
from events
group by week_number
order by week_number;
```

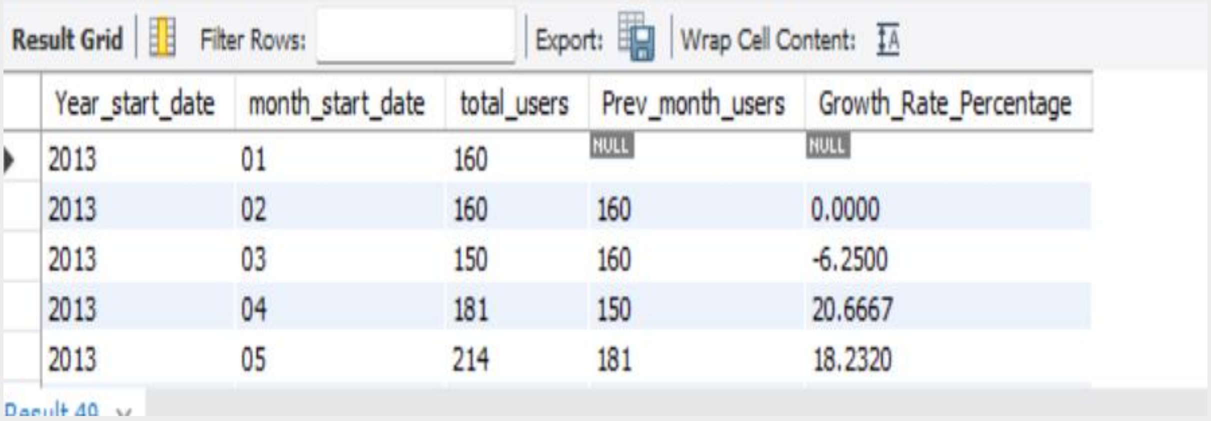
```
week_number  No_of_users
17           663
18           1068
19           1113
20           1154
21           1121
22           1186
```

Q.2) Write an SQL query to calculate the user growth for the product.

Solution:



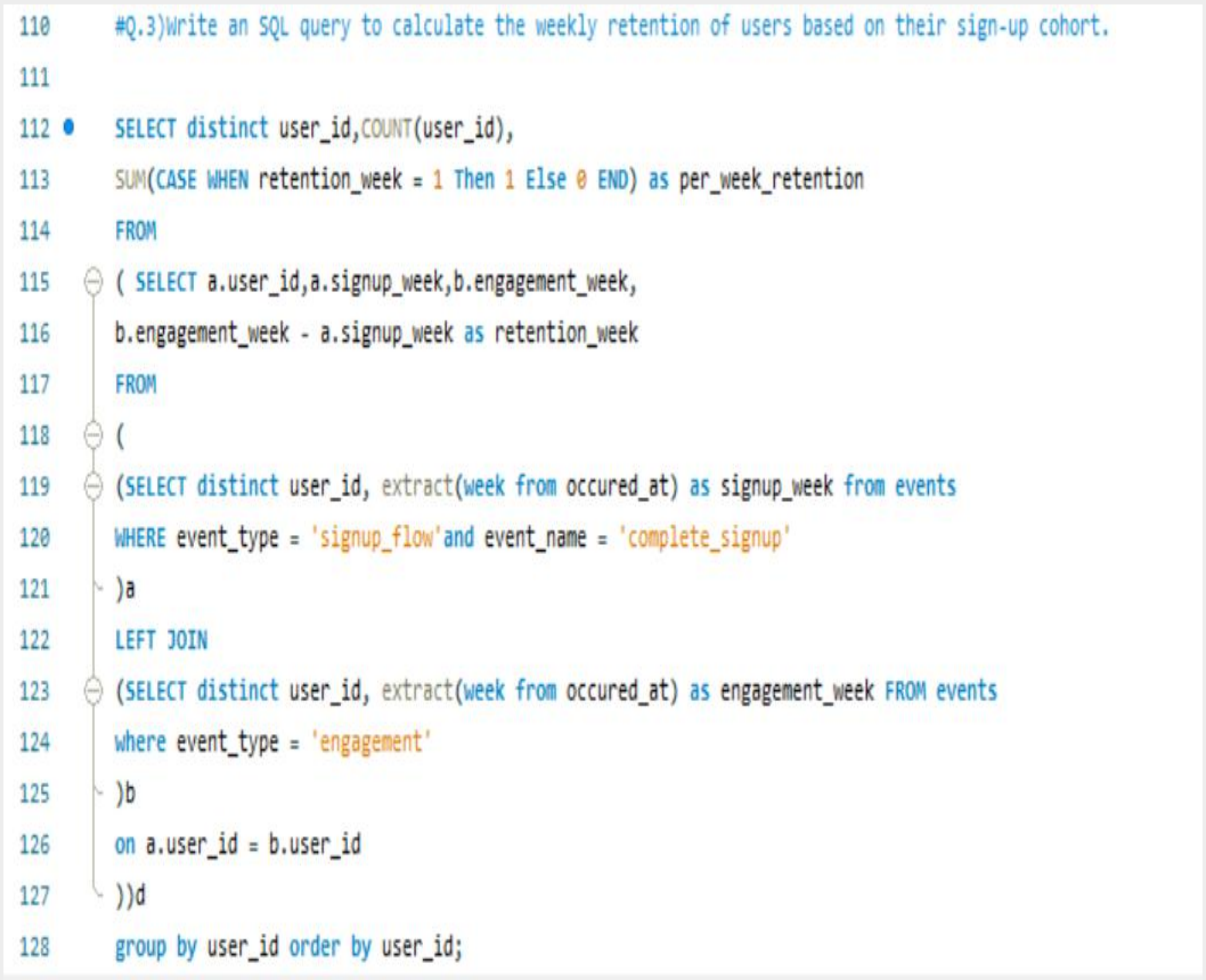
```
#Q.2)-: Write an SQL query to calculate the user growth for the product.

with monthlydata as (
SELECT
DATE_FORMAT(activated_at,'%Y') as Year_start_date,
DATE_FORMAT(activated_at,'%m') AS month_start_date,
COUNT(DISTINCT user_id) AS total_users
FROM users
GROUP BY Year_start_date,month_start_date
ORDER BY Year_start_date),
growthdata as (
select Year_start_date,month_start_date,total_users,
Lag(total_users) over (order by Year_start_date,month_start_date) as Prev_month_users from monthlydata )
select Year_start_date,month_start_date,total_users,Prev_month_users,
case when prev_month_users is not null
then ((total_users-Prev_month_users)/Prev_month_users)*100
else null
end as Growth_Rate_Percentage
from growthdata
order by Year_start_date, month_start_date;
```



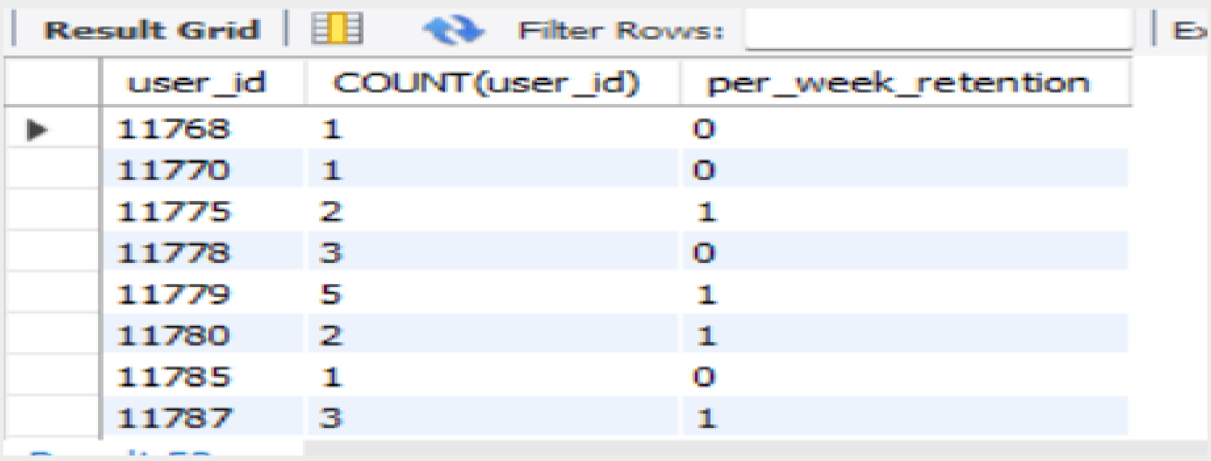
Year_start_date	month_start_date	total_users	Prev_month_users	Growth_Rate_Percentage
2013	01	160	NULL	NULL
2013	02	160	160	0.0000
2013	03	150	160	-6.2500
2013	04	181	150	20.6667
2013	05	214	181	18.2320

Q.3) Write an SQL query to calculate the weekly retention of users based on their sign-up cohort.

Solution:



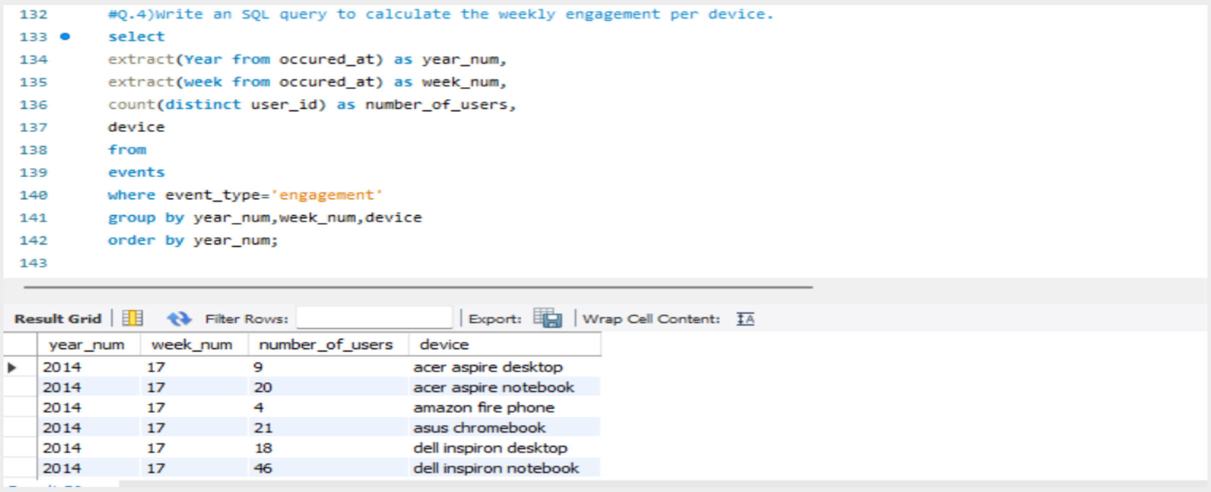
```
#Q.3)Write an SQL query to calculate the weekly retention of users based on their sign-up cohort.

SELECT distinct user_id,COUNT(user_id),
SUM(CASE WHEN retention_week = 1 Then 1 Else 0 END) as per_week_retention
FROM
( SELECT a.user_id,a.signup_week,b.engagement_week,
b.engagement_week - a.signup_week as retention_week
FROM
(
(SELECT distinct user_id, extract(week from occured_at) as signup_week from events
WHERE event_type = 'signup_flow'and event_name = 'complete_signup'
)a
LEFT JOIN
(SELECT distinct user_id, extract(week from occured_at) as engagement_week FROM events
where event_type = 'engagement'
)b
on a.user_id = b.user_id
))d
group by user_id order by user_id;
```



user_id	COUNT(user_id)	per_week_retention
11768	1	0
11770	1	0
11775	2	1
11778	3	0
11779	5	1
11780	2	1
11785	1	0
11787	3	1

Q.4) Write an SQL query to calculate the weekly engagement per device.

Solution:

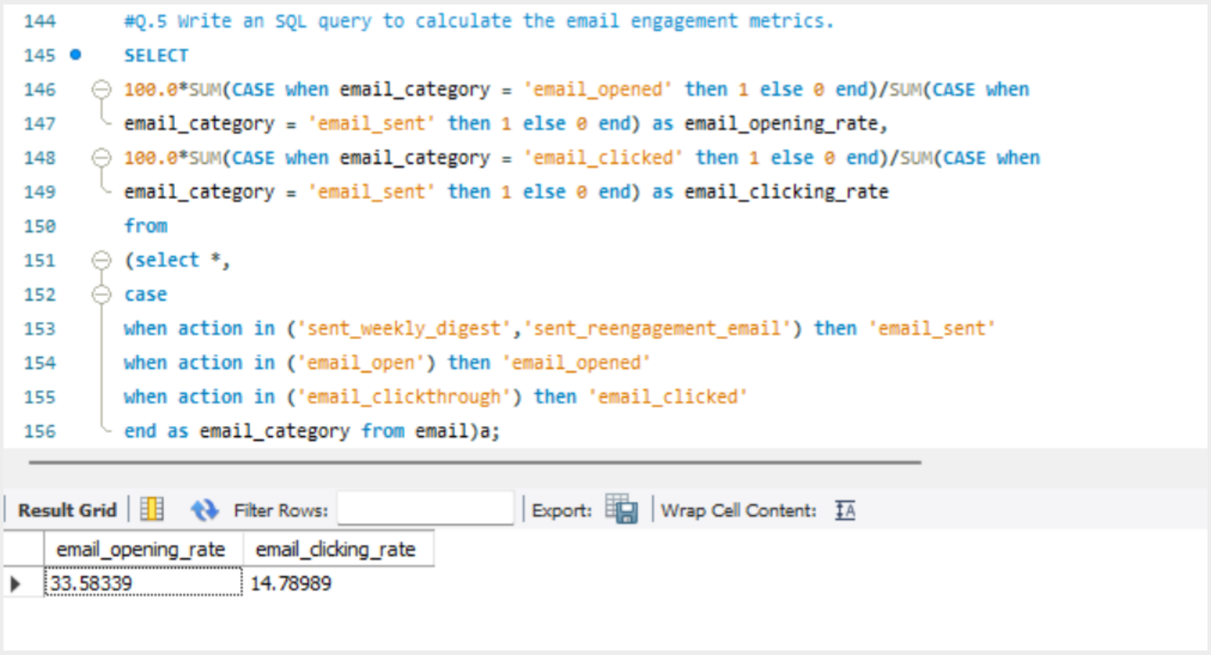


```
#Q.4)Write an SQL query to calculate the weekly engagement per device.
select
extract(Year from occured_at) as year_num,
extract(week from occured_at) as week_num,
count(distinct user_id) as number_of_users,
device
from
events
where event_type='engagement'
group by year_num,week_num,device
order by year_num;
```

year_num	week_num	number_of_users	device
2014	17	9	acer aspire desktop
2014	17	20	acer aspire notebook
2014	17	4	amazon fire phone
2014	17	21	asus chromebook
2014	17	18	dell inspiron desktop
2014	17	46	dell inspiron notebook

Q.5) Write an SQL query to calculate the email engagement metrics.

Solution:



```
#Q.5 Write an SQL query to calculate the email engagement metrics.
SELECT
100.0*SUM(CASE when email_category = 'email_opened' then 1 else 0 end)/SUM(CASE when
email_category = 'email_sent' then 1 else 0 end) as email_opening_rate,
100.0*SUM(CASE when email_category = 'email_clicked' then 1 else 0 end)/SUM(CASE when
email_category = '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_category from email)a;
```

email_opening_rate	email_clicking_rate
33.58339	14.78989

Software Used: MySQL Workbench 8.0

Thank You.