Operation Analytics and Investigating Metric Spike

A. **Project Description :-** . Analyzing a company's entire operational cycle is a critical step in the process of operational analytics. Finding areas for improvement within the organization is made easier by this analysis. In this project, with various datasets and tablesI have to derive insights from this data to answer questions posed by different departments within the company by using advanced SQL skills to analyze the data and provide valuable insights that can help improve the company's operations and understand sudden changes in key metrics.

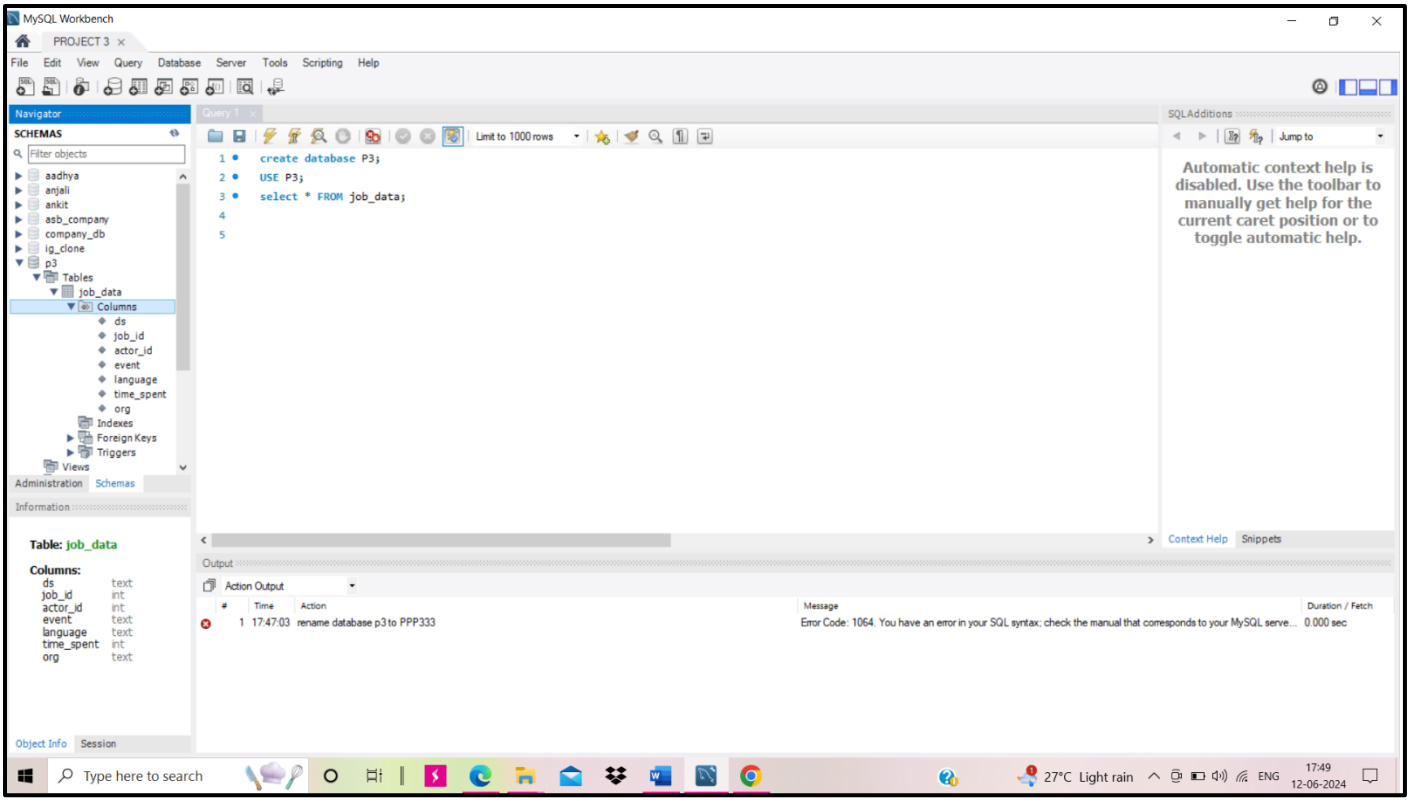
B. **Project approach :-** First the csv file data was imported in table of sql and then by using advance SQL skills the questions solutions were obtained.

C. **Tech-Stack Used :-** The tech stack included MySQL Workbench v8.0.30.0, a great tool for database queries because of its accuracy, speed, simplicity, and ease of use. It enables you to work with database objects, design, create, and browse your schemas; it also lets you design and execute SQL queries to manipulate with data that has been stored. Also Microsoft Excel was used to import/export data , Microsoft word was used to create presentation pdf document.

Project Insight

Case Study 1: Job Data Analysis

Before going for analysis all the raw data i.e CSV file is imported in the MY SQL in p3 database for futher analysis is inserted in MYSQL which can be seen in the following screenshots :-



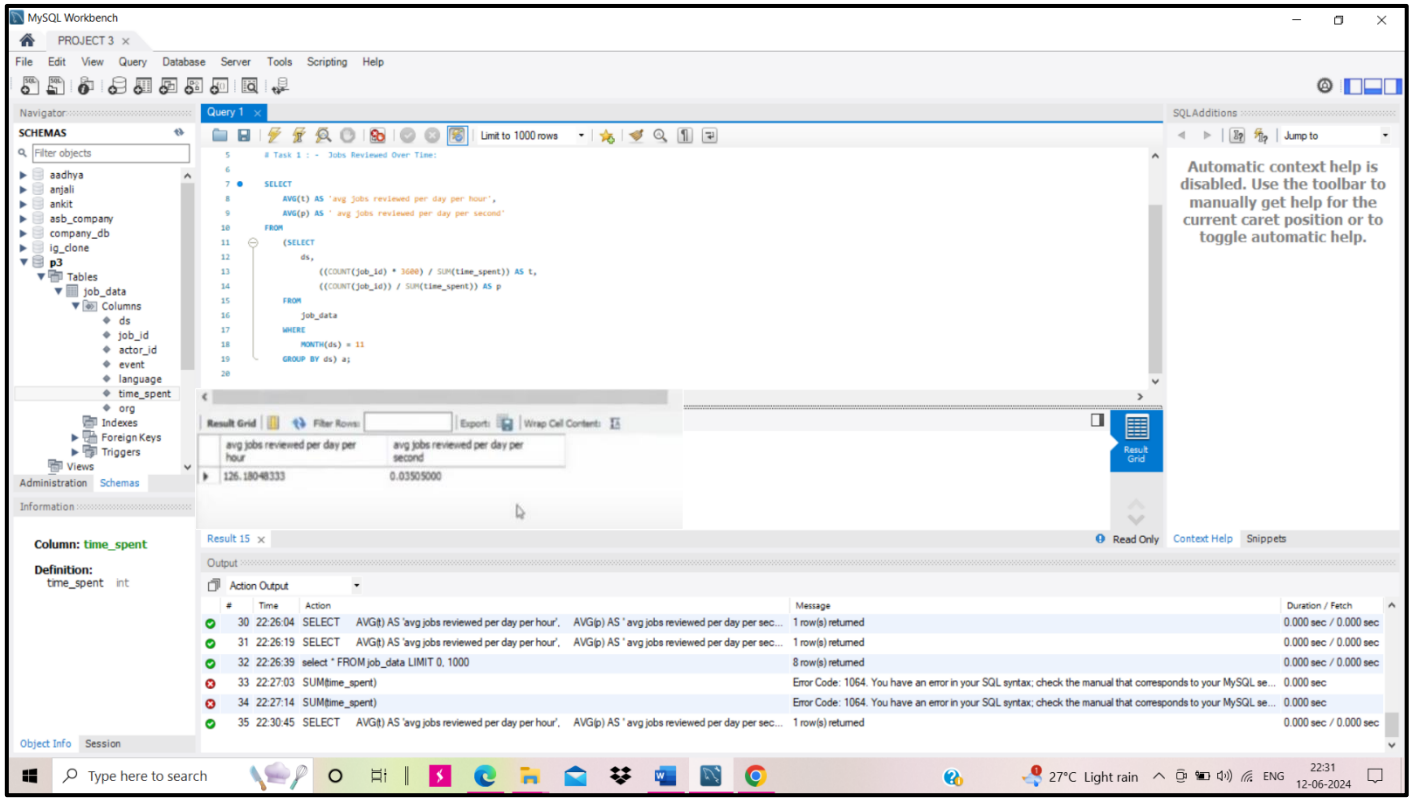
A) Jobs Reviewed Over Time:

- **Objective:To Calculate the number of jobs reviewed per hour for each day in November 2020.**

Conclusion /result :- avg jobs reviewed per day per hour =126.18048333

avg jobs reviewed per day per second = 0.03505000

➢ screenshot of code and after running code its output



Code used :-

```
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;
```

(B) Throughput Analysis

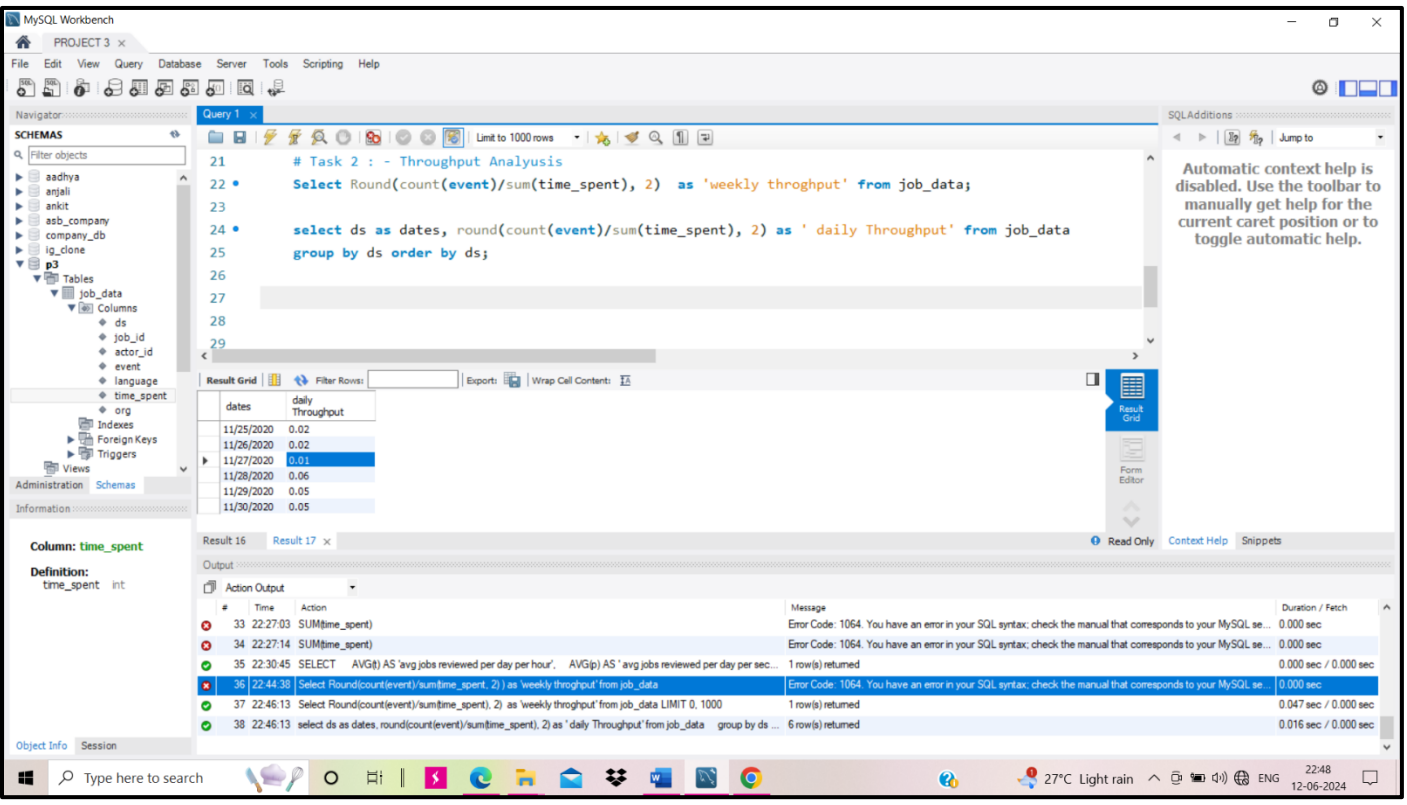
- **Objective: Calculate the 7-day rolling average of throughput (number of events per second).**

Conclusion /result :-

Dates	Daily Throughput
11/25/2020	0.02
11/26/2020	0.02
11/27/2020	0.01
11/28/2020	0.06
11/29/2020	0.05
11/30/2020	0.05

Weekly Throughput 0.03

➢ sccreenshot of code and after running code its output



Code used :-

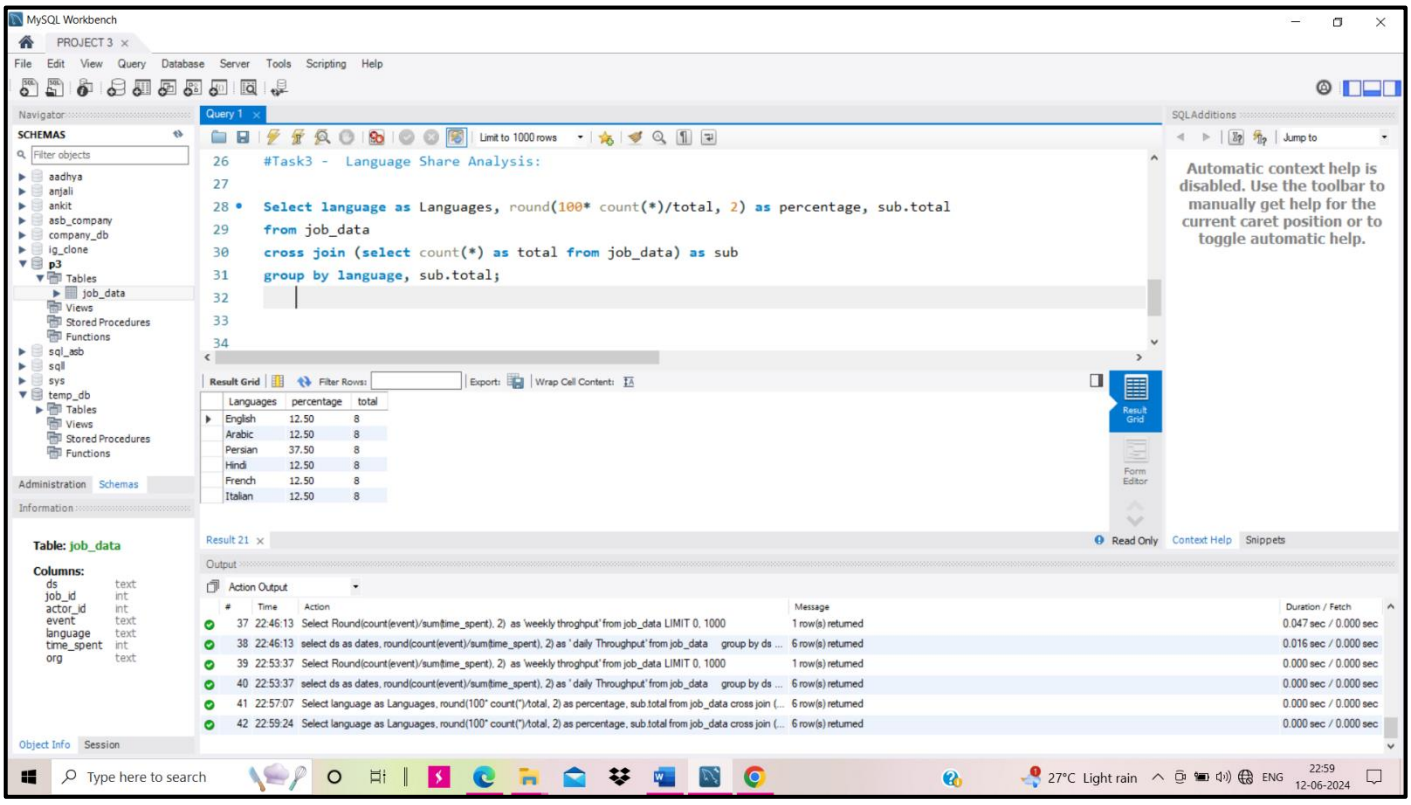
```
Select Round(count(event)/sum(time_spent), 2)  as 'weekly throghput' from job_data;

      select ds as dates, round(count(event)/sum(time_spent), 2) as ' daily Throughput' from job_data

   group by ds order by ds;
```

(C)Language Share Analysis:

- **Objective: Calculate the percentage share of each language in the last 30 days.**
- Conclusion /result :-
-

Languages	percentage	total
Arabic	12.50	8
English	12.50	8
French	12.50	8
Hindi	12.50	8
Italian	12.50	8
Persian	37.50	8

➢ screenshot of code and after running code its output



Code used :-

```
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;
```

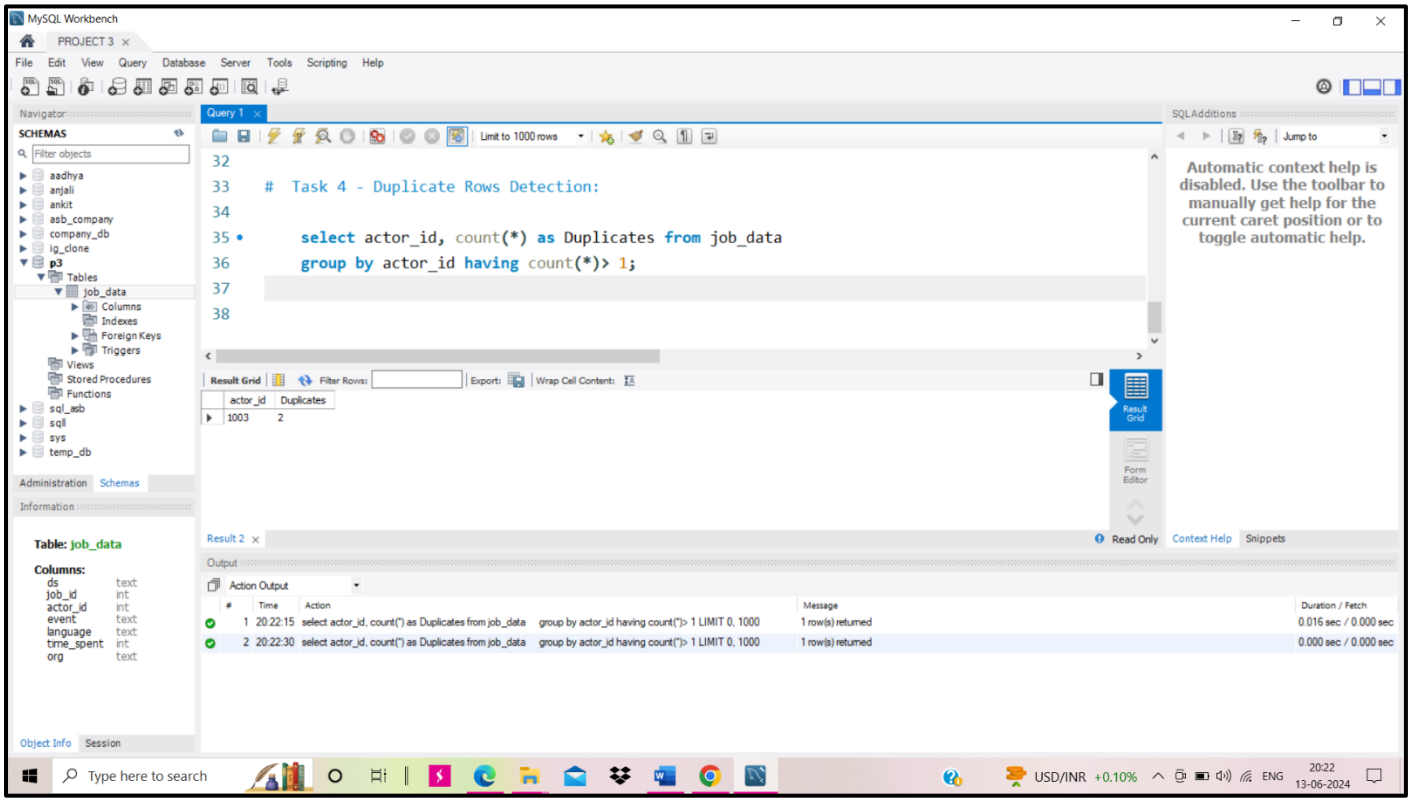
[4] Duplicate Rows Detection:

- **Objective: Identify duplicate rows in the data.**

- Conclusion/Result :-

Actor_id	duplicates
1003	2

- screenshot of code and after running code its output



Code used :-

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

Case Study 2: Investigating Metric Spike

There are three tables with which are doing this case study 2 as follows:-

- **users**: Contains one row per user, with descriptive information about that user's account.
- **events**: Contains one row per event, where an event is an action that a user has taken (e.g., login, messaging, search).
- **email_events**: Contains events specific to the sending of emails.

Before going for analysis all the raw data i.e CSV file is imported in the MY SQL in " case 2" database for futher which can be seen in the following screenshots :-

TABLE 1 – USERS

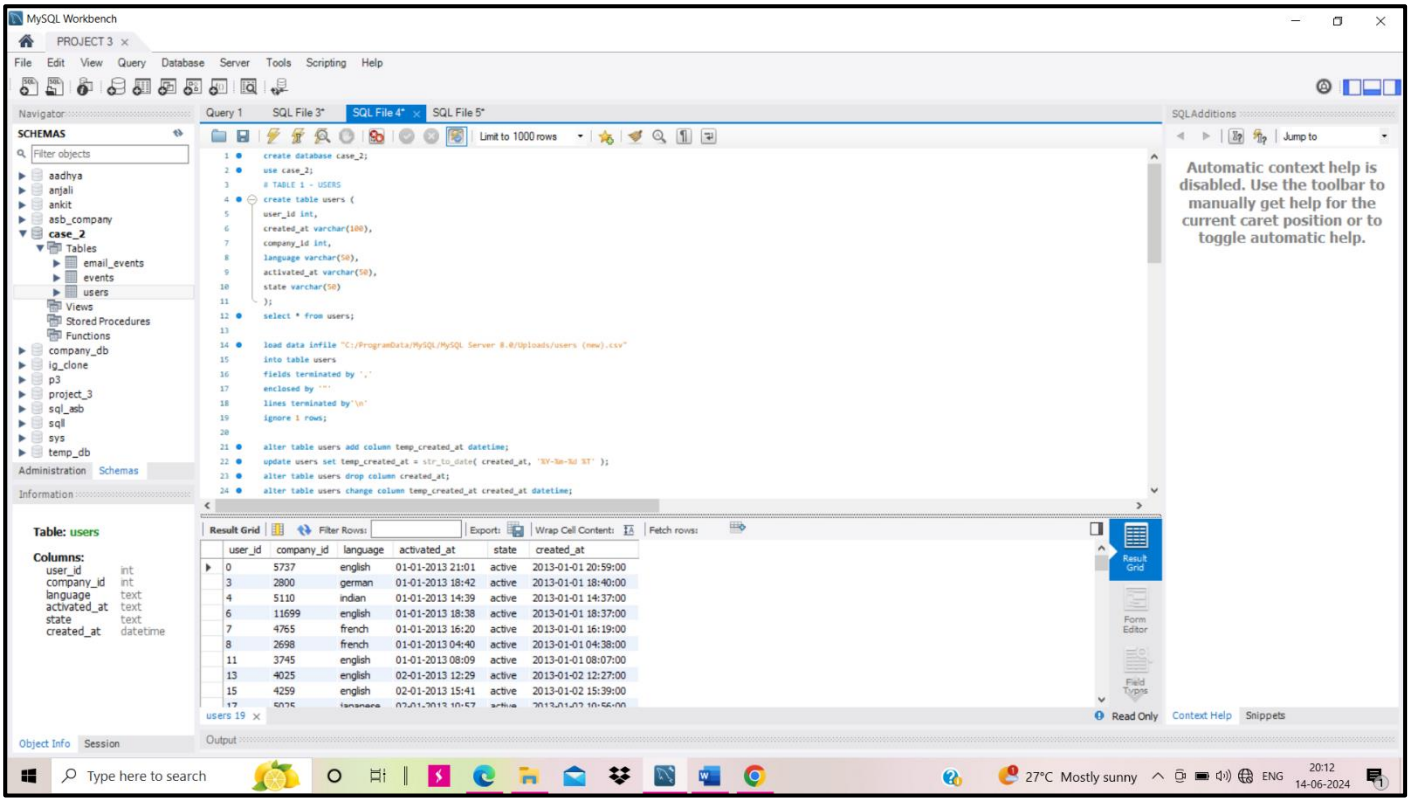


TABLE 2 –Events

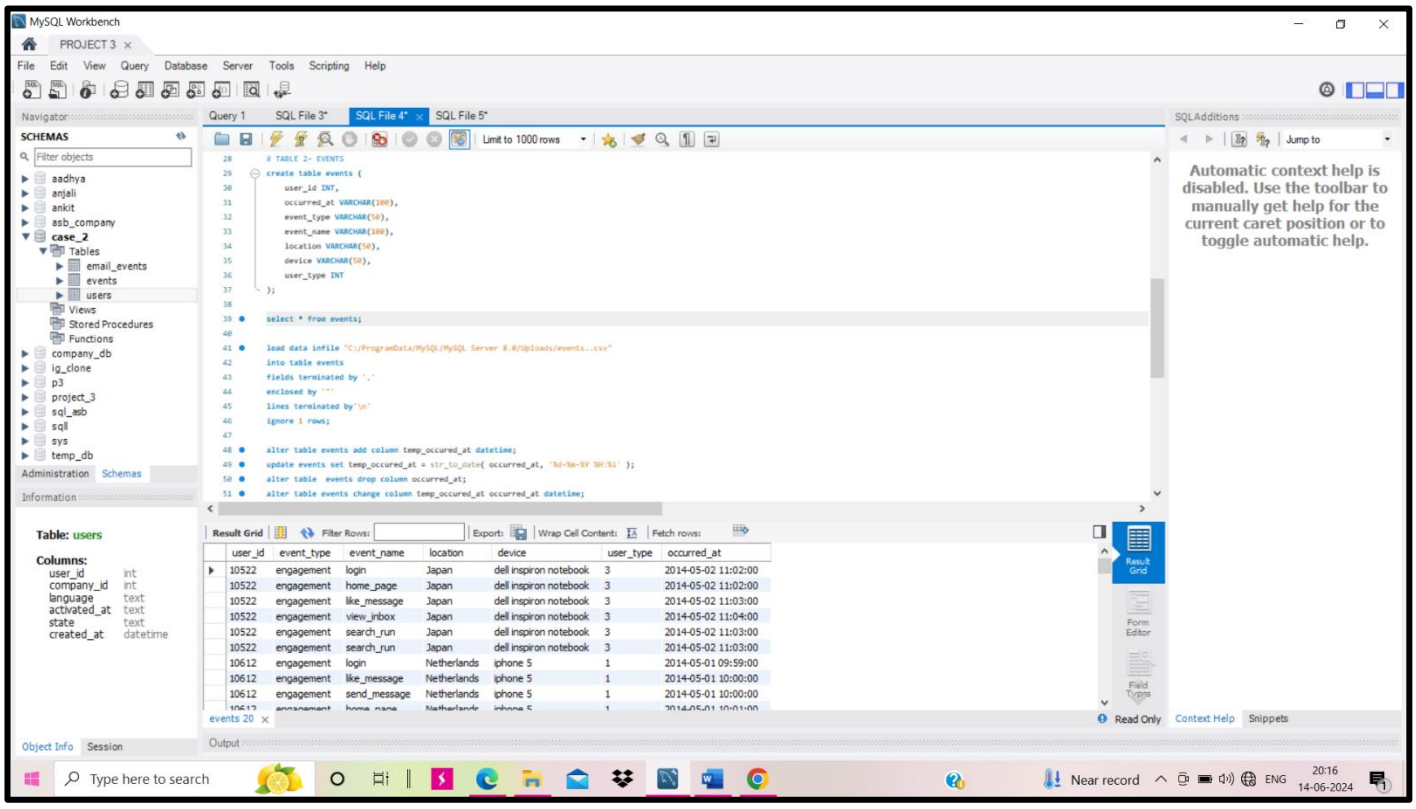
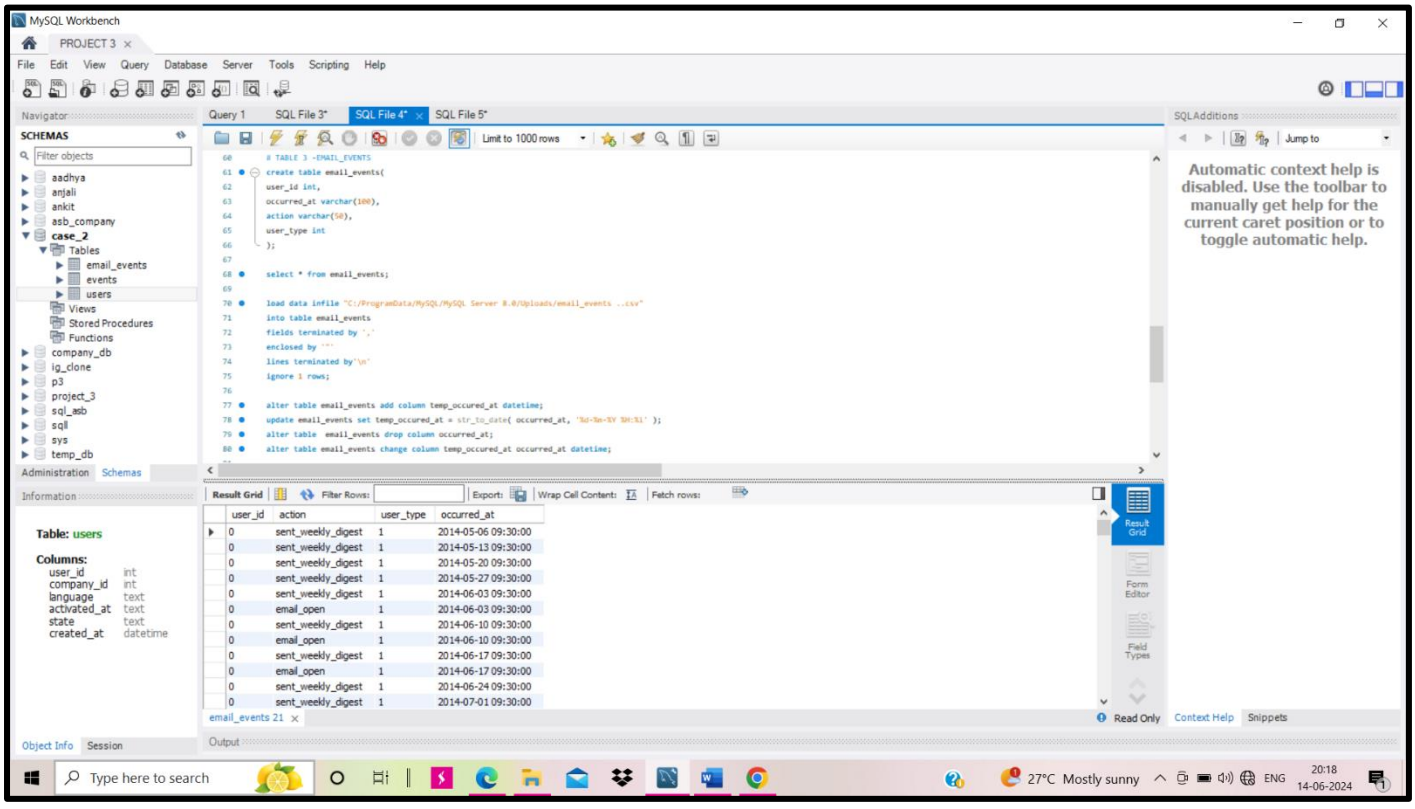


TABLE 3 –Email_events



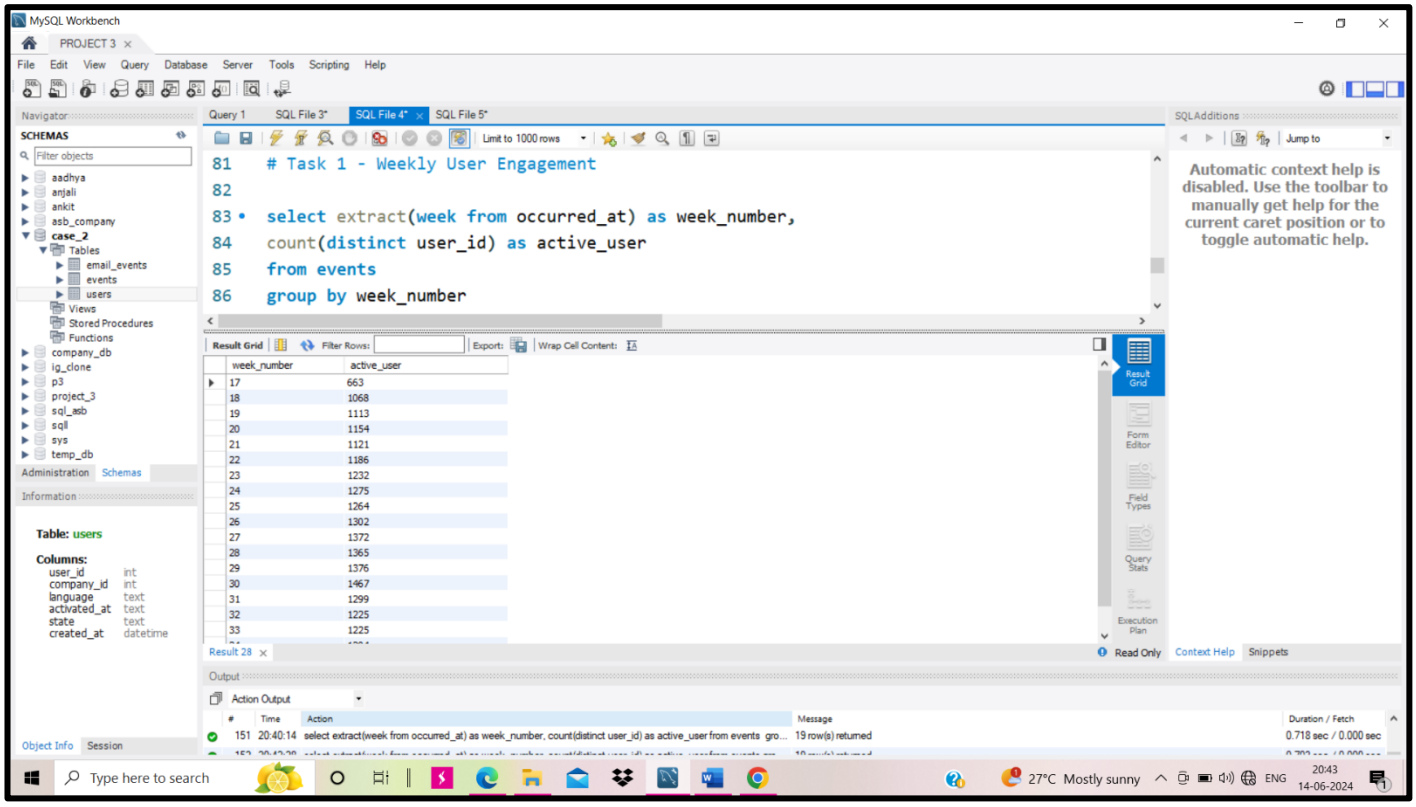
Task 1 - Weekly User Engagement:

- **Objective: Measure the activeness of users on a weekly basis.**
- **Conclusion/Result :-**

Week_number	Active_user
17	663
18	1068
19	1113
20	1154
21	1121
22	1186
23	1232
24	1275
25	1264

Week_number	Active_user
26	1302
27	1372
28	1365
29	1376
30	1467
31	1299
32	1225
33	1225
34	1204
35	104

- Screenshot of code and after running code its output



Code used:-

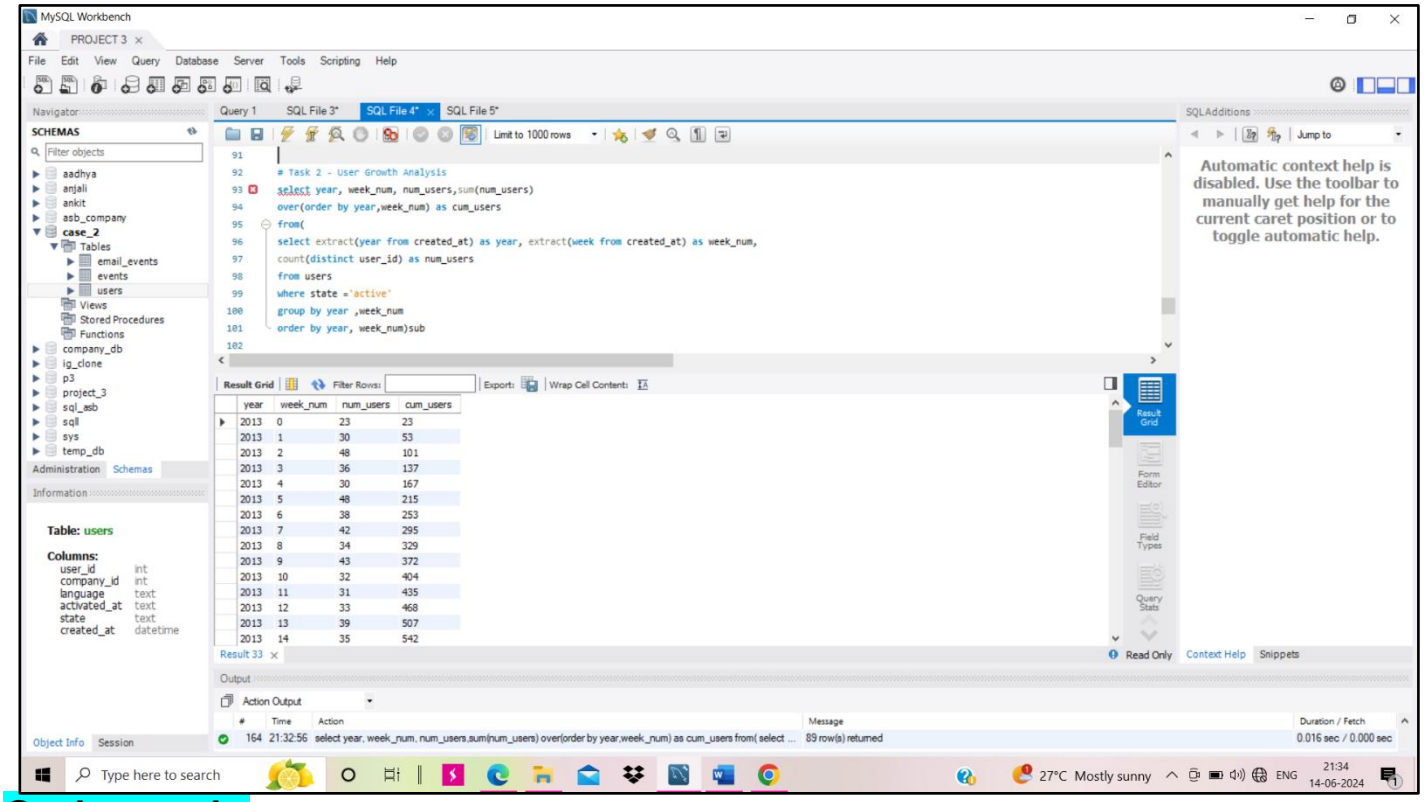
```
select extract(week from occurred_at) as week_number,
count(distinct user_id) as active_user
from events
group by week_number
```

Task 2 - User Growth Analysis

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

- **Conclusion/Result :- 33th week of 2014 has greatest number of users i.e 261 , The lowest number of users are 35th week of 2014 i.e 18 seen from the output obtained.**

➢ Screenshot of code and after running code its output



Code used:-

```
select year, week_num, num_users,sum(num_users)
over(order by year,week_num) as cum_users
from(
select extract(year from created_at) as year, extract(week from created_at) as week_num,
count(distinct user_id) as num_users
from users
where state ='active'
group by year ,week_num
order by year, week_num)sub
```

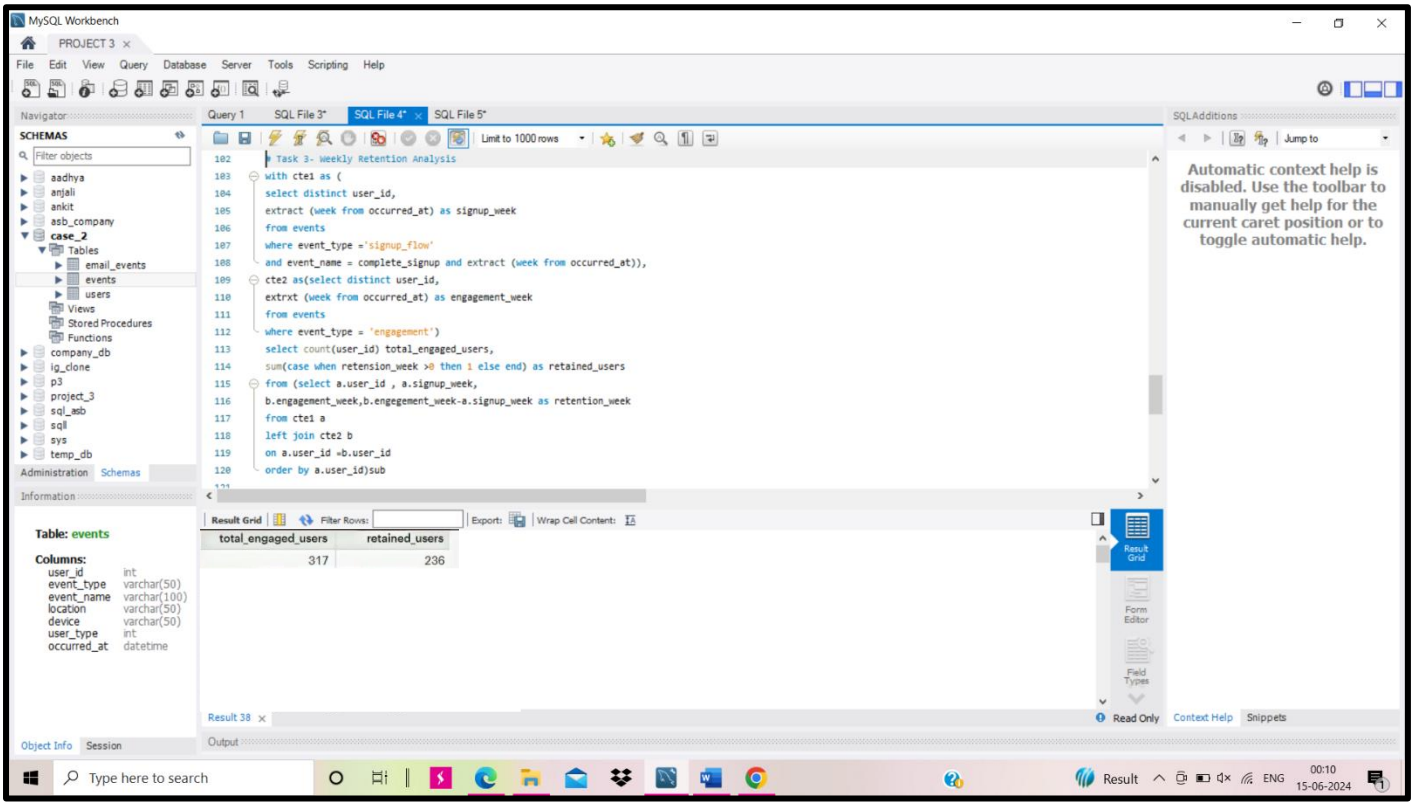
Task 3- Weekly Retention Analysis

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

- **Conclusion/Result:-**

Total engaged users	Total retained users
317	236

Screenshot of code and after running code its output



Code used:-

```
with cte1 as (
select distinct user_id,
extract (week from occurred_at) as signup_week
from events
where event_type ='signup_flow'
and event_name = complete_signup and extract (week from occurred_at)),
cte2 as(select distinct user_id,
extrxt (week from occurred_at) as engagement_week
```

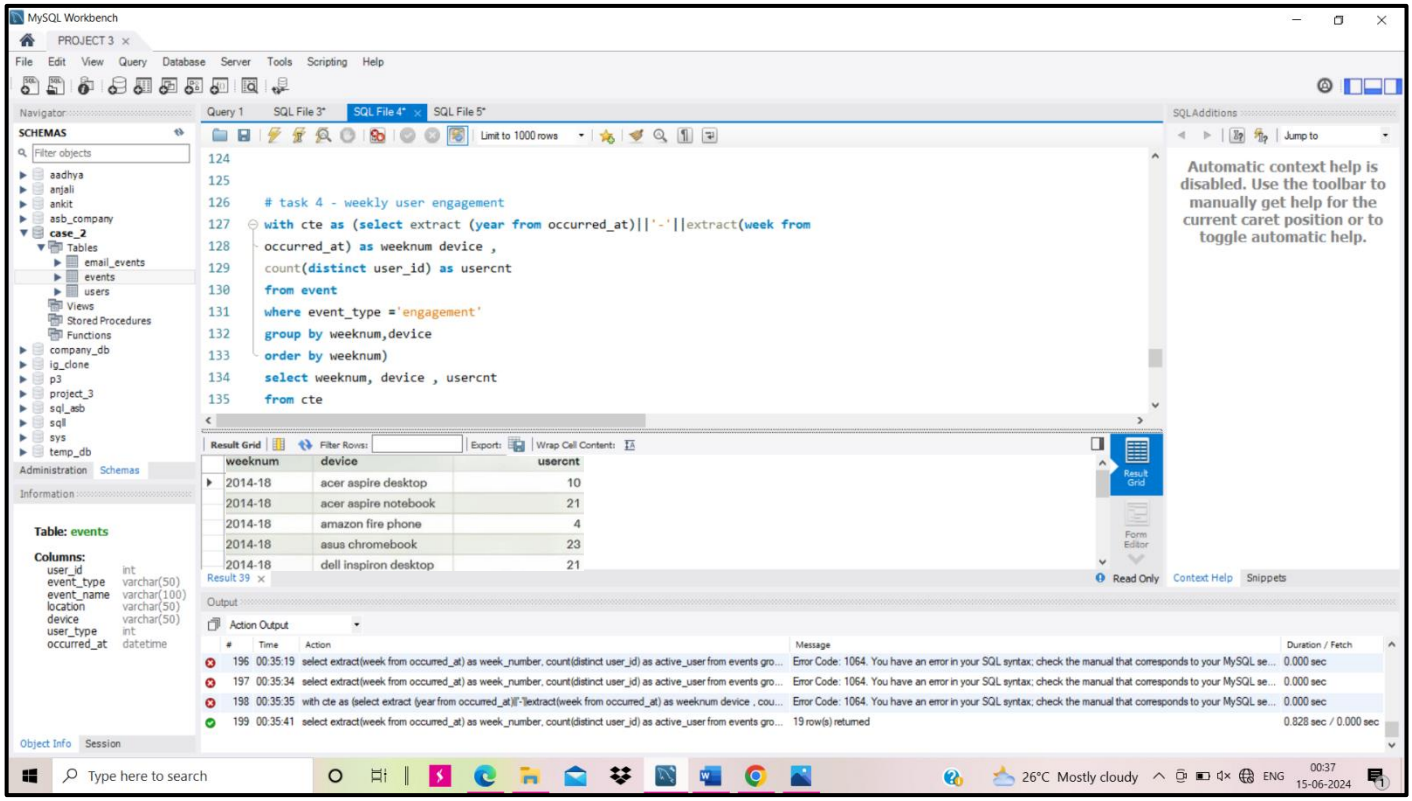
```
from events
where event_type = 'engagement')
select count(user_id) total_engaged_users,
sum(case when retension_week >0 then 1 else end) as retained_users
from (select a.user_id , a.signup_week,
b.engagement_week,b.engegement_week-a.signup_week as retention_week
from cte1 a
left join cte2 b
on a.user_id =b.user_id
order by a.user_id)sub
```

Task 4 -Weekly Engagement Per Device

- **Objective: Measure the activeness of users on a weekly basis per device**
- **Conclusion/Result:-**

weeknum	device	Usercnt
2014-18	Acer aspire desktop	10
2014-18	Acer aspire notebook	21
2014-18	Amazon fire phone	4
2014-18	Asus chromebook	23
2014-18	Dell inspiron desktop	21

- **Screenshot of code and after running code its output**



Code used - with cte as (select extract (year from occurred_at)||'-'||extract(week from

occurred_at) as weeknum device ,

count(distinct user_id) as usercnt

from event

where event_type ='engagement'

group by weeknum,device

order by weeknum)

select weeknum, device , usercnt

Task 5 - Email Engagement Analysis

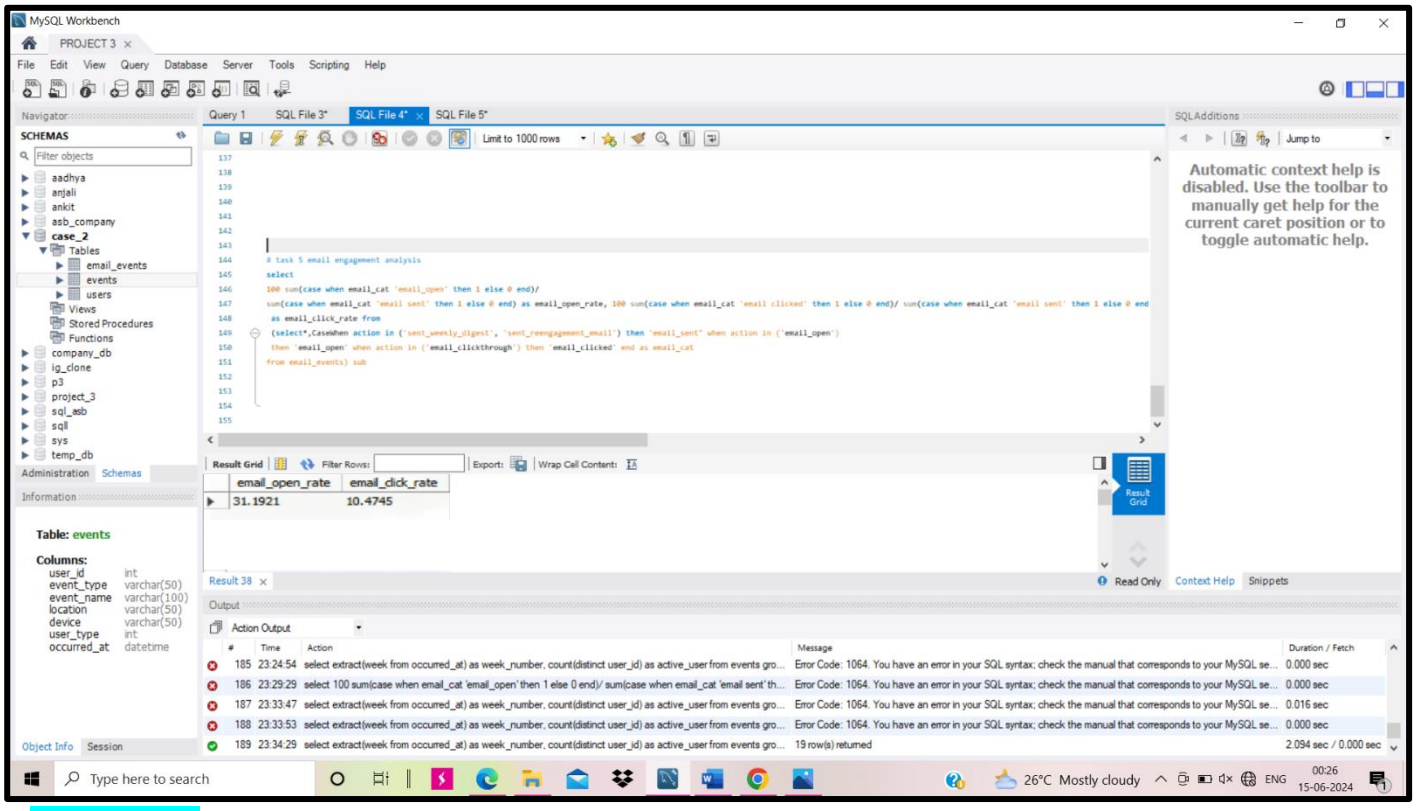
- **Objective: Analyze how users are engaging with the email service**

- **Conclusion/Result:-**

Email open rate	Email click rate
31.1921	10.4745

Out of total sent mail only 35.73 % mail are opened and 15.74% mail are only clicked.

- **Screenshot of code and after running code its output**



Code used:-

```
select
100 sum(case when email_cat 'email_open' then 1 else 0 end)/
sum(case when email_cat 'email sent' then 1 else 0 end) as email_open_rate, 100 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_click_rate from
 (select*,CaseWhen action in ('sent_weekly_digest', 'sent_reengagement_email') then 'email_sent" when action in ('email_open')
 then 'email_open' when action in ('email_clickthrough') then 'email_clicked' end as email_cat
from email_events) sub
```