OPERATION ANALYTICS AND INVESTIGATING METRIC SPIKES

PROJECT DESCRIPTION---

Operational Analytics is a crucial process that involves analysing a company's end-to-end operations. This analysis helps identify areas for improvement within the company and shared with various teams, such as operations, support, and marketing, helping them derive valuable insights from the data they collect.

One of the key aspects of Operational Analytics is investigating metric spikes. This involves understanding and explaining sudden changes in key metrics, such as a dip in daily user engagement or a drop in sales. These questions must be answered daily, making it crucial to understand how to investigate these metric spikes.

APPROACH ---

DATABASE CREATION

Created and inserted the values in the database using DDL & DML SQL queries provided by the Product Manager in the MySQL database using MySQL Workbench.

EXTRACTION OF INSIGHTS

After creating the database required insights are generated from the database table by running SQL queries in MySQL Workbench.

TECH-STACKED USED ---

Used MySQL Community Server which is a free and open source relational

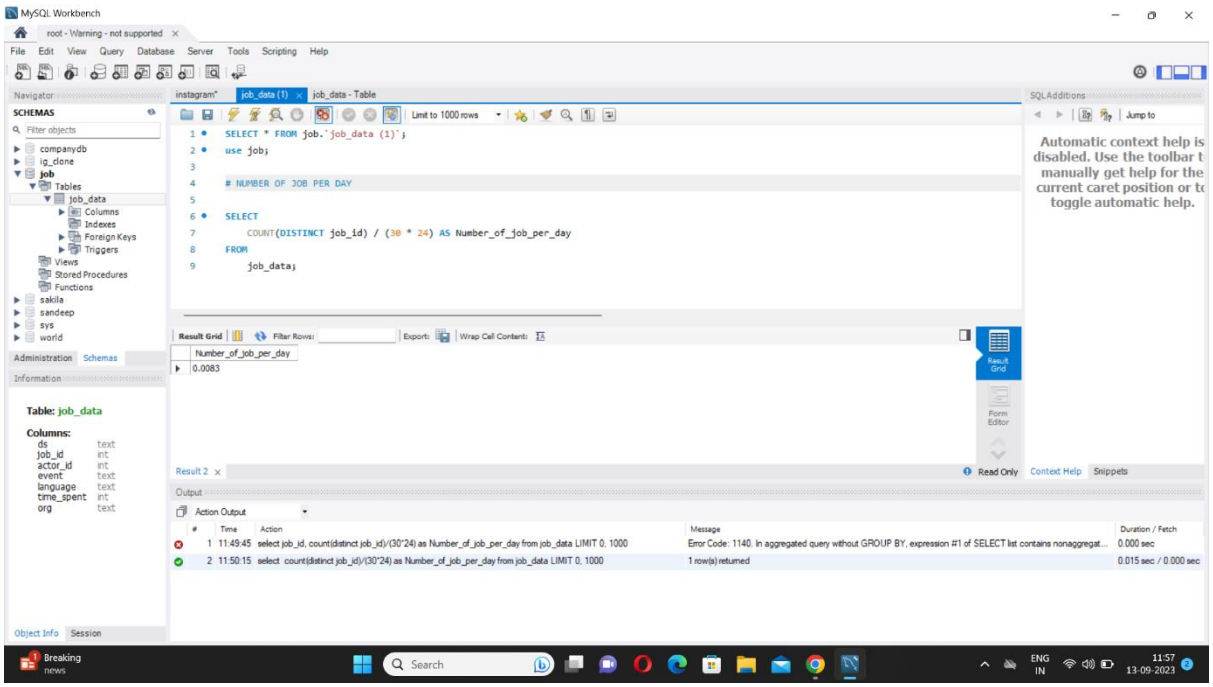
database management system that uses SQL.

INSIGHTS ---

While working in the data I came to know that the number of user were increasing in the engagement event with the increase the number of week , this indicates the growth .

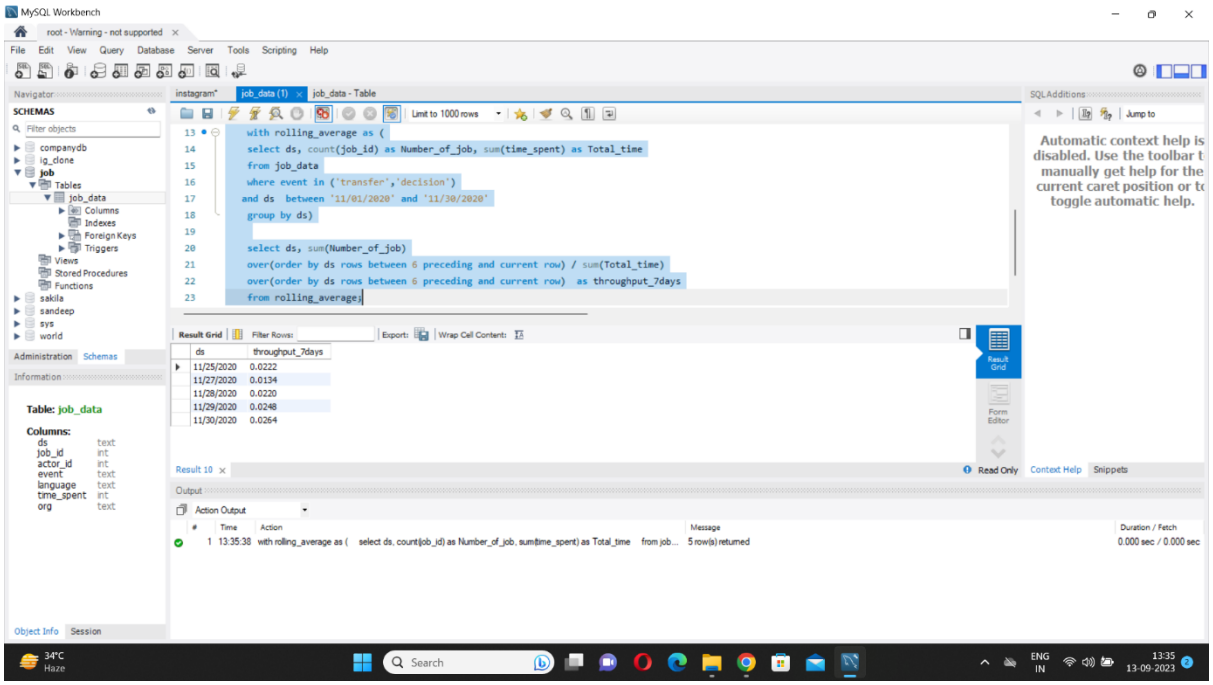
NUMBER OF JOB PER DAY

```
SELECT
    COUNT(DISTINCT job_id) / (30 * 24) AS Number_of_job_per_day
FROM
    job_data;
```



7 DAY ROLLING AVERAGE OF THROUGHPUT

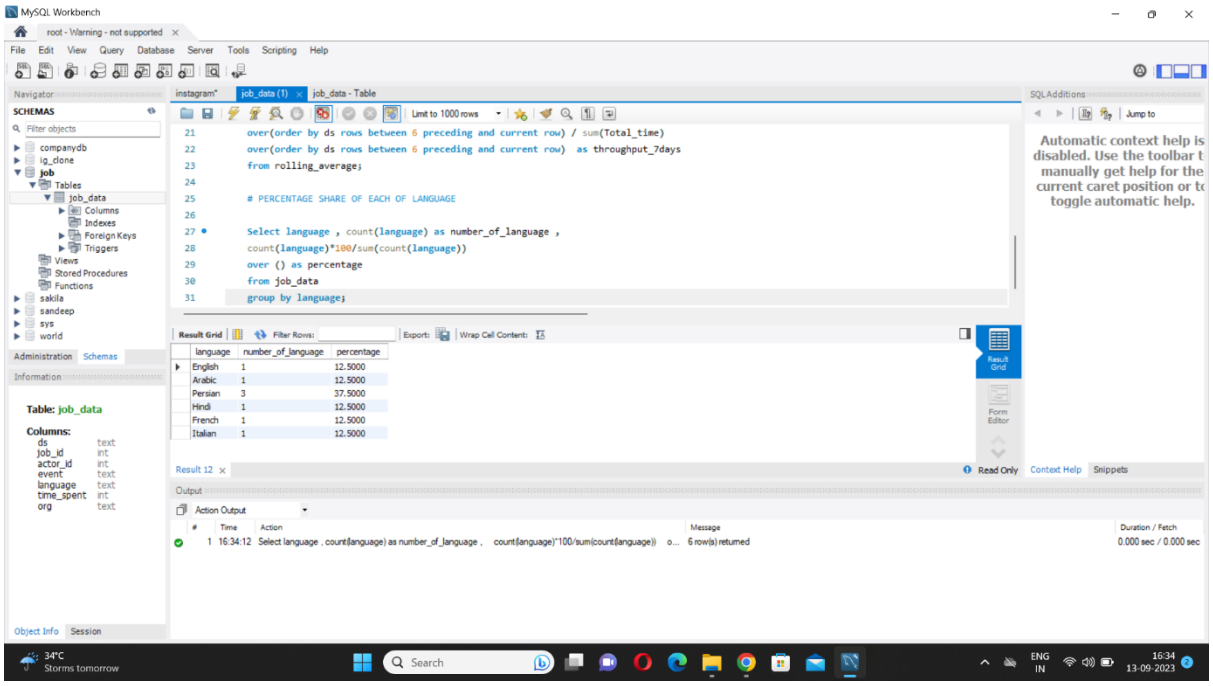
```
with rolling_average as (
select ds, count(job_id) as Number_of_job, sum(time_spent) as Total_time
from job_data
where event in ('transfer','decision')
and ds  between '11/01/2020' and '11/30/2020'
group by ds)

select ds, sum(Number_of_job)
over(order by ds rows between 6 preceding and current row) / sum(Total_time)
over(order by ds rows between 6 preceding and current row)  as throughput_7days
from rolling_average;
```



PERCENTAGE SHARE OF EACH OF LANGUAGE

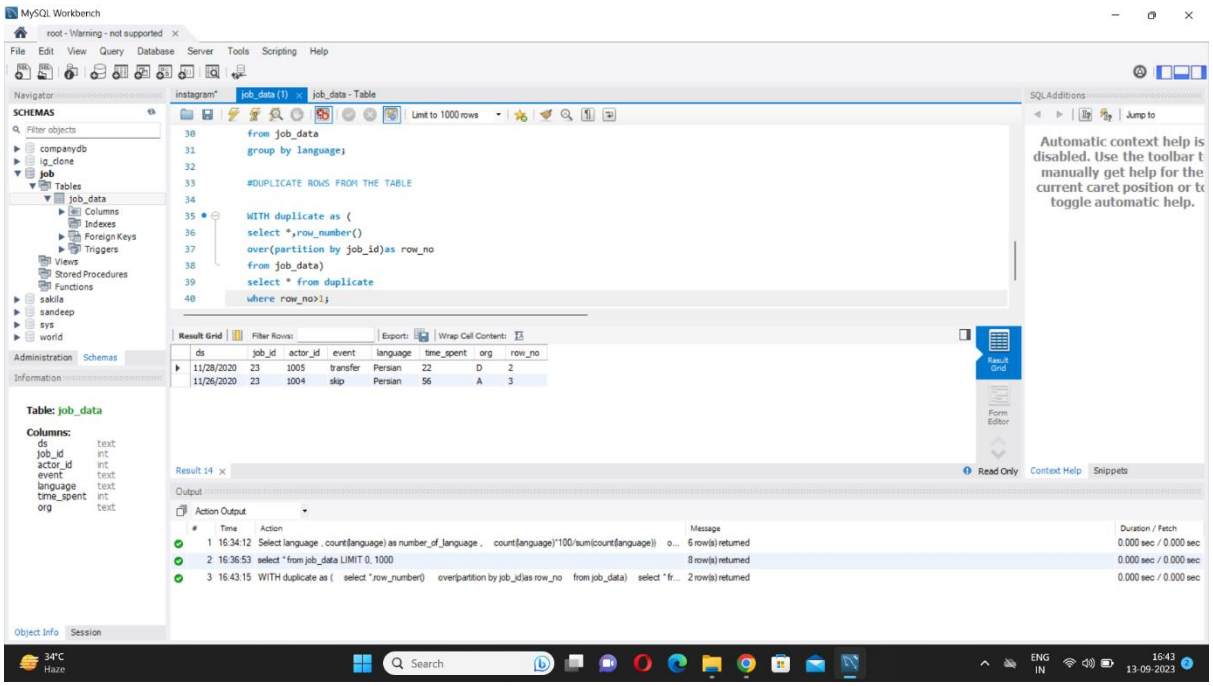
```
Select language , count(language) as number_of_language ,
```

```
count(language)*100/sum(count(language))

over () as percentage

from job_data

group by language;
```



DUPLICATE ROWS FROM THE TABLE

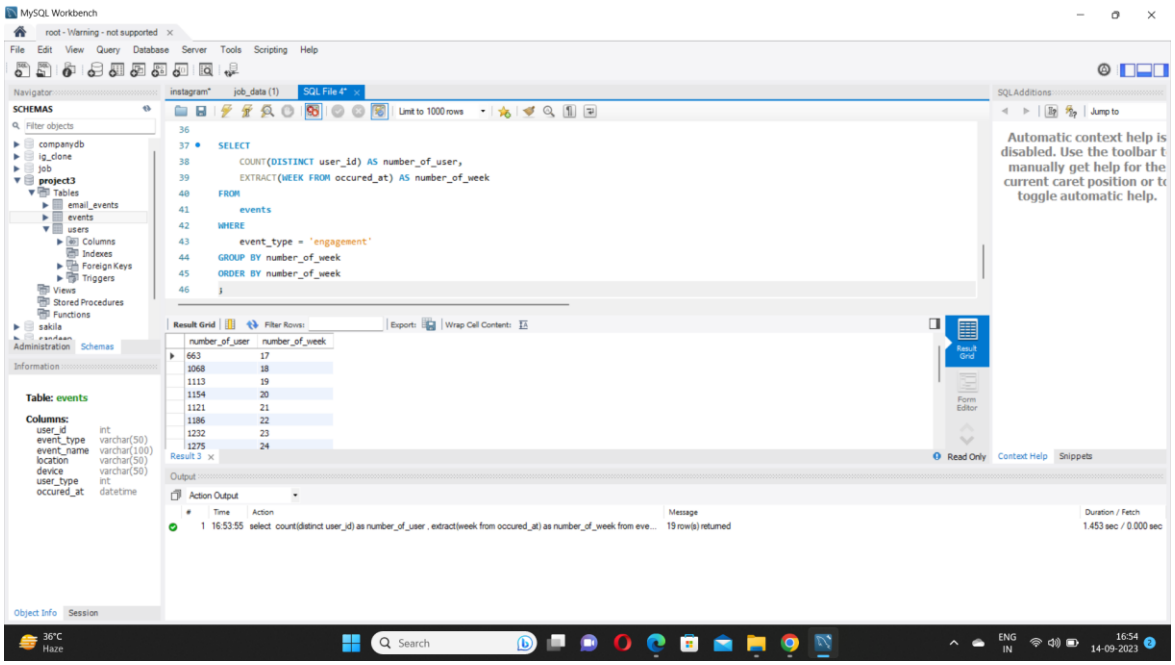
```
WITH duplicate as (

select *,row_number()

over(partition by job_id)as row_no

from job_data)

select * from duplicate

where row_no>1;
```



INVESTIGATING METRIC SPIKE

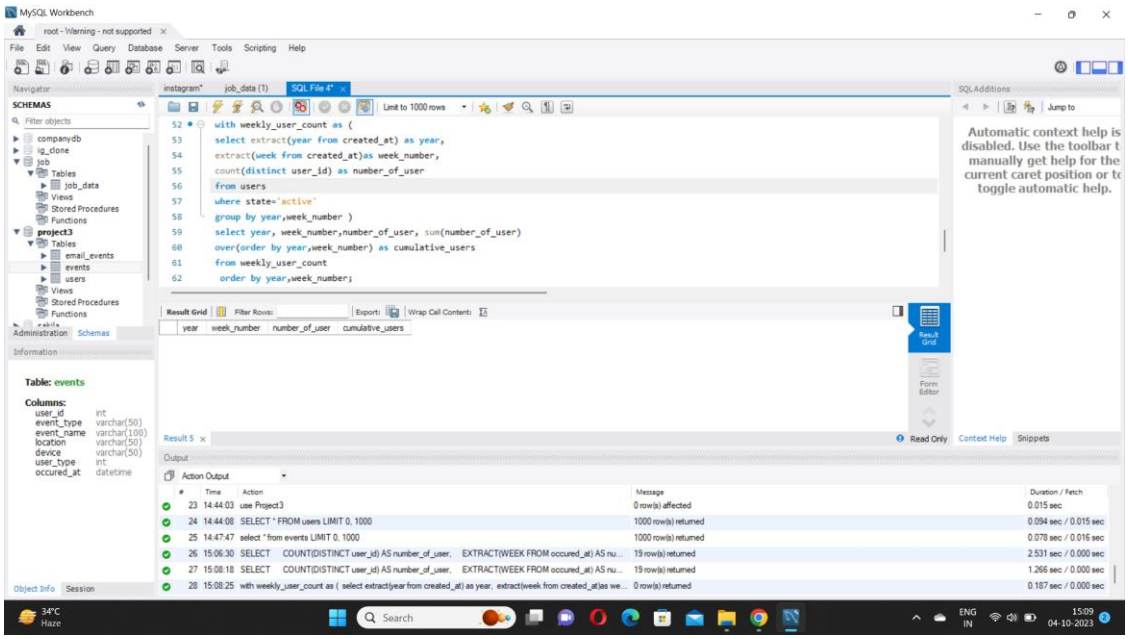
WEEKLY USER ENGAGEMENT

```
SELECT
    COUNT(DISTINCT user_id) AS number_of_user,
    EXTRACT(WEEK FROM occured_at) AS number_of_week
FROM events
WHERE
    event_type = 'engagement'
GROUP BY number_of_week
ORDER BY number_of_week
;
```



USER GROWTH FOR THE PRODUCT

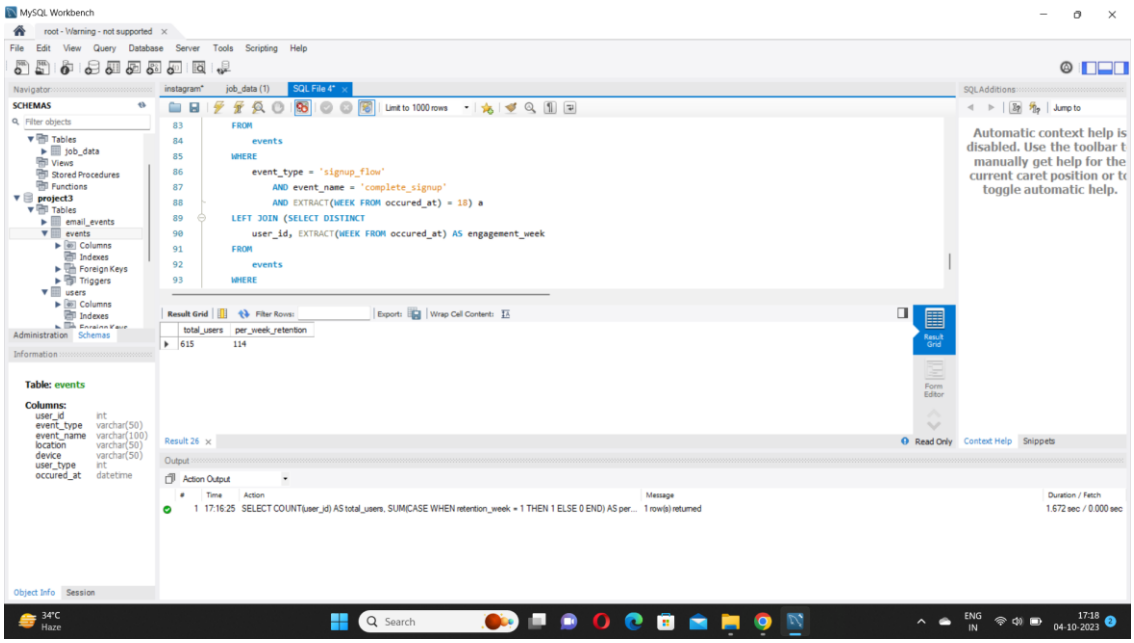
```
with weekly_user_count as (
select extract(year from created_at) as year,
extract(week from created_at)as week_number,
count(distinct user_id) as number_of_user
from users
where state='active'
group by year,week_number )
select year, week_number,number_of_user, sum(number_of_user)
over(order by year,week_number) as cumulative_users
from weekly_user_count
order by year , week_number ;
```



WEEKLY RETENTION ANALYSIS BASED ON SIGNUP

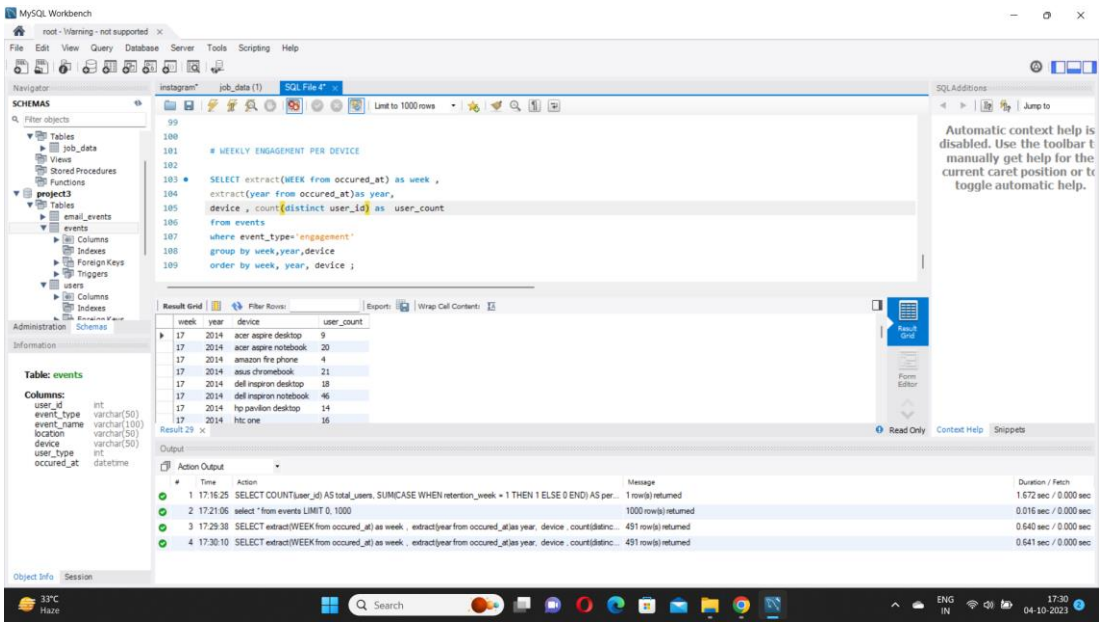
```
SELECT
  COUNT(user_id) AS total_users,
  SUM(CASE
    WHEN retention_week = 1 THEN 1
    ELSE 0
  END) AS per_week_retention
FROM
  (SELECT
    a.user_id,
      a.sign_up_week,
      b.engagement_week,
      b.engagement_week - a.sign_up_week AS retention_week
  FROM
    ((SELECT DISTINCT
    user_id, EXTRACT(WEEK FROM occured_at) AS sign_up_week
```

```
FROM
    events
WHERE
    event_type = 'signup_flow'
        AND event_name = 'complete_signup'
        AND EXTRACT(WEEK FROM occured_at) = 18) 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)
GROUP BY a.user_id , a.sign_up_week , b.engagement_week
ORDER BY a.user_id , a.sign_up_week) subquery
;
```



WEEKLY ENGAGEMENT PER DEVICE

```
SELECT
    EXTRACT(WEEK FROM occured_at) AS week,
    EXTRACT(YEAR FROM occured_at) AS year,
    device,
    COUNT(DISTINCT user_id) AS user_count
FROM
    events
WHERE
    event_type = 'engagement'
GROUP BY week , year , device
ORDER BY week , year , device;
```



EMAIL ENGAGEMENT ANALYSIS

```
SELECT
    SUM(action = 'email_open') / SUM(action = 'sent_weekly_digest') AS email_opening_rate,
```

SUM(action = 'email_clickthrough') / SUM(action = 'sent_weekly_digest') AS email_clickung_rate

FROM

email_events;