CASE STUDY 1: JOB DATA ANALYSIS

JOB DATA

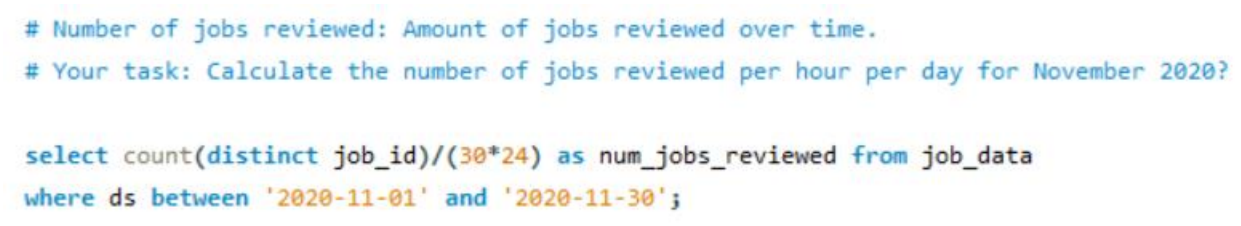
Software used: MySQL Workbench 8.0 CE

Task - 1

Jobs Reviewed Over Time:

- Objective: 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.

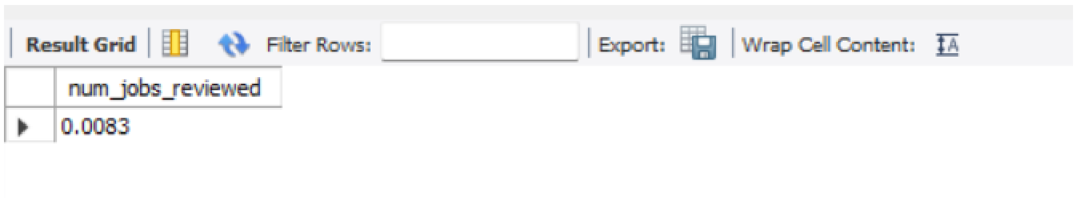
Code –



```
# Number of jobs reviewed: Amount of jobs reviewed over time.
# Your task: Calculate the number of jobs reviewed per hour per day for November 2020?

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

Output –



```
num_jobs_reviewed
0.0083
```

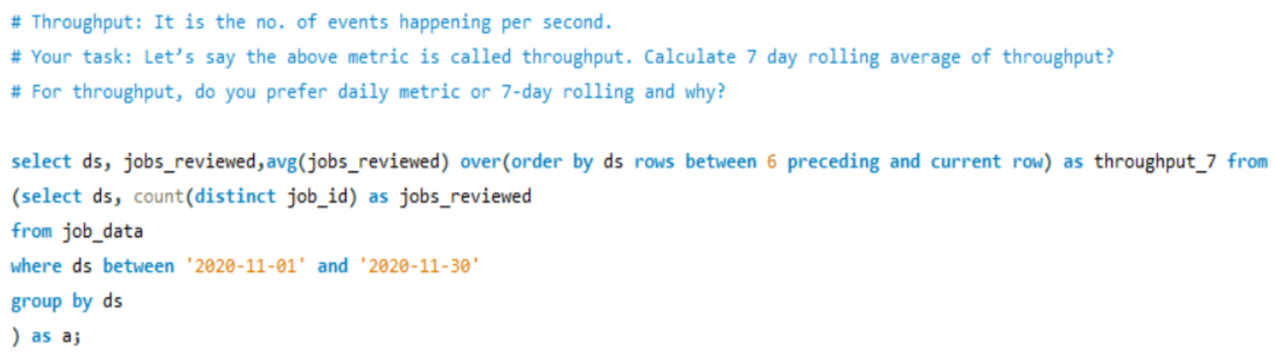
Task - 2

Throughput Analysis:

- Objective: 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.

Throughput: It is the no. of events happening per second

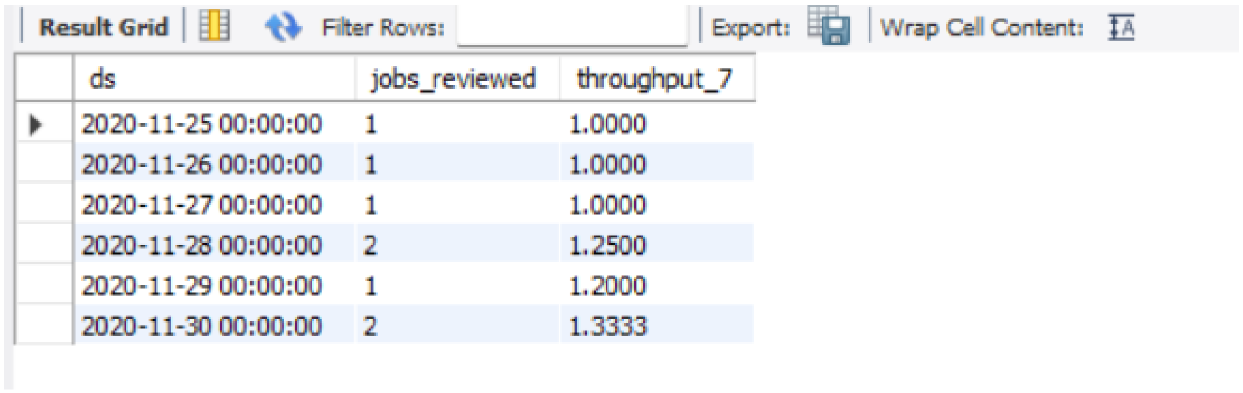
Code –



```
# Throughput: It is the no. of events happening per second.
# Your task: Let’s say the above metric is called throughput. Calculate 7 day rolling average of throughput?
# For throughput, do you prefer daily metric or 7-day rolling and why?

select ds, jobs_reviewed,avg(jobs_reviewed) over(order by ds rows between 6 preceding and current row) as throughput_7 from
(select ds, count(distinct job_id) as jobs_reviewed
from job_data
where ds between '2020-11-01' and '2020-11-30'
group by ds
) as a;
```

Output –



ds	jobs_reviewed	throughput_7
2020-11-25 00:00:00	1	1.0000
2020-11-26 00:00:00	1	1.0000
2020-11-27 00:00:00	1	1.0000
2020-11-28 00:00:00	2	1.2500
2020-11-29 00:00:00	1	1.2000
2020-11-30 00:00:00	2	1.3333

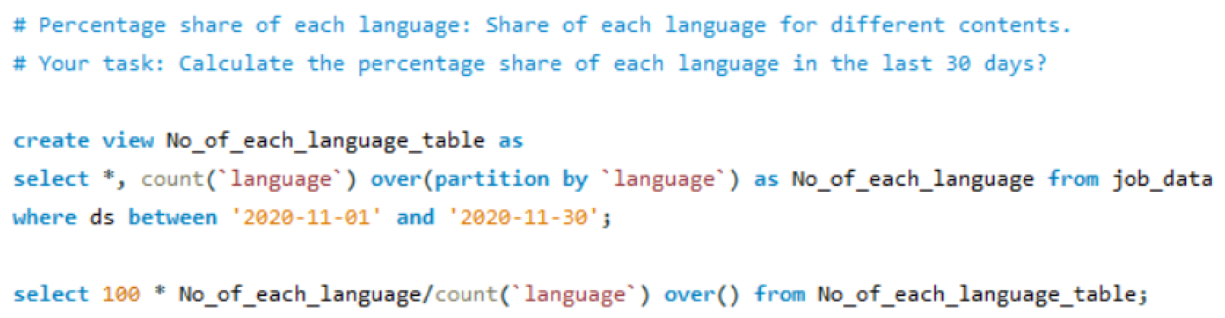
Task – 3

Language Share Analysis:

- Objective: 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.

Percentage share of each language: Share of each language for different contents.

Code –

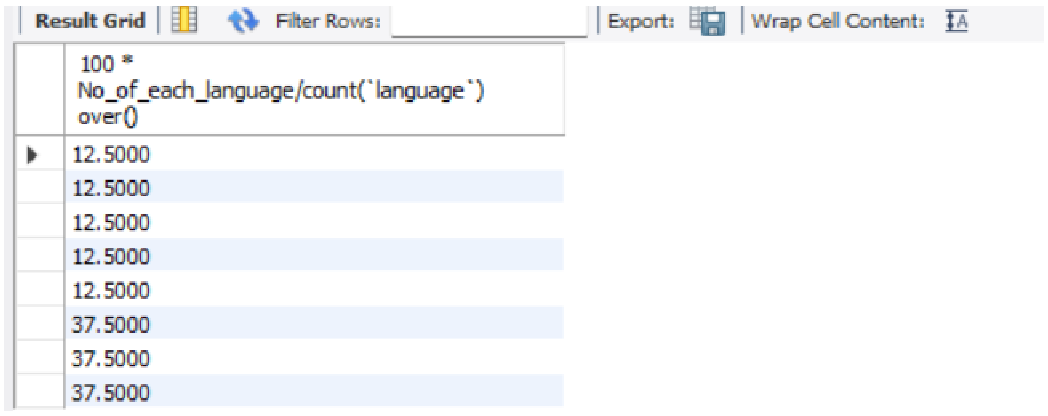


```
# Percentage share of each language: Share of each language for different contents.
# Your task: Calculate the percentage share of each language in the last 30 days?

create view No_of_each_language_table as
select *, count(`language`) over(partition by `language`) as No_of_each_language from job_data
where ds between '2020-11-01' and '2020-11-30';

select 100 * No_of_each_language/count(`language`) over() from No_of_each_language_table;
```

Output –



```
100 *
No_of_each_language/count(`language`)
over()
12.5000
12.5000
12.5000
12.5000
12.5000
37.5000
37.5000
37.5000
```

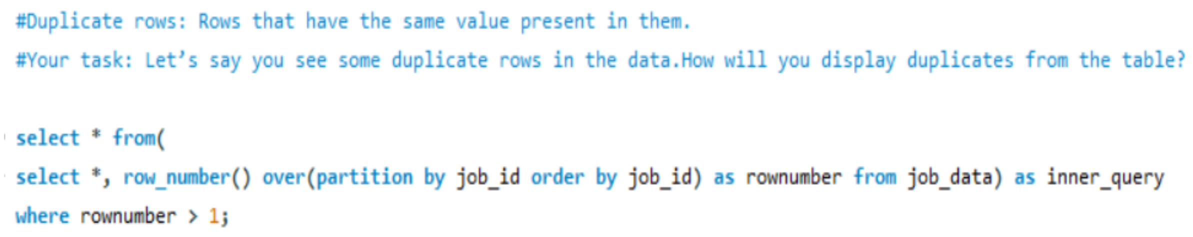
Task – 4

Duplicate Rows Detection:

- Objective: Identify duplicate rows in the data.
- Your Task: Write an SQL query to display duplicate rows from the job_data table.

Duplicate rows: Rows that have the same value present in them.

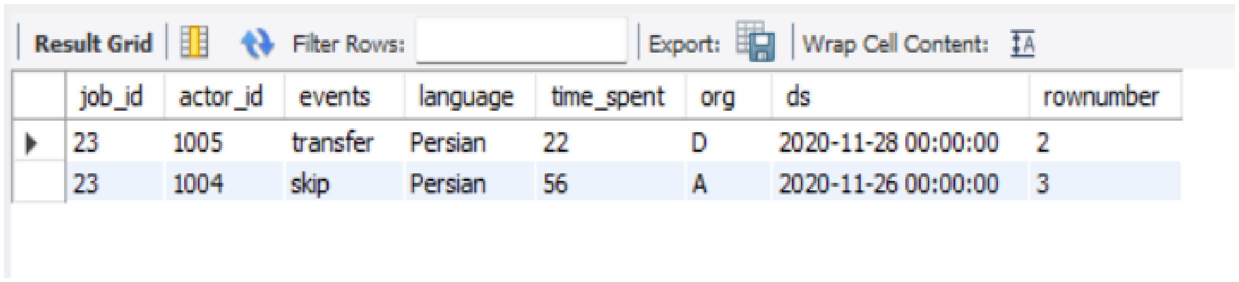
Code –



```
#Duplicate rows: Rows that have the same value present in them.
#Your task: Let's say you see some duplicate rows in the data.How will you display duplicates from the table?

select * from(
select *, row_number() over(partition by job_id order by job_id) as rownumber from job_data) as inner_query
where rownumber > 1;
```

Output –



job_id	actor_id	events	language	time_spent	org	ds	rownumber
23	1005	transfer	Persian	22	D	2020-11-28 00:00:00	2
23	1004	skip	Persian	56	A	2020-11-26 00:00:00	3

Case Study 2: Investigating Metric Spike

will be working with three tables:

- **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.

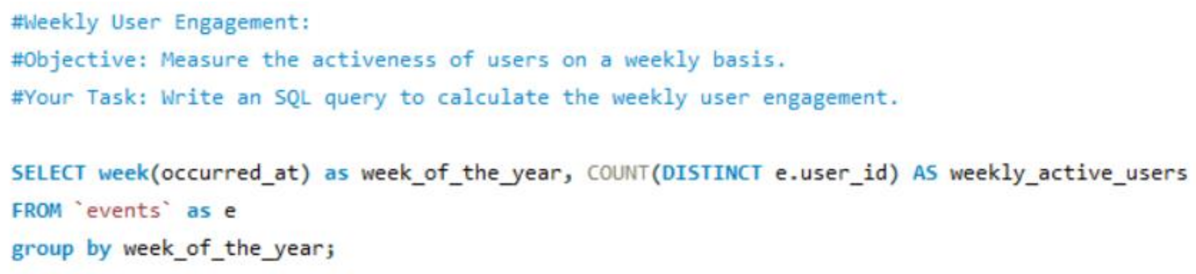
Tasks - 1

Weekly User Engagement:

- Objective: Measure the activeness of users on a weekly basis.
- Your Task: Write an SQL query to calculate the weekly user engagement.

User Engagement: To measure the activeness of a user. Measuring if the user finds quality in a product/service.

Code –



```
#Weekly User Engagement:
#Objective: Measure the activeness of users on a weekly basis.
#Your Task: Write an SQL query to calculate the weekly user engagement.

SELECT week(occurred_at) as week_of_the_year, COUNT(DISTINCT e.user_id) AS weekly_active_users
FROM `events` as e
group by week_of_the_year;
```

Output –

Output of the task in the below link –

Weekly User Engagement output.cs

https://drive.google.com/file/d/1X7tdwCwmiHyLoQqjS_CULHB03cLIeroW/view?usp=sharing

Task – 2

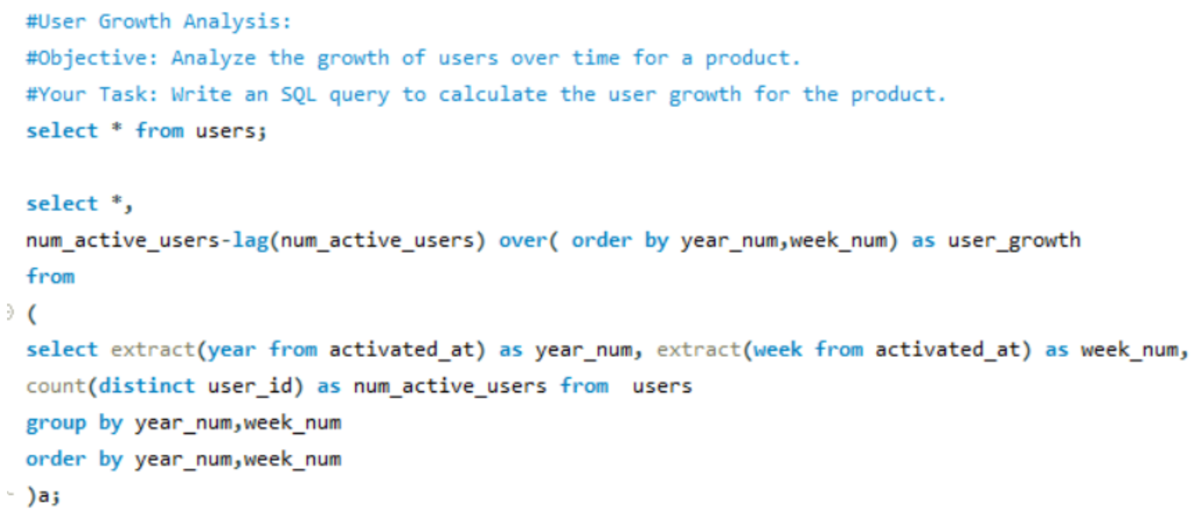
User Growth Analysis:

- Objective: Analyze the growth of users over time for a product.
- Your Task: Write an SQL query to calculate the user growth for the product.

User Growth: Amount of users growing over time for a product.

Growth = Number of active users per week

Code –



```
#User Growth Analysis:
#Objective: Analyze the growth of users over time for a product.
#Your Task: Write an SQL query to calculate the user growth for the product.
select * from users;

select *,
num_active_users-lag(num_active_users) over( order by year_num,week_num) as user_growth
from
(
select extract(year from activated_at) as year_num, extract(week from activated_at) as week_num,
count(distinct user_id) as num_active_users from  users
group by year_num,week_num
order by year_num,week_num
)a;
```

Output –

Output of the task in the below link –

User Growth Analysis output.csv

https://drive.google.com/file/d/1k7EMRpnfOSlMEyiDiGL7bIjJSwWVMGWi/view?usp=sharing

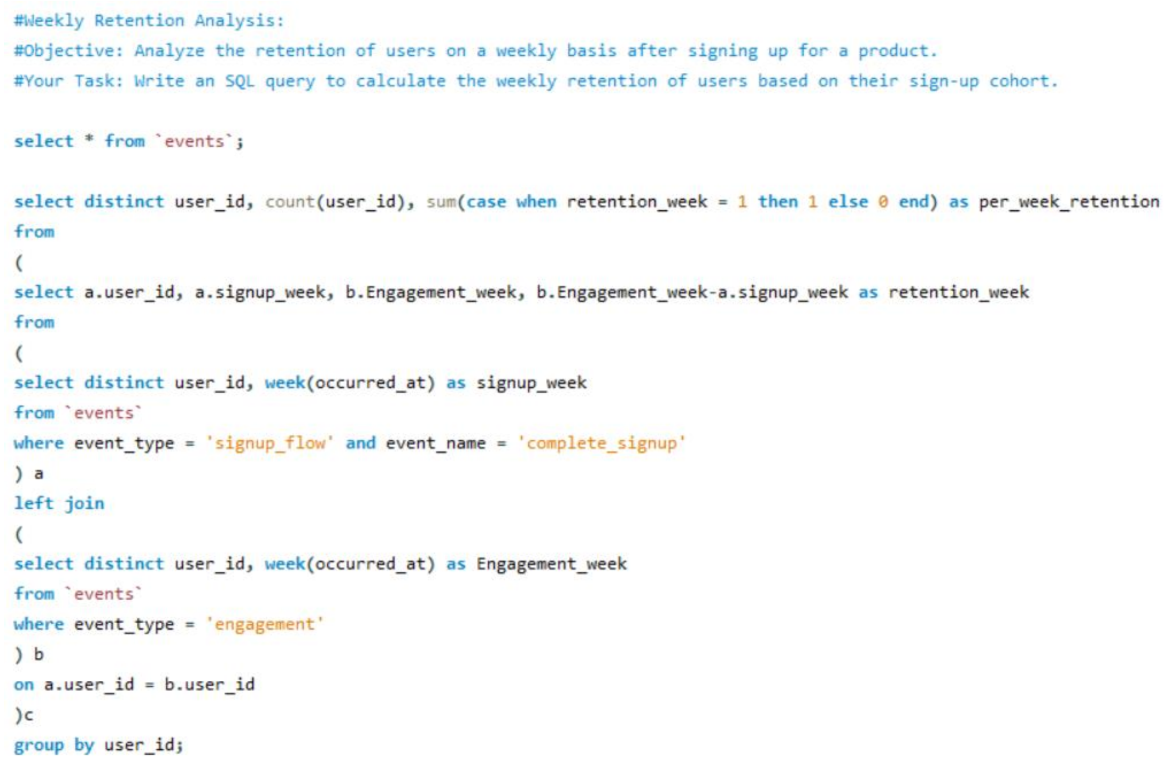
Task – 3

Weekly Retention Analysis:

- Objective: 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.

Weekly Retention: Users getting retained weekly after signing-up for a product.

Code –



```
#Weekly Retention Analysis:
#Objective: 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.

select * from `events`;

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, week(occurred_at) as signup_week
from `events`
where event_type = 'signup_flow' and event_name = 'complete_signup'
) a
left join
(
select distinct user_id, week(occurred_at) as Engagement_week
from `events`
where event_type = 'engagement'
) b
on a.user_id = b.user_id
)c
group by user_id;
```

Output –

Output of the task in the below link –

Weekly Retention Analysis output.csv

https://drive.google.com/file/d/1HqpYU12gJbMPlXTIy2RK10xbhRZBpiY9/view?usp=sharing

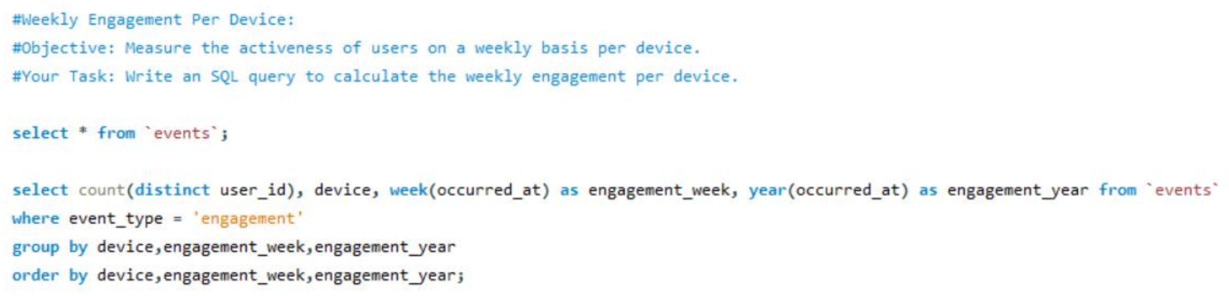
Task – 4

Weekly Engagement Per Device:

- Objective: Measure the activeness of users on a weekly basis per device.
- Your Task: Write an SQL query to calculate the weekly engagement per device.

Weekly Engagement: To measure the activeness of a user. Measuring if the user finds quality in a product/service weekly.

Code –



```
#Weekly Engagement Per Device:
#Objective: Measure the activeness of users on a weekly basis per device.
#Your Task: Write an SQL query to calculate the weekly engagement per device.

select * from `events`;

select count(distinct user_id), device, week(occurred_at) as engagement_week, year(occurred_at) as engagement_year from `events`
where event_type = 'engagement'
group by device,engagement_week,engagement_year
order by device,engagement_week,engagement_year;
```

Output –

Output of the task in the below link –

Weekly Engagement
Per Device output.csv

https://drive.google.com/file/d/1_AfH-rqYQ9jYRh3VI-Joju3dUnJKrJjB/view?usp=sharing

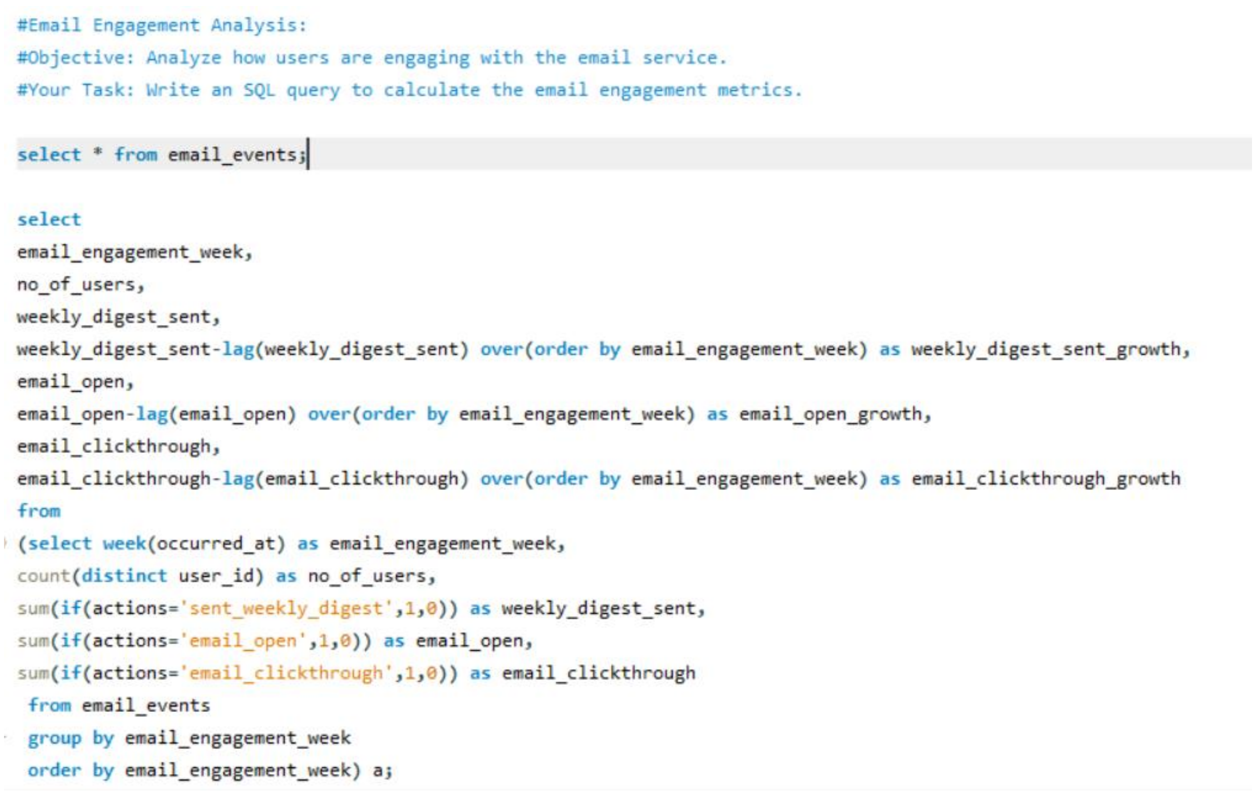
Task – 5

Email Engagement Analysis:

- Objective: Analyze how users are engaging with the email service.
- Your Task: Write an SQL query to calculate the email engagement metrics.

Email Engagement: Users engaging with the email service.

Code –



```
#Email Engagement Analysis:
#Objective: Analyze how users are engaging with the email service.
#Your Task: Write an SQL query to calculate the email engagement metrics.

select * from email_events;

select
email_engagement_week,
no_of_users,
weekly_digest_sent,
weekly_digest_sent-lag(weekly_digest_sent) over(order by email_engagement_week) as weekly_digest_sent_growth,
email_open,
email_open-lag(email_open) over(order by email_engagement_week) as email_open_growth,
email_clickthrough,
email_clickthrough-lag(email_clickthrough) over(order by email_engagement_week) as email_clickthrough_growth
from
(select week(occurred_at) as email_engagement_week,
count(distinct user_id) as no_of_users,
sum(if(actions='sent_weekly_digest',1,0)) as weekly_digest_sent,
sum(if(actions='email_open',1,0)) as email_open,
sum(if(actions='email_clickthrough',1,0)) as email_clickthrough
 from email_events
 group by email_engagement_week
 order by email_engagement_week) a;
```

Output –

Output of the task in the below link –

Email Engagement Analysis output.csv

https://drive.google.com/file/d/1H3fn8AYn_Jrf6ixHt_Uiv1GczYwcnpOK/view?usp=sharing