Project Description: The project is about operation analytics and investigating metric space. The project is about understanding user engagement with the company and the dips in user metrics.

I want to handle the analysis by understanding the datasets provided first. Then break down the table into columns and solve the query.

Approach: I take this project as a new challenge and learn new things from this project.

As I mentioned earlier, I executed the analysis by taking the required tables and after successfully getting the output studying it until I understand it and then coming to a conclusion about the analysis.

Tech-Stack Used: I used 'Microsoft SQL Server Management Studio' in this project. I Used this server because I am familiar with this software and easy to type and execute the code.

Insights: I learned about the timestamp and datetime formats from this project which was quite challenging because this was the first time I was using.

The key observations I made through this project are:

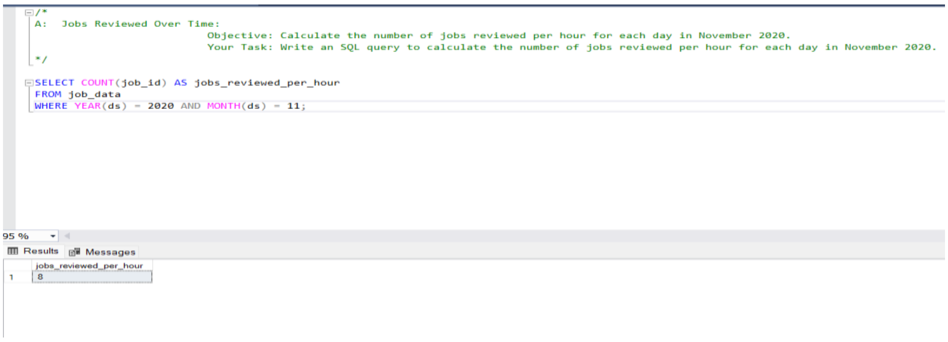
i) The **Persian** language has the highest share percentage among all other languages.

ii) The **email** engagement is around 90,000 which was huge. That means the major mode of communication is through mail.

iii) The **Macbook Pro** was the second in the weekly user engagement per device. There are no other users related to Android near Mac.

Result: I got to know about new topics related to SQL. Timestamp and datetime functions. Also, I understood how to import the data from a CSV file into the management studio.

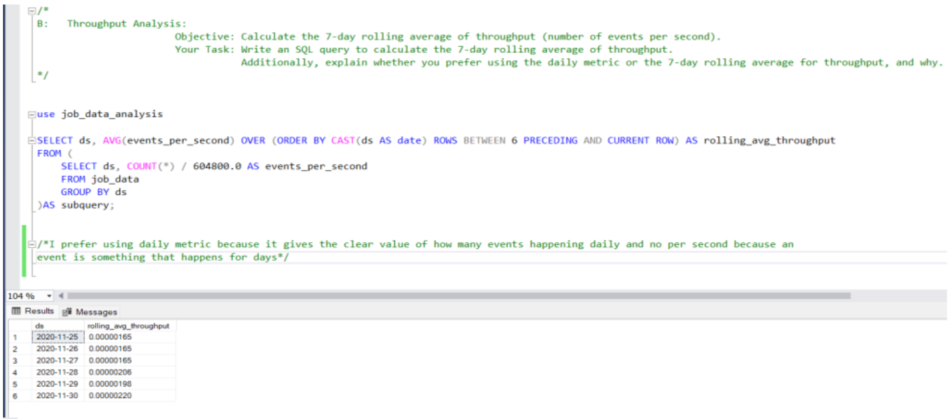
Outputs of the queries:



```
/*
A:   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.
*/

SELECT COUNT(job_id) AS jobs_reviewed_per_hour
FROM job_data
WHERE YEAR(ds) = 2020 AND MONTH(ds) = 11;
```

95 %

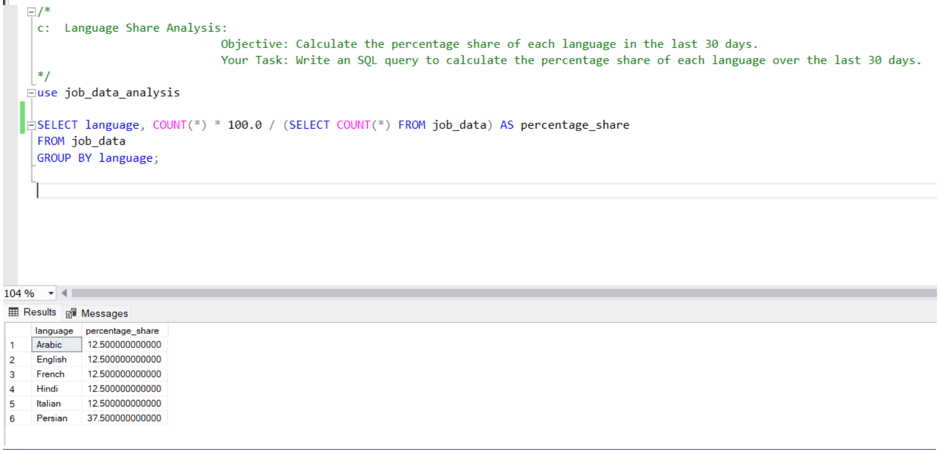
	jobs_reviewed_per_hour
1	8



```
/*
B:   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.
*/


use job_data_analysis

SELECT ds, AVG(events_per_second) OVER (ORDER BY CAST(ds AS date) ROWS BETWEEN 6 PRECEDING AND CURRENT ROW) AS rolling_avg_throughput
FROM (
    SELECT ds, COUNT(*) / 604800.0 AS events_per_second
    FROM job_data
    GROUP BY ds
)AS subquery;


/*I prefer using daily metric because it gives the clear value of how many events happening daily and no per second because an
event is something that happens for days*/
```

104 %

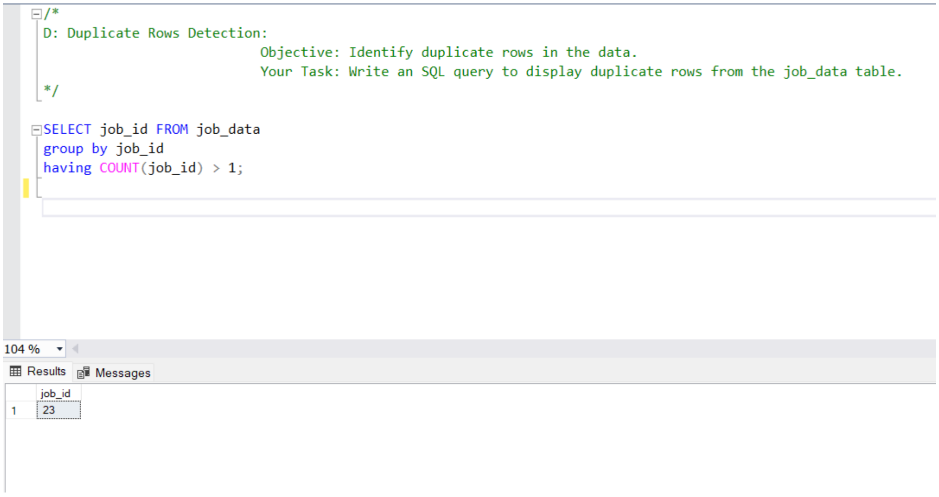
	ds	rolling_avg_throughput
1	2020-11-25	0.00000165
2	2020-11-26	0.00000165
3	2020-11-27	0.00000165
4	2020-11-28	0.00000206
5	2020-11-29	0.00000198
6	2020-11-30	0.00000220



```
/*
c:  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.
*/
use job_data_analysis

SELECT language, COUNT(*) * 100.0 / (SELECT COUNT(*) FROM job_data) AS percentage_share
FROM job_data
GROUP BY language;
```

104 %

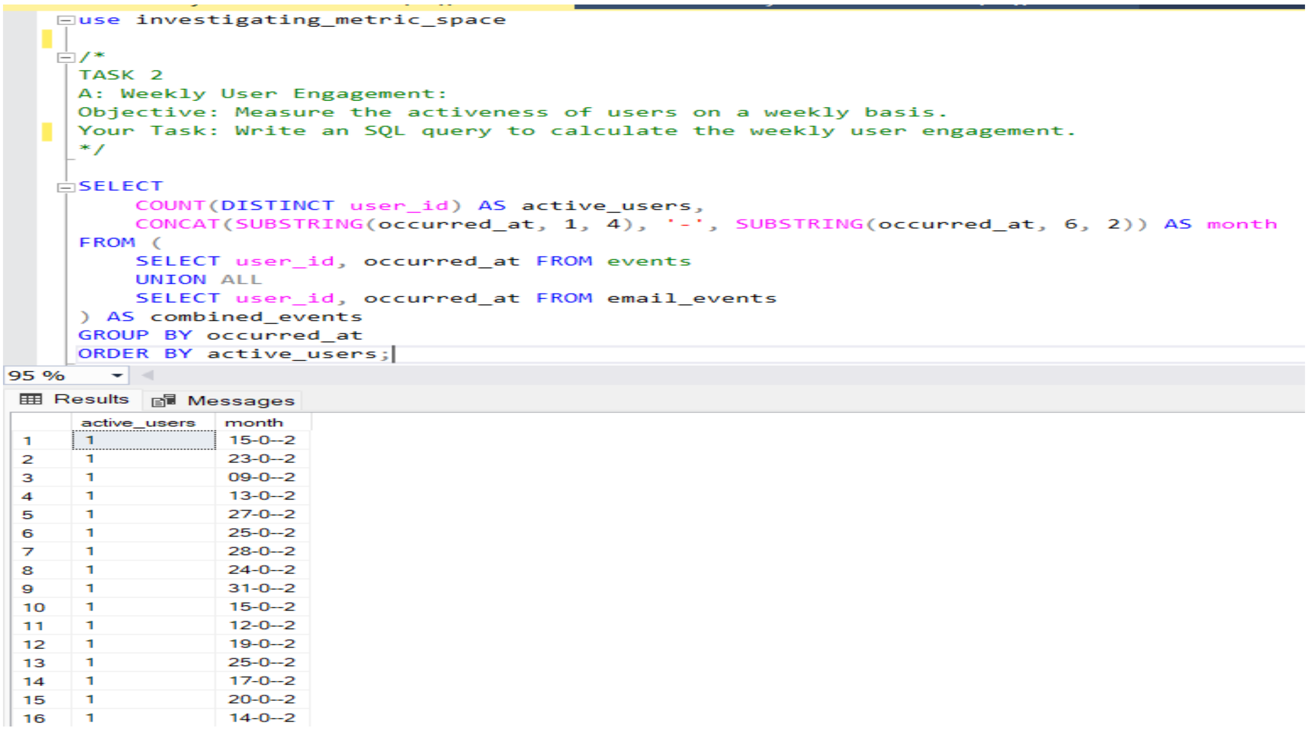
	language	percentage_share
1	Arabic	12.500000000000
2	English	12.500000000000
3	French	12.500000000000
4	Hindi	12.500000000000
5	Italian	12.500000000000
6	Persian	37.500000000000



```
/*
D: 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.
*/

SELECT job_id FROM job_data
group by job_id
having COUNT(job_id) > 1;
```

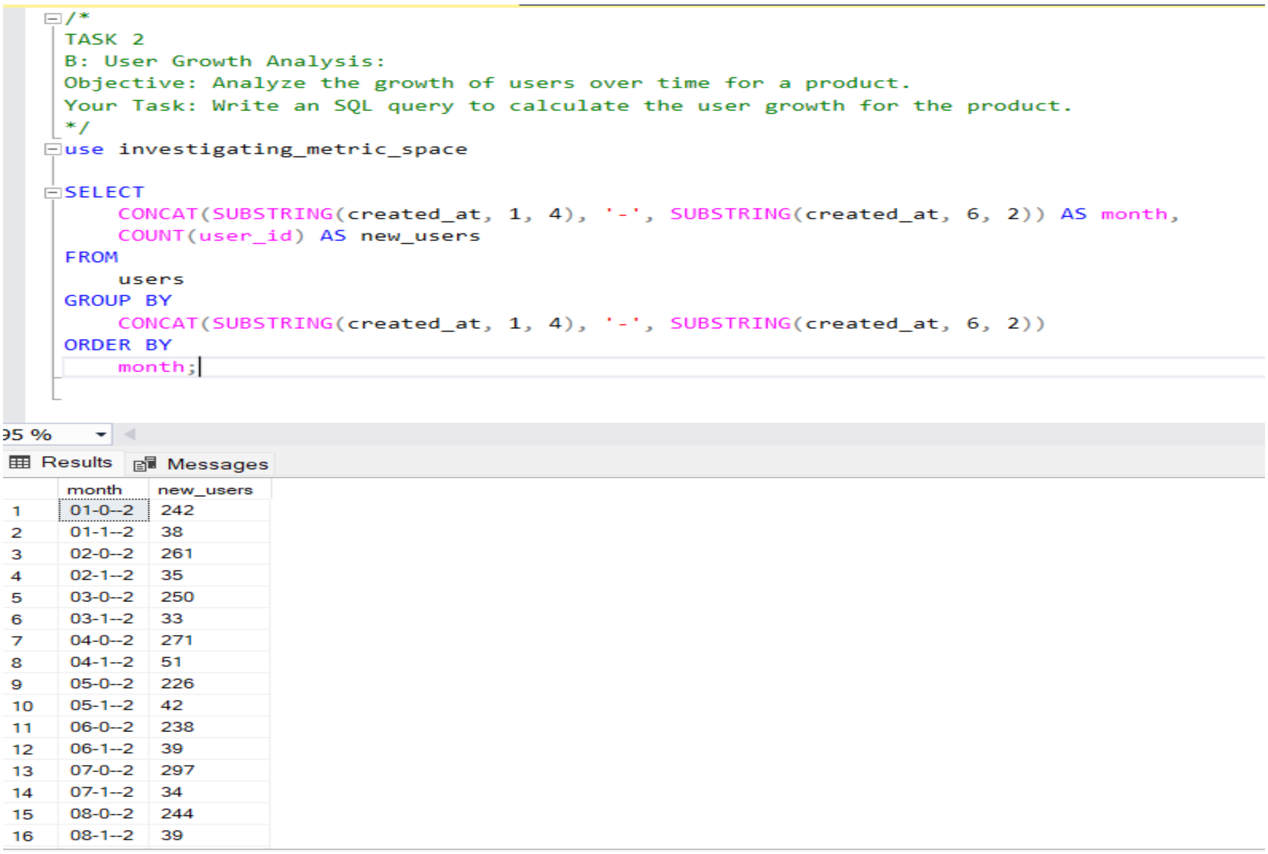
104 %

	job_id
1	23



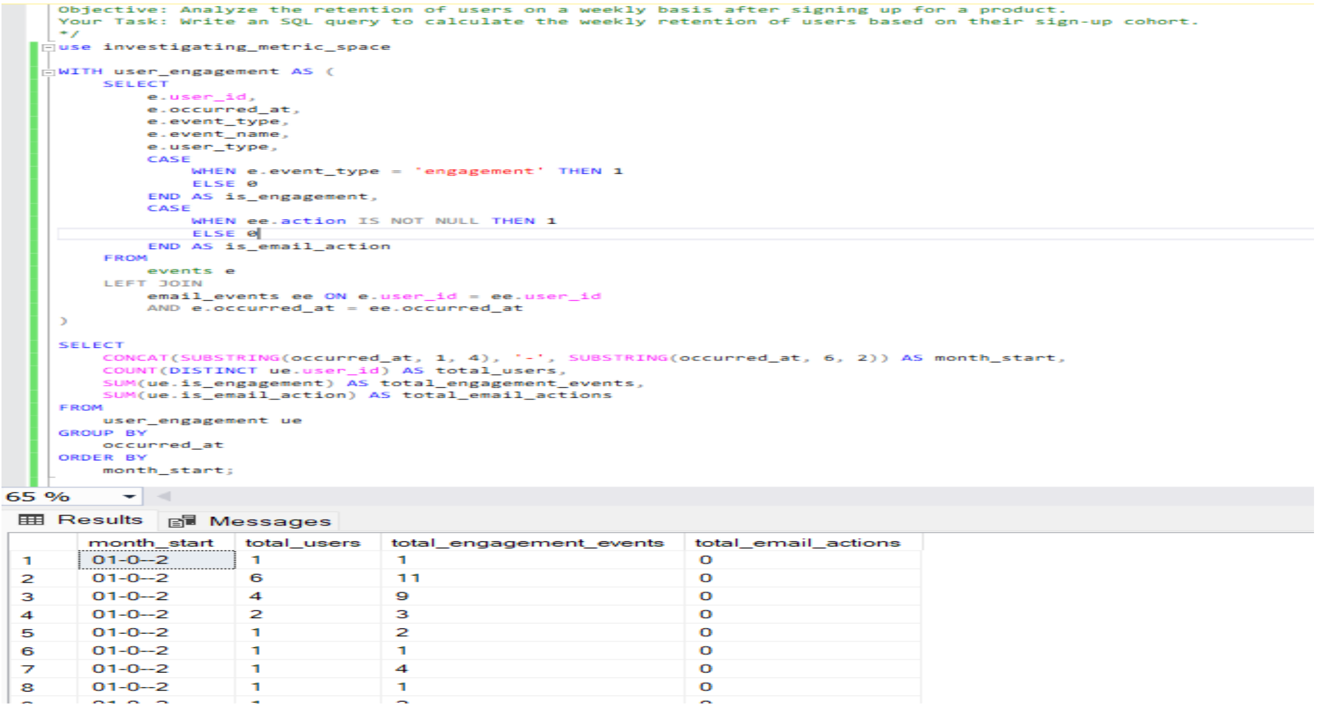
```
use investigating_metric_space

/*
TASK 2
A: 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
    COUNT(DISTINCT user_id) AS active_users,
    CONCAT(SUBSTRING(occurred_at, 1, 4), '-', SUBSTRING(occurred_at, 6, 2)) AS month
FROM (
    SELECT user_id, occurred_at FROM events
    UNION ALL
    SELECT user_id, occurred_at FROM email_events
) AS combined_events
GROUP BY occurred_at
ORDER BY active_users;
```

	active_users	month
1	1	15-0--2
2	1	23-0--2
3	1	09-0--2
4	1	13-0--2
5	1	27-0--2
6	1	25-0--2
7	1	28-0--2
8	1	24-0--2
9	1	31-0--2
10	1	15-0--2
11	1	12-0--2
12	1	19-0--2
13	1	25-0--2
14	1	17-0--2
15	1	20-0--2
16	1	14-0--2



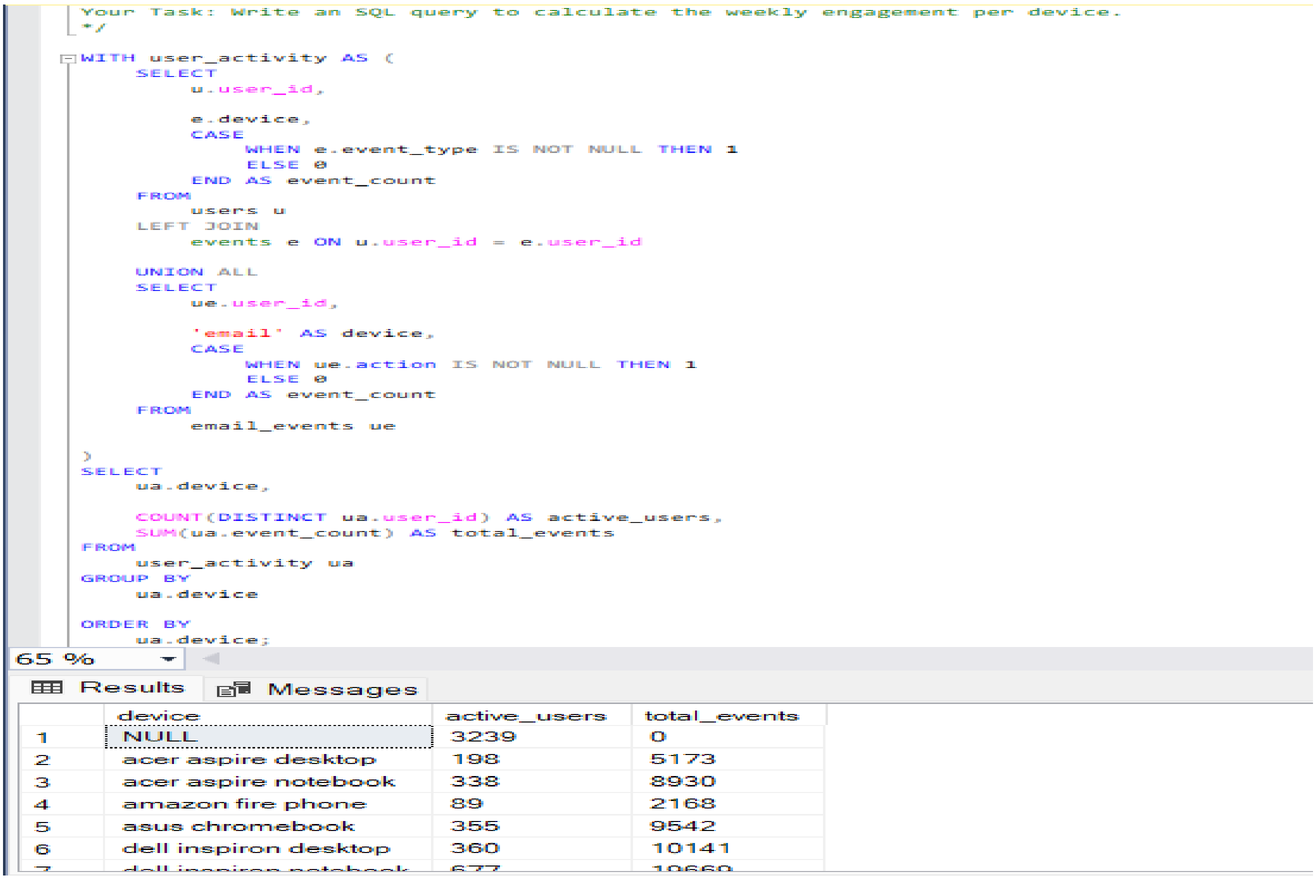
```
/*
TASK 2
B: User Growth Analysis:
Objective: Analyze the growth of users over time for a product.
Your Task: Write an SQL query to calculate the user growth for the product.
*/
use investigating_metric_space

SELECT
    CONCAT(SUBSTRING(created_at, 1, 4), '-', SUBSTRING(created_at, 6, 2)) AS month,
    COUNT(user_id) AS new_users
FROM
    users
GROUP BY
    CONCAT(SUBSTRING(created_at, 1, 4), '-', SUBSTRING(created_at, 6, 2))
ORDER BY
    month;
```

	month	new_users
1	01-0--2	242
2	01-1--2	38
3	02-0--2	261
4	02-1--2	35
5	03-0--2	250
6	03-1--2	33
7	04-0--2	271
8	04-1--2	51
9	05-0--2	226
10	05-1--2	42
11	06-0--2	238
12	06-1--2	39
13	07-0--2	297
14	07-1--2	34
15	08-0--2	244
16	08-1--2	39



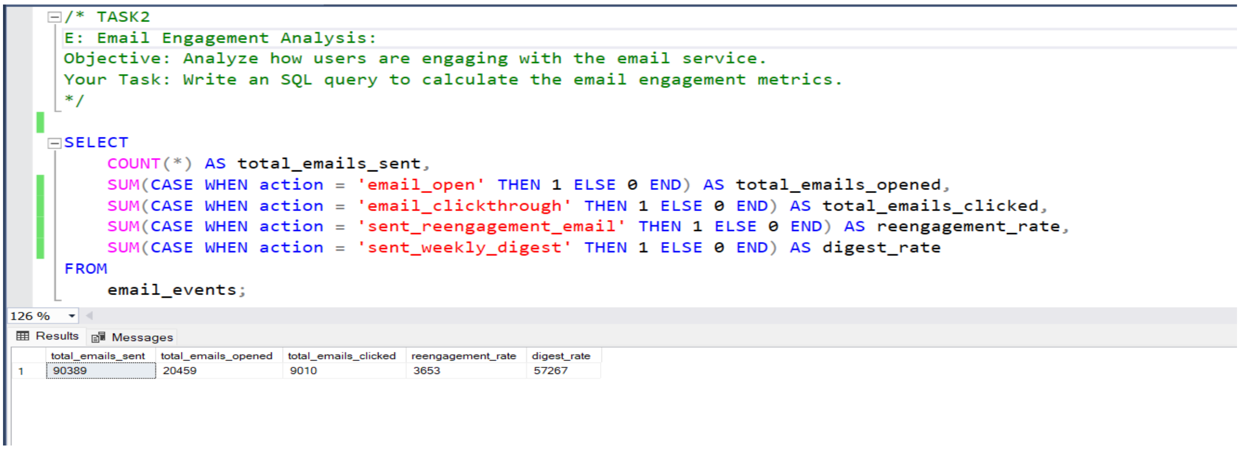
```sql
Objective: Analyze the retention of users on a weekly basis after signing up for a product.
Your Task: Write an SQL query to calculate the weekly retention of users based on their sign-up cohort.
*/
use investigating_metric_space

WITH user_engagement AS (
    SELECT
        e.user_id,
        e.occurred_at,
        e.event_type,
        e.event_name,
        e.user_type,
        CASE
            WHEN e.event_type = 'engagement' THEN 1
            ELSE 0
        END AS is_engagement,
        CASE
            WHEN ee.action IS NOT NULL THEN 1
            ELSE 0
        END AS is_email_action
    FROM
        events e
    LEFT JOIN
        email_events ee ON e.user_id = ee.user_id
        AND e.occurred_at = ee.occurred_at
)

SELECT
    CONCAT(SUBSTRING(occurred_at, 1, 4), '-', SUBSTRING(occurred_at, 6, 2)) AS month_start,
    COUNT(DISTINCT ue.user_id) AS total_users,
    SUM(ue.is_engagement) AS total_engagement_events,
    SUM(ue.is_email_action) AS total_email_actions
FROM
    user_engagement ue
GROUP BY
    occurred_at
ORDER BY
    month_start;
```

	month_start	total_users	total_engagement_events	total_email_actions
1	01-0--2	1	1	0
2	01-0--2	6	11	0
3	01-0--2	4	9	0
4	01-0--2	2	3	0
5	01-0--2	1	2	0
6	01-0--2	1	1	0
7	01-0--2	1	4	0
8	01-0--2	1	1	0



```sql
Your Task: Write an SQL query to calculate the weekly engagement per device.
*/

WITH user_activity AS (
    SELECT
        u.user_id,

        e.device,
        CASE
            WHEN e.event_type IS NOT NULL THEN 1
            ELSE 0
        END AS event_count
    FROM
        users u
    LEFT JOIN
        events e ON u.user_id = e.user_id

    UNION ALL
    SELECT
        ue.user_id,

        'email' AS device,
        CASE
            WHEN ue.action IS NOT NULL THEN 1
            ELSE 0
        END AS event_count
    FROM
        email_events ue

)
SELECT
    ua.device,

    COUNT(DISTINCT ua.user_id) AS active_users,
    SUM(ua.event_count) AS total_events
FROM
    user_activity ua
GROUP BY
    ua.device

ORDER BY
    ua.device;
```

	device	active_users	total_events
1	NULL	3239	0
2	acer aspire desktop	198	5173
3	acer aspire notebook	338	8930
4	amazon fire phone	89	2168
5	asus chromebook	355	9542
6	dell inspiron desktop	360	10141



```
/* TASK2
E: 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
    COUNT(*) AS total_emails_sent,
    SUM(CASE WHEN action = 'email_open' THEN 1 ELSE 0 END) AS total_emails_opened,
    SUM(CASE WHEN action = 'email_clickthrough' THEN 1 ELSE 0 END) AS total_emails_clicked,
    SUM(CASE WHEN action = 'sent_reengagement_email' THEN 1 ELSE 0 END) AS reengagement_rate,
    SUM(CASE WHEN action = 'sent_weekly_digest' THEN 1 ELSE 0 END) AS digest_rate
FROM
    email_events;
```

	total_emails_sent	total_emails_opened	total_emails_clicked	reengagement_rate	digest_rate
1	90389	20459	9010	3653	57267