SUMIT SHAMLAL CHAURE

Operation Analytics and Investigating Metric Spike

Trainity Project 2 – Advanced SQL

Introduction

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

Description

Operational Analytics is a crucial process that involves analyzing a company's end-to-end operations. This analysis helps identify areas for improvement within the company. As a Data Analyst, you'll work closely 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. As a Data Analyst, you'll need to answer these questions daily, making it crucial to understand how to investigate these metric spikes.

In this project, you'll take on the role of a Lead Data Analyst at a company like Microsoft. You'll be provided with various datasets and tables, and your task will be to derive insights from this data to answer questions posed by different departments within the company. Your goal is to use your 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.

There are two Case Study in these assignment :

1. Job Data Analysis
2. Investigating Metric Spike

Requirements -

1) Project Description :

The aim of the project is to find the user analytics involved in operations of a job activity pertaining to job reviews done and the time spent on job. The investigation metric spike helps us to identify the sudden changes in key metrics like user engagement activities, drop in sales, productivity etc.

2) Approach :

Firstly I created the database and tables from the provided CSV data – for the first case study I manually wrote the query to create and insert data but the case study 2 required to import large data from csv so we made a blank table and then used a special query called 'Inline Load File' to load the data & then change the table to adjust the datatypes where needed. Then using basic query we got the required results for the operations. For the visualization part I made the csv data into excel tables and generated the screenshots of both the query and table for results.For some tables we need to do operations like table alterations too.

The analysis based on query has insights at the bottom of the screengrabs to let the mentor understand the aim of each analysis.

3) Tech-Stack Used :

MySQL – For the main query and results part I have relied on MySQL version 8.1.0.

Excel – The tables were generated from the csv data of the query results of sql and imported in excel.

Word – The report is written in word/docx format using MS Word and then exported to pdf.

Drive – To upload all the essential files attached in the report for reference and for pdf upload.

4) Insights :

The summary for each query is given with the screenshot but to summarize the overall thing I came to the conclusion that i learnt about the business scenario where multiple tables contains related data which when combined together brings in deeper and more meaningful insights which are necessary to let the company grow and also helps the data analysts to generate insights that can help them predict the future activities and engagements.

5) Result :

The project has helped me get the gist of real life examples of database operations from table creation, data insertion management to the complexity involved in connecting various tables to store and get relevant data. The analysis helps to know about the parameter and inputs required to track the activities

. I learnt about joins and select statements more practically with the activity relating to functions which helped me more efficiently do querying operations.

Main Querying Part & Analysis -

SQL Task

<u>Case Study 1: Job Data Analysis</u>

A. Jobs Reviewed Over Time

B. Throughput Analysis

C. Language Share Analysis

D. Duplicate Rows Detection

<u>Case Study 2: Investigating Metric Spike</u>

A. Weekly User Engagement

B. User Growth Analysis

C. Weekly Retention Analysis

D. Weekly Engagement Per Device

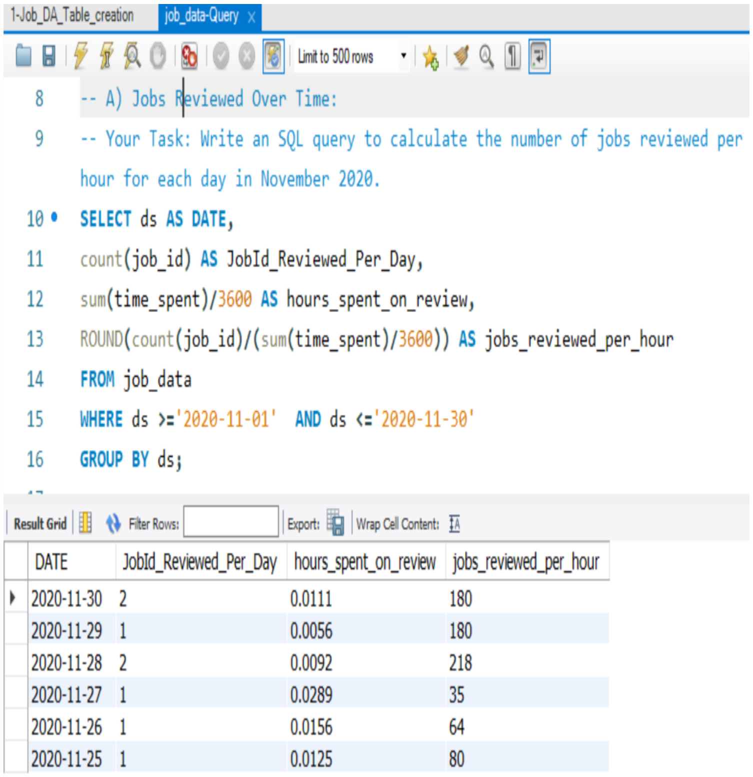
E. Email Engagement Analysis

A) Case Study 1: Job Data Analysis:

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

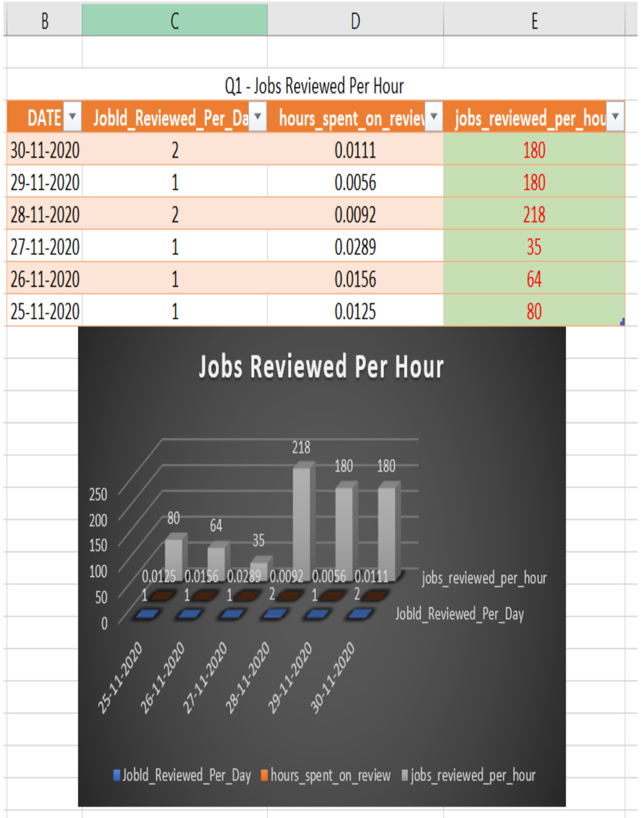
Your Task: Write an SQL query to calculate the number of jobs reviewed per hour for each day in November 2020.

Query



```sql
-- A) Jobs Reviewed Over Time:
-- Your Task: Write an SQL query to calculate the number of jobs reviewed per hour for each day in November 2020.
SELECT ds AS DATE,
count(job_id) AS JobId_Reviewed_Per_Day,
sum(time_spent)/3600 AS hours_spent_on_review,
ROUND(count(job_id)/(sum(time_spent)/3600)) AS jobs_reviewed_per_hour
FROM job_data
WHERE ds >='2020-11-01'  AND ds <='2020-11-30'
GROUP BY ds;
```

DATE	JobId_Reviewed_Per_Day	hours_spent_on_review	jobs_reviewed_per_hour
2020-11-30	2	0.0111	180
2020-11-29	1	0.0056	180
2020-11-28	2	0.0092	218
2020-11-27	1	0.0289	35
2020-11-26	1	0.0156	64
2020-11-25	1	0.0125	80



Q1 - Jobs Reviewed Per Hour

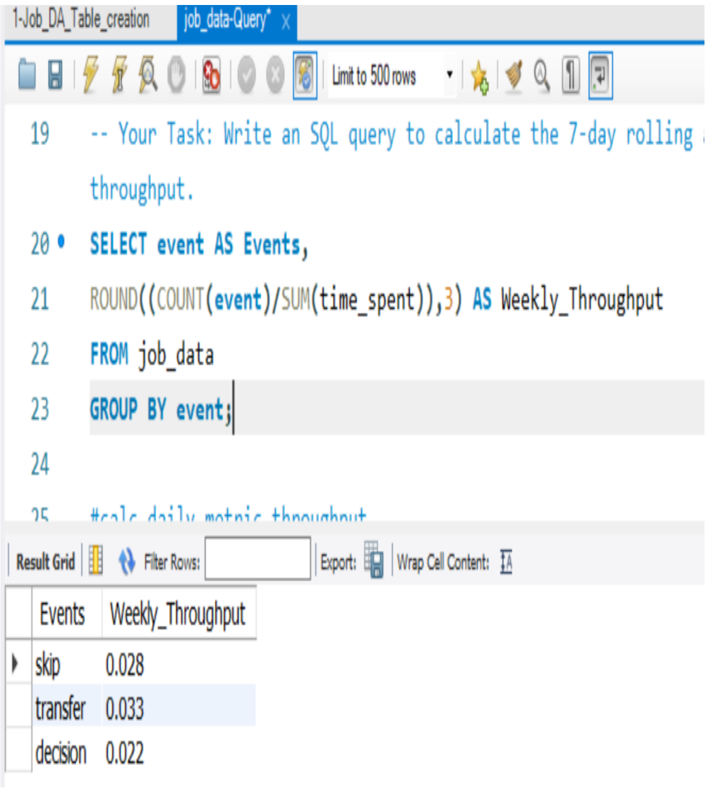
DATE	JobId_Reviewed_Per_Da	hours_spent_on_reviev	jobs_reviewed_per_hou
30-11-2020	2	0.0111	180
29-11-2020	1	0.0056	180
28-11-2020	2	0.0092	218
27-11-2020	1	0.0289	35
26-11-2020	1	0.0156	64
25-11-2020	1	0.0125	80

Insights : From the above query we have come to know that if we take the hours spent every hour on a job review then do the sum of time spent on each job for the day we get the result of our question. The bar chart shows us the distribution of tasks per day and the number of task done per hour according to the time taken for each review (*The highlighted column is our desired output*).

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

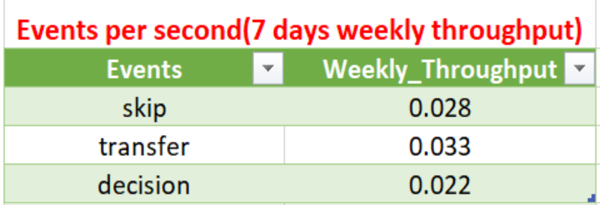
Your Task: Write an SQL query to calculate the 7-day rolling average of throughput. Additionally, explain whether you prefer using the daily metric or the 7-day rolling average for throughput, and why.

Query



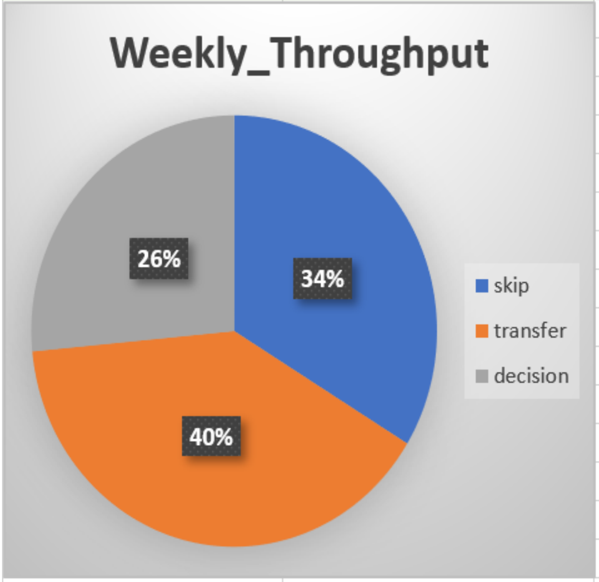
```
-- Your Task: Write an SQL query to calculate the 7-day rolling
throughput.
SELECT event AS Events,
ROUND((COUNT(event)/SUM(time_spent)),3) AS Weekly_Throughput
FROM job_data
GROUP BY event;

#calc daily metric throughput
```

Events	Weekly_Throughput
skip	0.028
transfer	0.033
decision	0.022

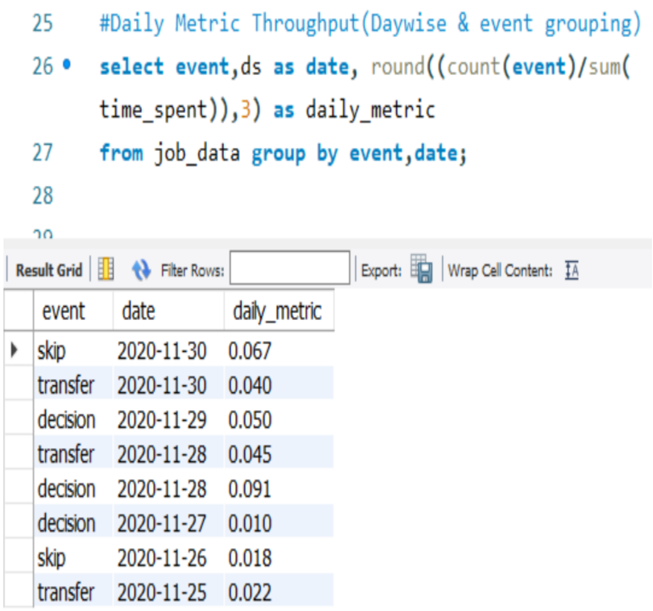


Events per second(7 days weekly throughput)

Events	Weekly_Throughput
skip	0.028
transfer	0.033
decision	0.022

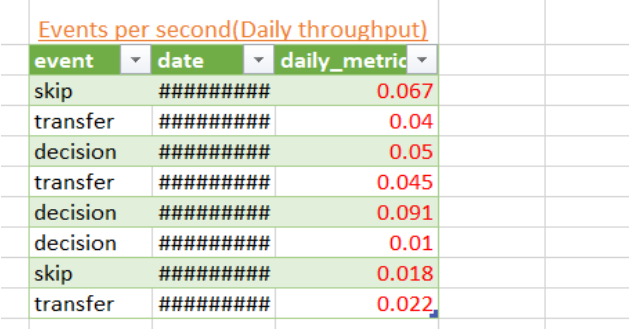


***Img -** Weekly Throughput shows a smooth distribution of events done for the week and distribution is more readable. Weekly metric helps us to understand which events holds account for the most work/reviews.*



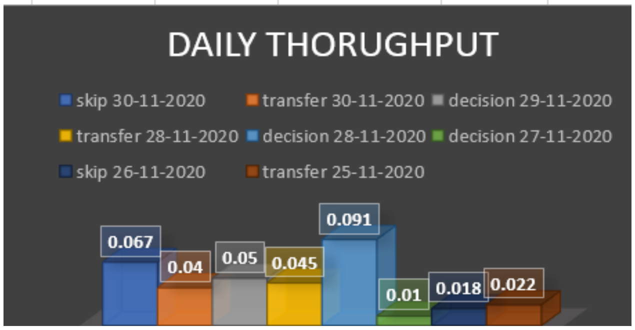
```
#Daily Metric Throughput(Daywise & event grouping)
select event,ds as date, round((count(event)/sum(
time_spent)),3) as daily_metric
from job_data group by event,date;
```

event	date	daily_metric
skip	2020-11-30	0.067
transfer	2020-11-30	0.040
decision	2020-11-29	0.050
transfer	2020-11-28	0.045
decision	2020-11-28	0.091
decision	2020-11-27	0.010
skip	2020-11-26	0.018
transfer	2020-11-25	0.022



Events per second(Daily throughput)

event	date	daily_metric
skip	##########	0.067
transfer	##########	0.04
decision	##########	0.05
transfer	##########	0.045
decision	##########	0.091
decision	##########	0.01
skip	##########	0.018
transfer	##########	0.022



***Img -** The daily throughput based on the events has distributed the work over daily basis which is not that readable for the insights if we look at the event wise distributions too.*

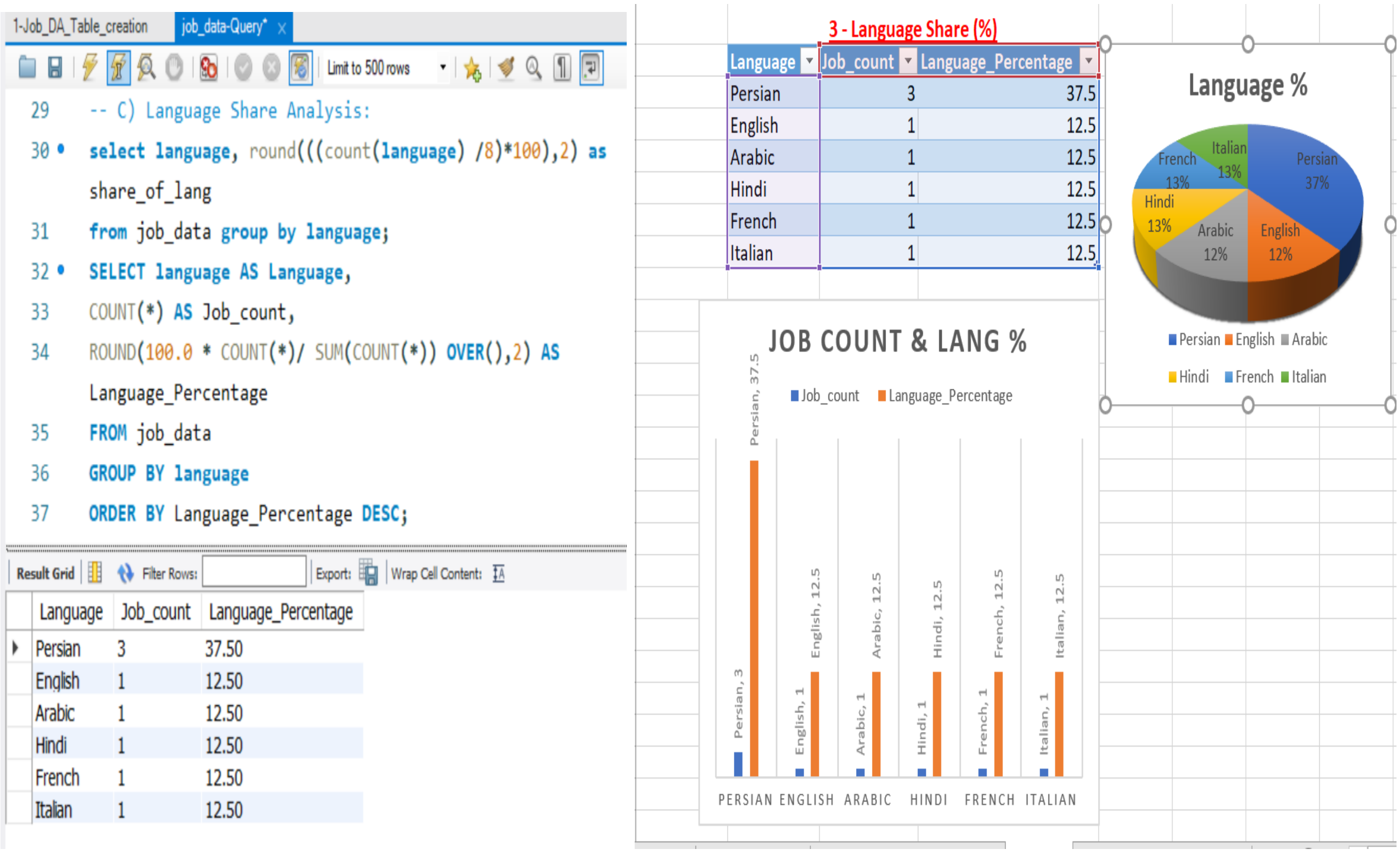
Insights : From the above charts and values we can infer the weekly throughput of the various events based on a weekly metric over events and daily metric over the events and date distribution.

Throughput is the number of events happening per second. For calculating the 7-day rolling average, we used the number of events per second for each day & divided it by the week to get the average needed.From the insights I have inferred that it is better to take a 7 day rolling average over the daily metric as the daily data shows fluctuations and does not give estimated data of events average but the weekly throughput helps us to understand the events handled on seconds basis for the week much more efficiently.

3.Language Share Analysis : Calculate the percentage share of each language in the last 30 days .

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

Query



1-Job_DA_Table_creation | job_data-Query*

```
-- C) Language Share Analysis:
select language, round(((count(language) /8)*100),2) as
share_of_lang
from job_data group by language;
SELECT language AS Language,
COUNT(*) AS Job_count,
ROUND(100.0 * COUNT(*)/ SUM(COUNT(*)) OVER(),2) AS
Language_Percentage
FROM job_data
GROUP BY language
ORDER BY Language_Percentage DESC;
```

Language	Job_count	Language_Percentage
Persian	3	37.50
English	1	12.50
Arabic	1	12.50
Hindi	1	12.50
French	1	12.50
Italian	1	12.50

3 - Language Share (%)

Language	Job_count	Language_Percentage
Persian	3	37.5
English	1	12.5
Arabic	1	12.5
Hindi	1	12.5
French	1	12.5
Italian	1	12.5

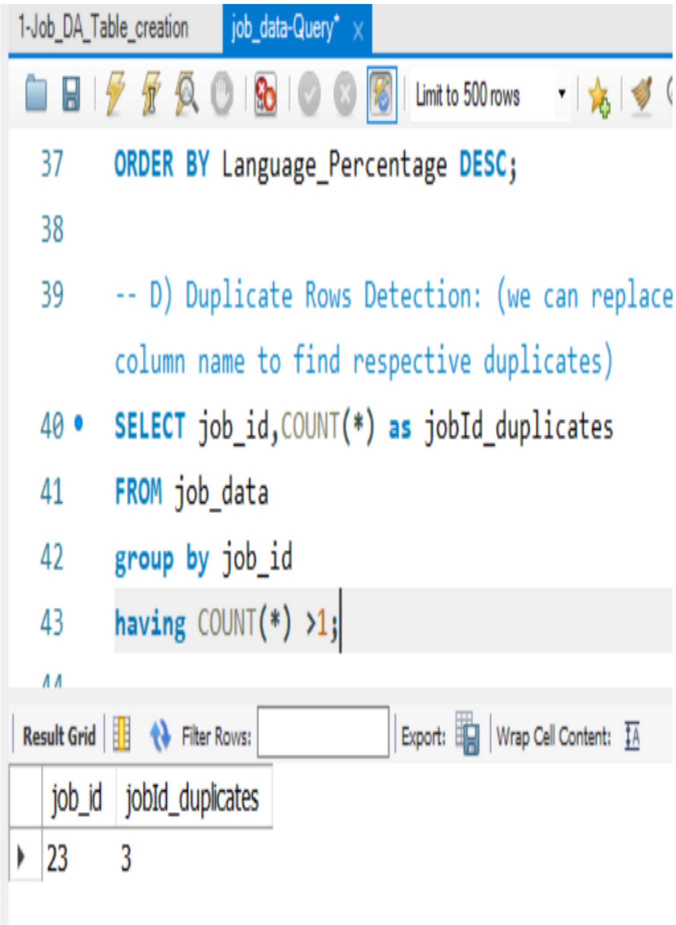
Img - Language Percentage Shares

Insight : *The above insight shows us that* ***Persian*** *language has the highest percentage share of all the 8 languages in our job id followed by all the other language having 1 counts from total count.*

4.Duplicate Rows Detection : Identify duplicate rows in the data.

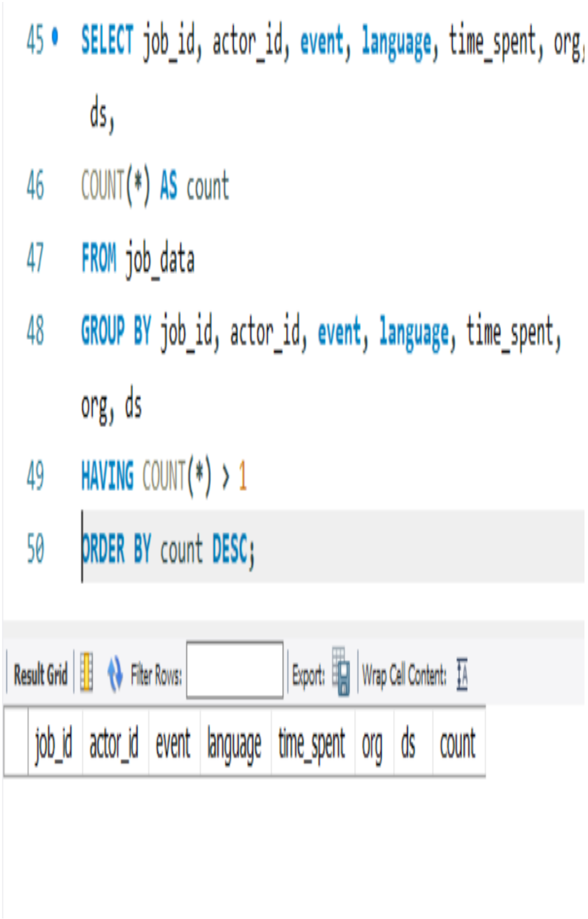
Your Task: Write an SQL query to display duplicate rows from the job_data table.

Query



```
ORDER BY Language_Percentage DESC;

-- D) Duplicate Rows Detection: (we can replace column name to find respective duplicates)
SELECT job_id,COUNT(*) as jobId_duplicates
FROM job_data
group by job_id
having COUNT(*) >1;
```

job_id	jobId_duplicates
23	3



```
SELECT job_id, actor_id, event, language, time_spent, org,
ds,
COUNT(*) AS count
FROM job_data
GROUP BY job_id, actor_id, event, language, time_spent,
org, ds
HAVING COUNT(*) > 1
ORDER BY count DESC;
```

job_id	actor_id	event	language	time_spent	org	ds	count

Img - *1st image considers job_id column while 2nd image is considering every column*

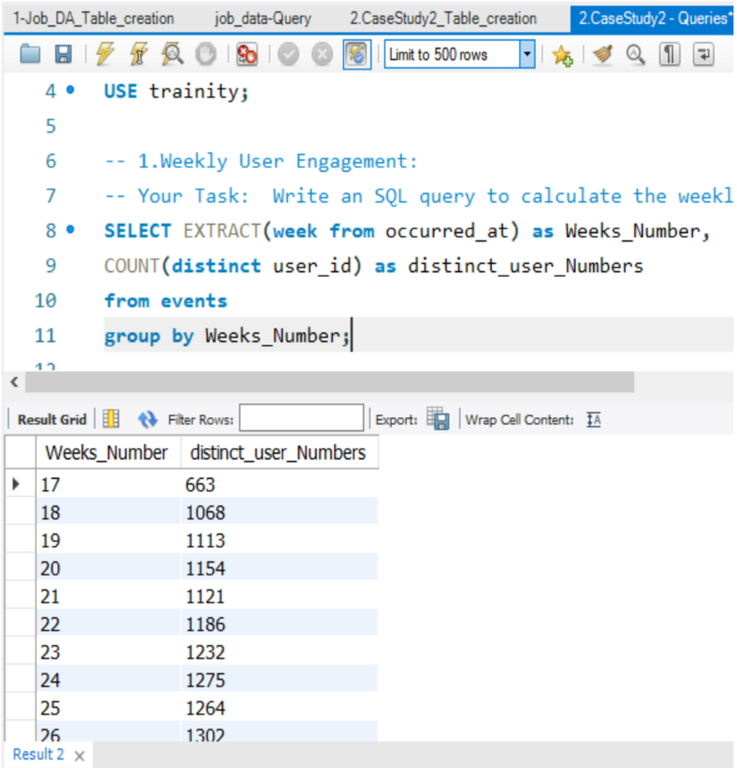
Insights : *If we want to find the duplicate rows we can do a count and use the column name on which we want to find the duplicates in above query if we replace the job_id with say event or with actor_id it will query for any duplicates present in that column as we have not been given any specific primary key or column we can check the different duplicates as needed.In 2nd query we tried checking for whole* *columns if they are repeated so we don't get any results as any specific row does not contain a completely duplicate row.*

B) Case Study 2: Investigating Metric Spike:

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

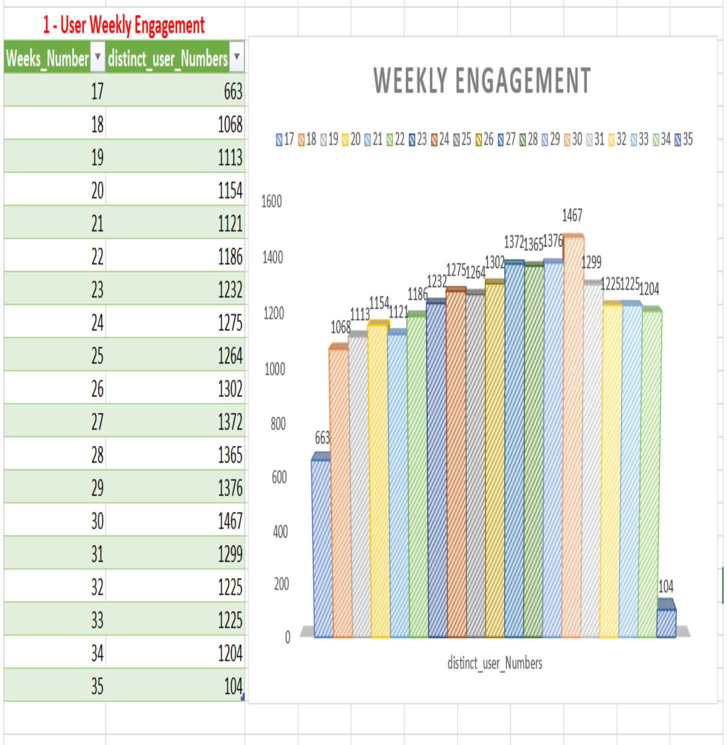
Your Task: Write an SQL query to calculate the weekly user engagement.

Query



```
USE trainity;

-- 1.Weekly User Engagement:
-- Your Task:  Write an SQL query to calculate the weekl
SELECT EXTRACT(week from occurred_at) as Weeks_Number,
COUNT(distinct user_id) as distinct_user_Numbers
from events
group by Weeks_Number;
```

Weeks_Number	distinct_user_Numbers
17	663
18	1068
19	1113
20	1154
21	1121
22	1186
23	1232
24	1275
25	1264
26	1302



1 - User Weekly Engagement

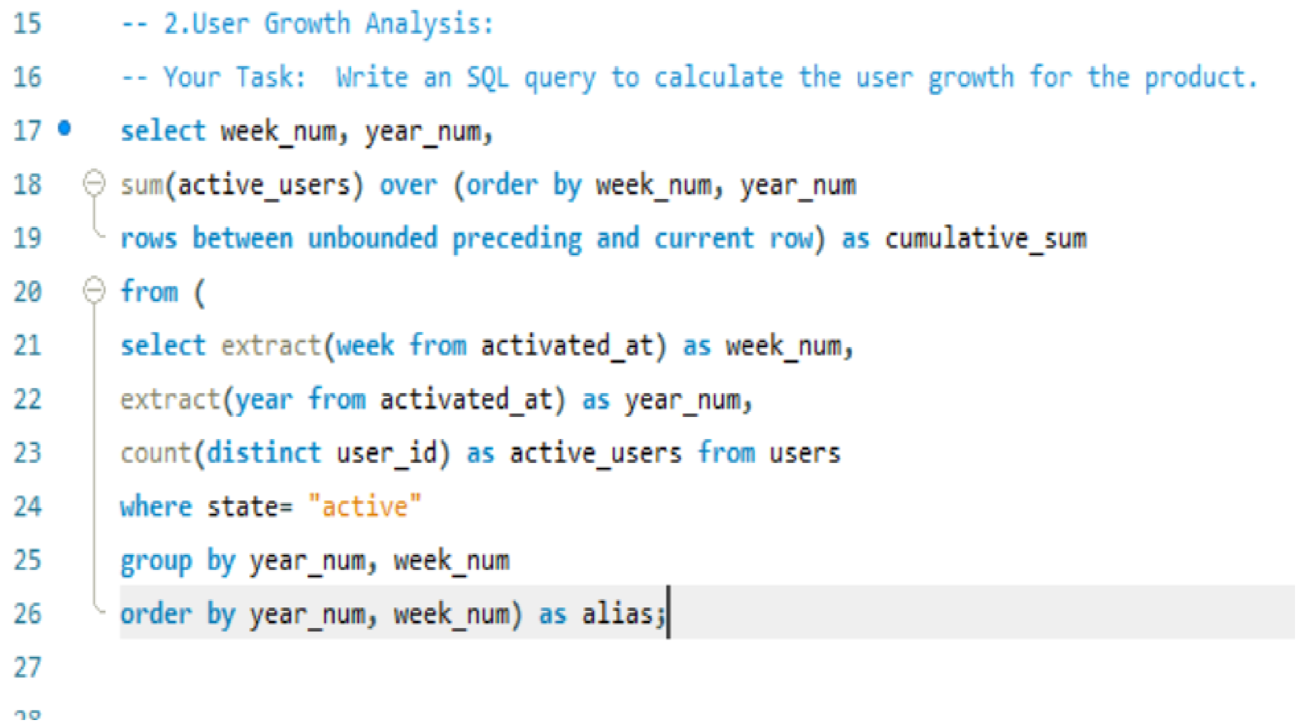
Weeks_Number	distinct_user_Numbers
17	663
18	1068
19	1113
20	1154
21	1121
22	1186
23	1232
24	1275
25	1264
26	1302
27	1372
28	1365
29	1376
30	1467
31	1299
32	1225
33	1225
34	1204
35	104

Insights : *From the above query we get the user engagement that is unique user visit according to the week numbers.This demograph helps us to gather information about the events density and user uniqueness at the events.*

2.User Growth Analysis: Analyze the growth of users over time for a product.

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

Query



```
-- 2.User Growth Analysis:
-- Your Task:  Write an SQL query to calculate the user growth for the product.
select week_num, year_num,
sum(active_users) over (order by week_num, year_num
rows between unbounded preceding and current row) as cumulative_sum
from (
select extract(week from activated_at) as week_num,
extract(year from activated_at) as year_num,
count(distinct user_id) as active_users from users
where state= "active"
group by year_num, week_num
order by year_num, week_num) as alias;
```

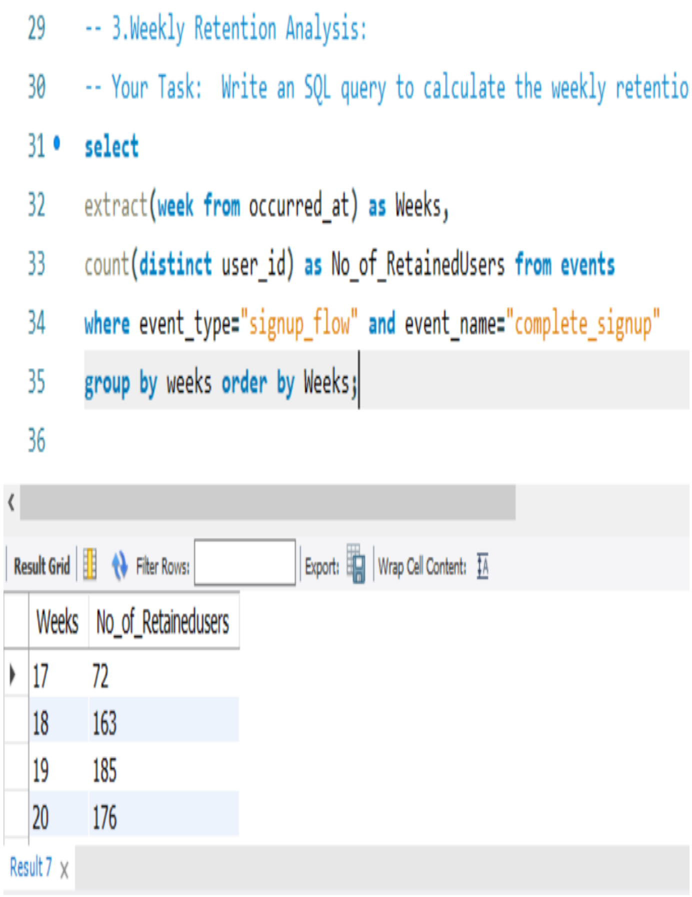
Insights : *From the above query we get to find out about the number of users on the platform and the growing numbers according to week and year wise when we see at the active state of users column.*

This data would help us determine strategy for future growth planning and ad or other revenue generation activities.

3.Weekly Retention Analysis: Analyze the retention of users on a weekly basis after signing up for a product.

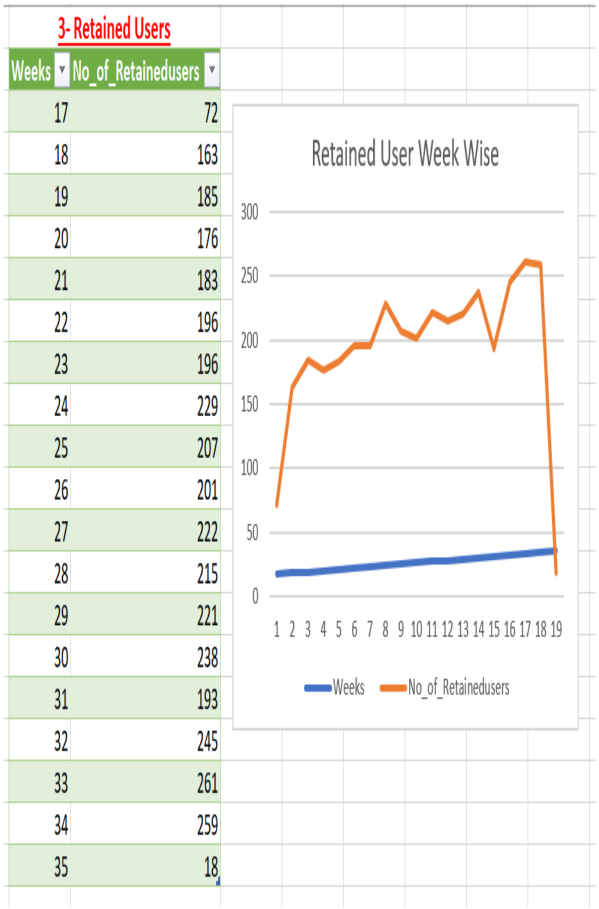
Your Task: Write an SQL query to calculate the weekly retention of users based on their sign-up cohort..

Query



```
-- 3.Weekly Retention Analysis:
-- Your Task: Write an SQL query to calculate the weekly retentio
select
extract(week from occurred_at) as Weeks,
count(distinct user_id) as No_of_RetainedUsers from events
where event_type="signup_flow" and event_name="complete_signup"
group by weeks order by Weeks;
```

Weeks	No_of_Retainedusers
17	72
18	163
19	185
20	176



3- Retained Users

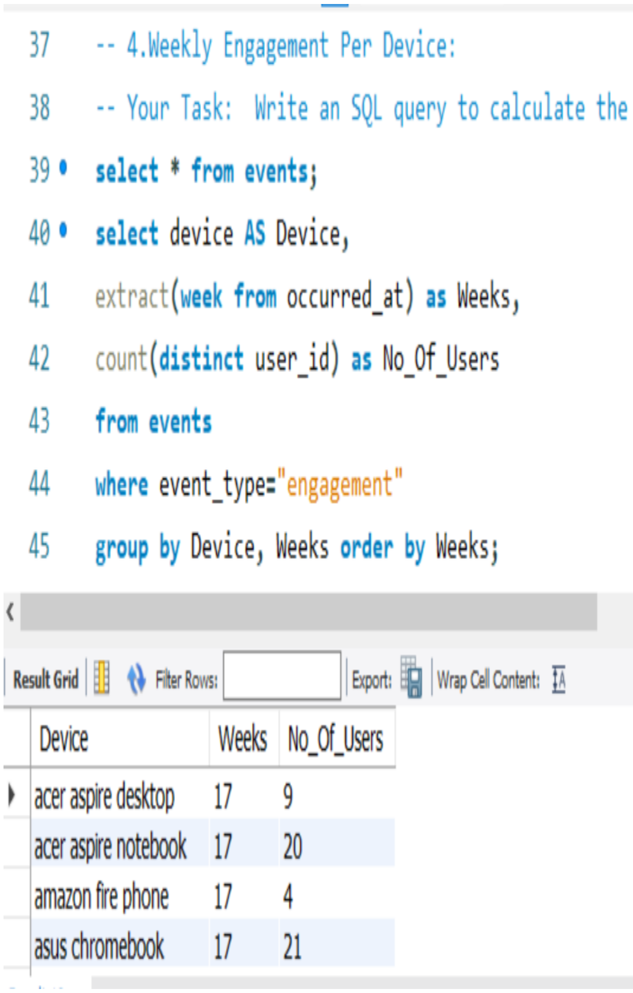
Weeks	No_of_Retainedusers
17	72
18	163
19	185
20	176
21	183
22	196
23	196
24	229
25	207
26	201
27	222
28	215
29	221
30	238
31	193
32	245
33	261
34	259
35	18

Insight ; *From above query we have found the number of users who have signed up into our cohort program by completing the signup flow and complete the signup process.*

4.Weekly Engagement Per Device: Measure the activeness of users on a weekly basis per device.

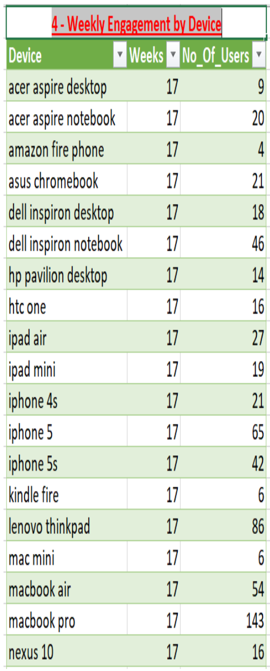
Your Task: Write an SQL query to calculate the weekly engagement per device.

Query



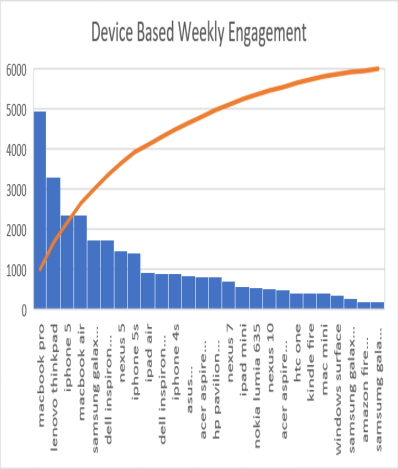
```
-- 4.Weekly Engagement Per Device:
-- Your Task:  Write an SQL query to calculate the
select * from events;
select device AS Device,
extract(week from occurred_at) as Weeks,
count(distinct user_id) as No_Of_Users
from events
where event_type="engagement"
group by Device, Weeks order by Weeks;
```

Device	Weeks	No_Of_Users
acer aspire desktop	17	9
acer aspire notebook	17	20
amazon fire phone	17	4
asus chromebook	17	21



4 - Weekly Engagement by Device

Device	Weeks	No_Of_Users
acer aspire desktop	17	9
acer aspire notebook	17	20
amazon fire phone	17	4
asus chromebook	17	21
dell inspiron desktop	17	18
dell inspiron notebook	17	46
hp pavilion desktop	17	14
htc one	17	16
ipad air	17	27
ipad mini	17	19
iphone 4s	17	21
iphone 5	17	65
iphone 5s	17	42
kindle fire	17	6
lenovo thinkpad	17	86
mac mini	17	6
macbook air	17	54
macbook pro	17	143
nexus 10	17	16



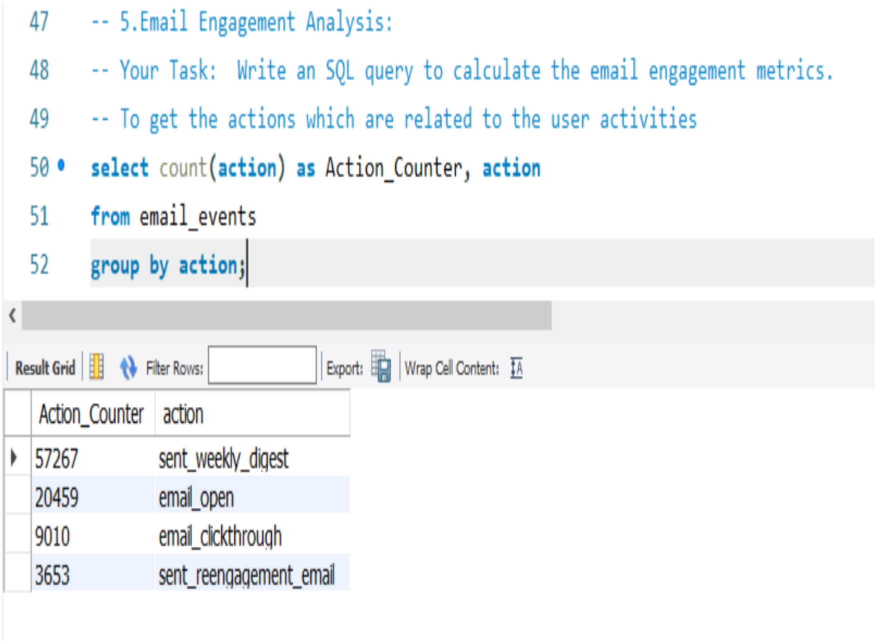
Insights: *The above insight shows us the weekly demographic of the various devices used to access the events page and the number of user using the device in a certain weeks period.These data helps us to optimize our site/page/app to suite to majority of the devices/OS and thereby increasing our reach.*

The data shows that majority of the users prefer to use ***macbook pro.***

5.Email Engagement Analysis: Analyze how users are engaging with the email service

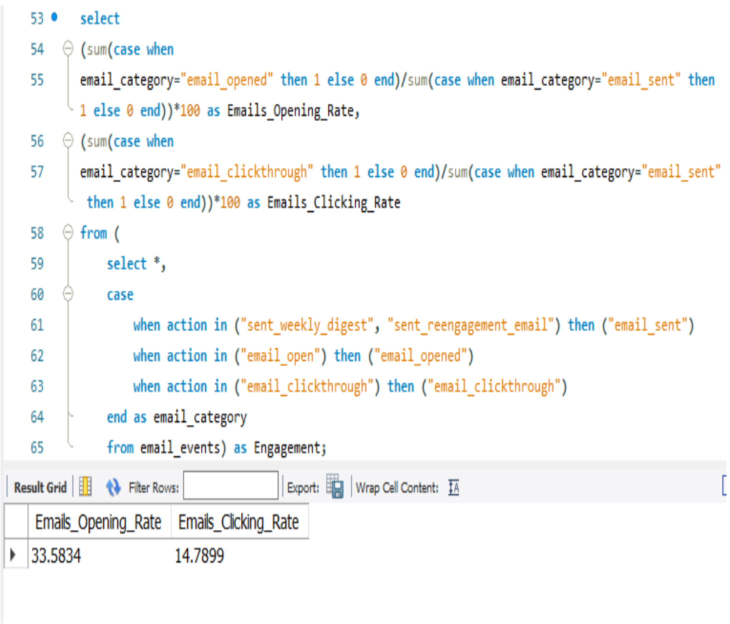
Your Task: Write an SQL query to calculate the email engagement metrics.

Query



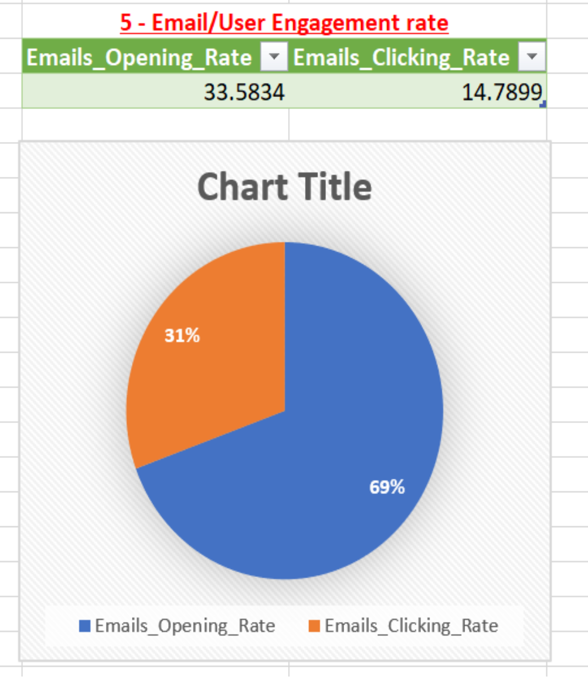
```
-- 5.Email Engagement Analysis:
-- Your Task:  Write an SQL query to calculate the email engagement metrics.
-- To get the actions which are related to the user activities
select count(action) as Action_Counter, action
from email_events
group by action;
```

Action_Counter	action
57267	sent_weekly_digest
20459	email_open
9010	email_clickthrough
3653	sent_reengagement_email



```
select
(sum(case when
email_category="email_opened" then 1 else 0 end)/sum(case when email_category="email_sent" then
1 else 0 end))*100 as Emails_Opening_Rate,
(sum(case when
email_category="email_clickthrough" then 1 else 0 end)/sum(case when email_category="email_sent"
 then 1 else 0 end))*100 as Emails_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_clickthrough")
    end as email_category
    from email_events) as Engagement;
```

Emails_Opening_Rate	Emails_Clicking_Rate
33.5834	14.7899



5 - Email/User Engagement rate

Emails_Opening_Rate	Emails_Clicking_Rate
33.5834	14.7899

Q1 - *Query to check various user activities involved.* ***Q2 -*** *Shows the actual engagement activity based on various activities we got from 1st querry.*

***Insights** : From the above query we can say that from the emails sent to user the email opening rate is approximately 33.6% and the user engagement with the emails is around 15%. This shows that half of the user who opens the mail dont engage into an action or engagement activity and the overall opening rate shows that the marketing team and creative department needs to give focus into bringing the customers on board and making them interact with our platform more.*

THANK YOU