Operation Analytics and Investigating Metric Spike

Based on Advanced SQL fundamentals

Description

This project involves analyzing job review data and user engagement metrics to derive meaningful insights. The study is divided into two case studies:

1. **Job Data Analysis:**
    - Analyzing job reviews over time.
    - Measuring throughput using a rolling average.
    - Determining language distribution.
    - Detecting duplicate job records.

2. **Investigating Metric Spike:**
    - Measuring weekly user engagement.
    - Understanding user growth.
    - Analyzing user retention based on signup cohorts.
    - Evaluating device-based engagement.
    - Investigating email engagement trends.

The insights from this study will help optimize workflow efficiency and user engagement strategies.

Approach

In this project, we will follow a structured SQL-based analytical approach-

- Data preparation – data collection and cleaning of table named job_data_final
- Create database – import the provided database into MySQL Workbench
- Data correction – we convert date column (which is in string format) into date format.
- Data exploration – understanding the table structures and relationships.
- Query execution – run multiple SQL queries to answer the provided questions.
- Analysis – analysing the results and summarize the key insights.
- Report – document findings with SQL queries and Outputs.

Tech-Stack Used

In this project, we have used-

- MySQL Workbench 8.0 CE – Used for database management and executing SQL queries.
- Microsoft Excel – Used for visualization and summarizing data.
- Google Drive – Hosting and sharing reports.

Case Study 1: Job Data Analysis

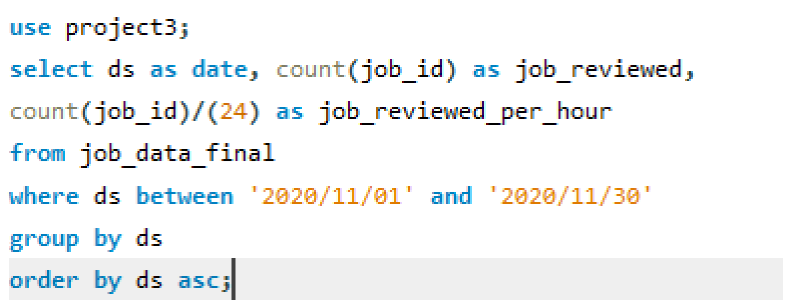
We will be working with a table named job_data_final with the following columns:

- job_id: Unique identifier of jobs
- actor_id: Unique identifier of actor
- event: The type of event (decision/skip/transfer).
- language: The Language of the content
- time_spent: Time spent to review the job in seconds.
- org: The Organization of the actor
- ds: The date in the format yyyy/mm/dd (stored as text).

SQL Tasks:

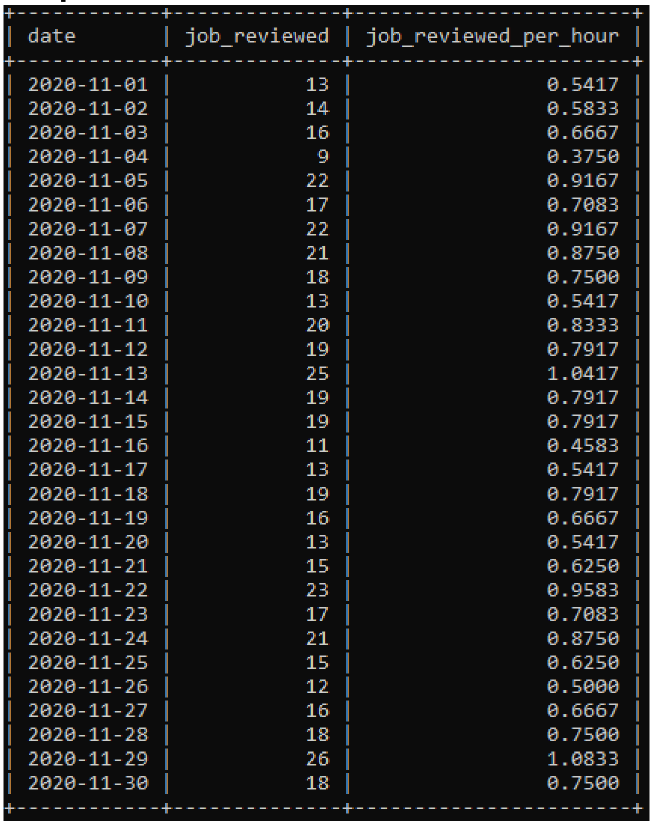
Jobs Reviewed Over Time

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

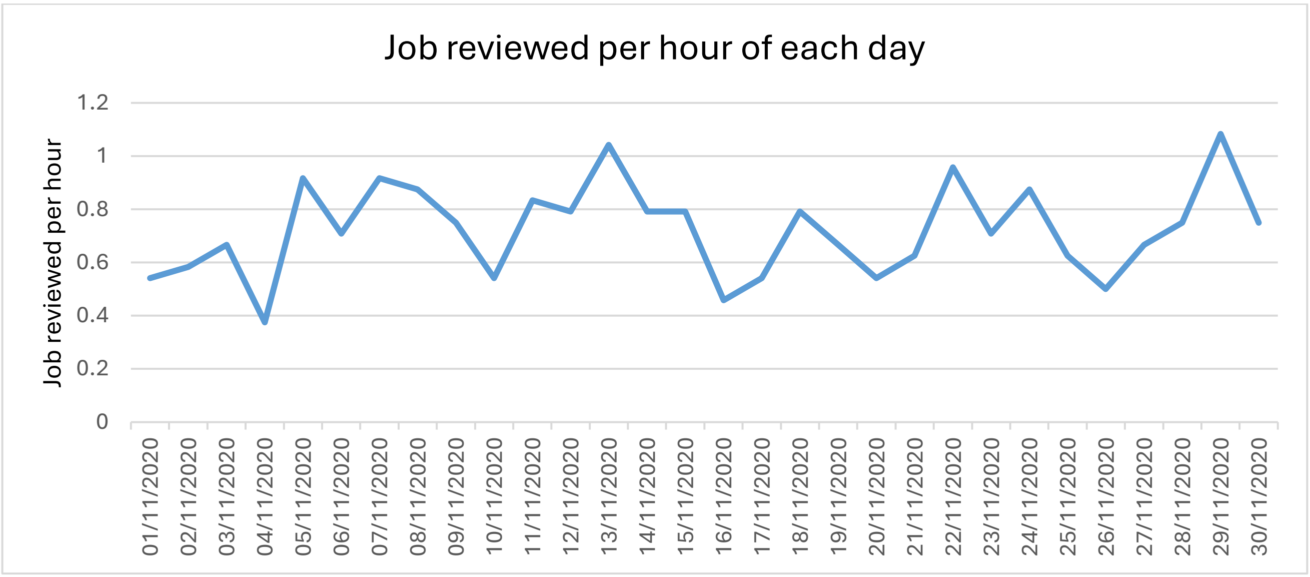


```sql
use project3;
select ds as date, count(job_id) as job_reviewed,
count(job_id)/(24) as job_reviewed_per_hour
from job_data_final
where ds between '2020/11/01' and '2020/11/30'
group by ds
order by ds asc;
```

Output

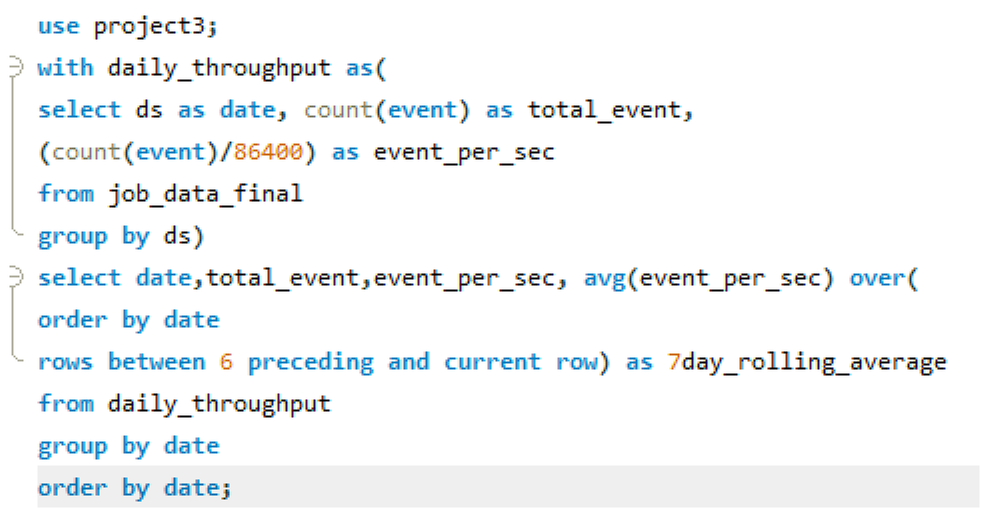


```
+------------+--------------+-----------------------+
| date       | job_reviewed | job_reviewed_per_hour |
+------------+--------------+-----------------------+
| 2020-11-01 |           13 |                0.5417 |
| 2020-11-02 |           14 |                0.5833 |
| 2020-11-03 |           16 |                0.6667 |
| 2020-11-04 |            9 |                0.3750 |
| 2020-11-05 |           22 |                0.9167 |
| 2020-11-06 |           17 |                0.7083 |
| 2020-11-07 |           22 |                0.9167 |
| 2020-11-08 |           21 |                0.8750 |
| 2020-11-09 |           18 |                0.7500 |
| 2020-11-10 |           13 |                0.5417 |
| 2020-11-11 |           20 |                0.8333 |
| 2020-11-12 |           19 |                0.7917 |
| 2020-11-13 |           25 |                1.0417 |
| 2020-11-14 |           19 |                0.7917 |
| 2020-11-15 |           19 |                0.7917 |
| 2020-11-16 |           11 |                0.4583 |
| 2020-11-17 |           13 |                0.5417 |
| 2020-11-18 |           19 |                0.7917 |
| 2020-11-19 |           16 |                0.6667 |
| 2020-11-20 |           13 |                0.5417 |
| 2020-11-21 |           15 |                0.6250 |
| 2020-11-22 |           23 |                0.9583 |
| 2020-11-23 |           17 |                0.7083 |
| 2020-11-24 |           21 |                0.8750 |
| 2020-11-25 |           15 |                0.6250 |
| 2020-11-26 |           12 |                0.5000 |
| 2020-11-27 |           16 |                0.6667 |
| 2020-11-28 |           18 |                0.7500 |
| 2020-11-29 |           26 |                1.0833 |
| 2020-11-30 |           18 |                0.7500 |
+------------+--------------+-----------------------+
```



Insight: Tracks the trend of job reviews over time to identify activity patterns.
Interpretation: An increasing trend indicates higher engagement, while declines may signal reduced user interest or platform issues.

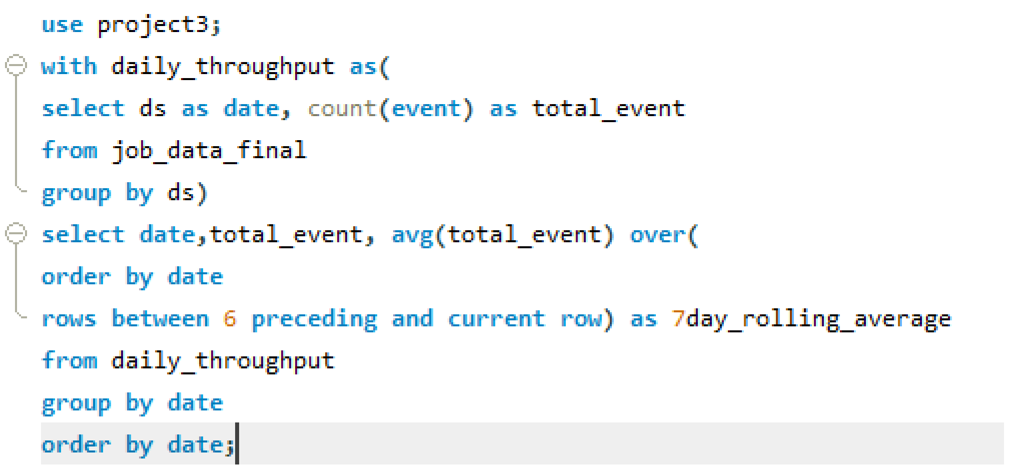
Throughput Analysis
Objective: Calculate the 7-day rolling average of throughput (number of events per second).
Task: Write an SQL query to calculate the 7-day rolling average of throughput. Additionally, explain whether you prefer using the daily metric or the 7-day rolling average for throughput, and why.



```sql
use project3;
with daily_throughput as(
select ds as date, count(event) as total_event,
(count(event)/86400) as event_per_sec
from job_data_final
group by ds)
select date,total_event,event_per_sec, avg(event_per_sec) over(
order by date
rows between 6 preceding and current row) as 7day_rolling_average
from daily_throughput
group by date
order by date;
```

Output
Check output using below link-
https://drive.google.com/file/d/1nZyR2dNXsU7SQcm6N340dVddWNcj999z/view?usp=drive_link

But if we want to calculate 7 day rolling average of throughput on the basis of events per day then,



```sql
use project3;
with daily_throughput as(
select ds as date, count(event) as total_event
from job_data_final
group by ds)
select date,total_event, avg(total_event) over(
order by date
rows between 6 preceding and current row) as 7day_rolling_average
from daily_throughput
group by date
order by date;
```

Output
Check output using below link-
https://drive.google.com/file/d/10B9PlMHu6BaPJFNOEkRTku4KDUSkXKPs/view?usp=drive_link

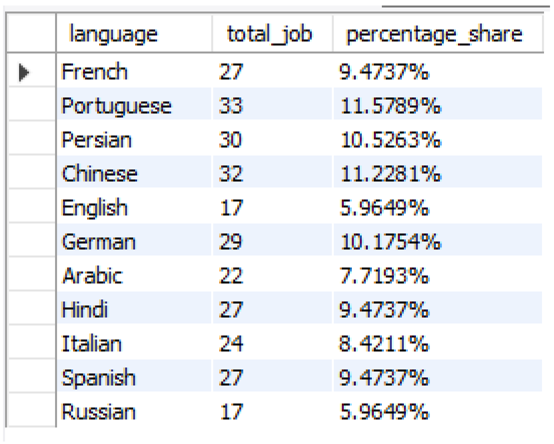
Insight: Measures the efficiency of job processing over time.
Interpretation: Higher throughput suggests improved operational efficiency, while lower throughput may indicate bottlenecks or delays.

“I will preferably use 7-day rolling average because 7-day rolling average smooths short-term fluctuations and making it easier to identify trends compared to daily metrics.”

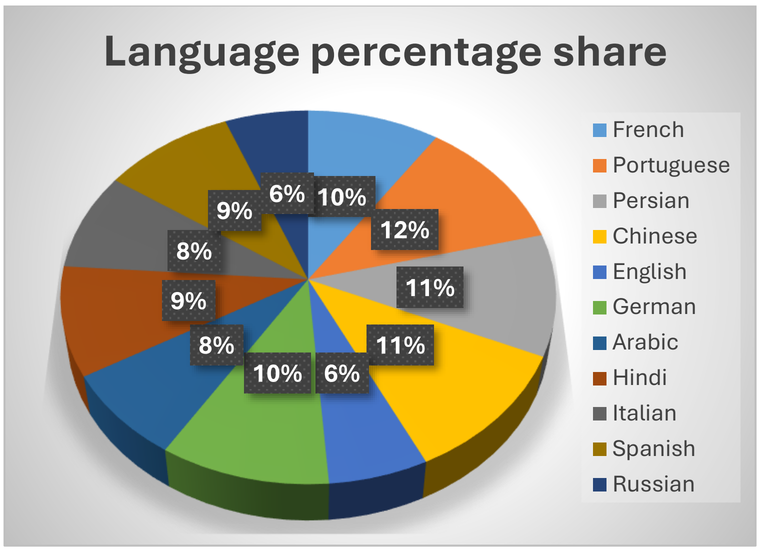
Language Share Analysis
Objective: Calculate the percentage share of each language in the last 30 days.
Task: Write an SQL query to calculate the percentage share of each language over the last 30 days.

```
use project3;
select language, count(job_id) as total_job,
concat((count(job_id)/(select count(language)
from job_data_final
where ds>=date_sub(
(select max(ds) from job_data_final), interval 30 day)))*100,"%") as percentage_share
from job_data_final
where ds>=date_sub((select max(ds) from job_data_final), interval 30 day)
group by language;
```

Output



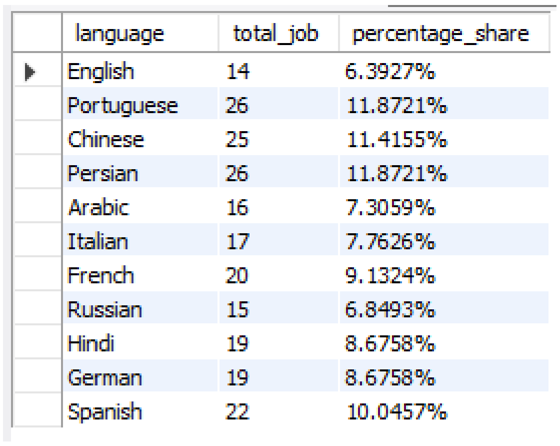
language	total_job	percentage_share
French	27	9.4737%
Portuguese	33	11.5789%
Persian	30	10.5263%
Chinese	32	11.2281%
English	17	5.9649%
German	29	10.1754%
Arabic	22	7.7193%
Hindi	27	9.4737%
Italian	24	8.4211%
Spanish	27	9.4737%
Russian	17	5.9649%



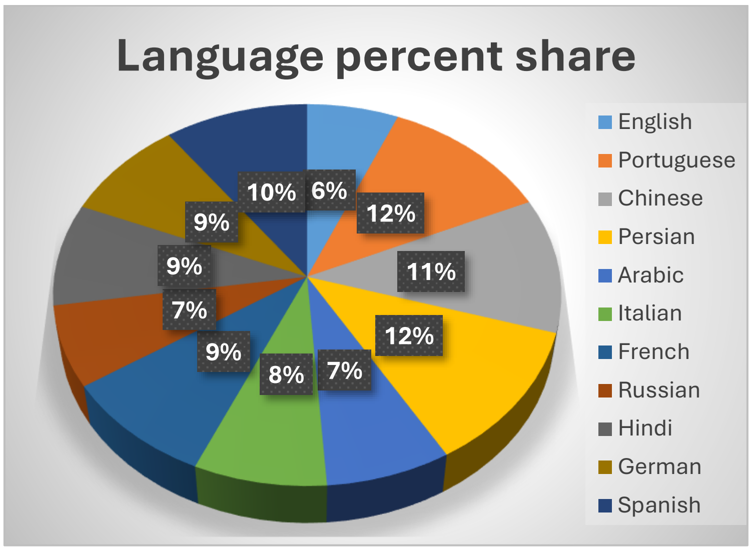
If we want to calculate percentage share for 30 days from current date then,

```
use project3;
select language, count(job_id) as total_job,
concat((count(job_id)/(select count(language)
from job_data_final
where ds>=date_sub(
curdate(), interval 30 day)))*100,"%") as percentage_share
from job_data_final
where ds>=date_sub(curdate(), interval 30 day)
group by language;
```

Output



language	total_job	percentage_share
English	14	6.3927%
Portuguese	26	11.8721%
Chinese	25	11.4155%
Persian	26	11.8721%
Arabic	16	7.3059%
Italian	17	7.7626%
French	20	9.1324%
Russian	15	6.8493%
Hindi	19	8.6758%
German	19	8.6758%
Spanish	22	10.0457%



Insight: Identifies the distribution of different languages used in job postings.
Interpretation: Helps understand language preferences, aiding in localization strategies and market expansion efforts.

Duplicate Rows Detection
Objective: Identify duplicate rows in the data.
Task: Write an SQL query to display duplicate rows from the job_data_final table.

```sql
use project3;
SELECT ds, job_id, language, event,actor_id, org,
COUNT(*) AS duplicate_count
FROM job_data_final
GROUP BY ds, job_id, language, event, actor_id, org
HAVING COUNT(*) > 1;
```

Output
Check output using below link-
https://drive.google.com/file/d/1bf8gxl93ZLpgWwlqHLtyqgYvEH2Cr4Mq/view?usp=drive_link

Insight: Detects duplicate entries in the dataset.
Interpretation: Helps maintain data integrity and accuracy by identifying and removing redundant records.

Case Study 2: Investigating Metric Spike

We 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.

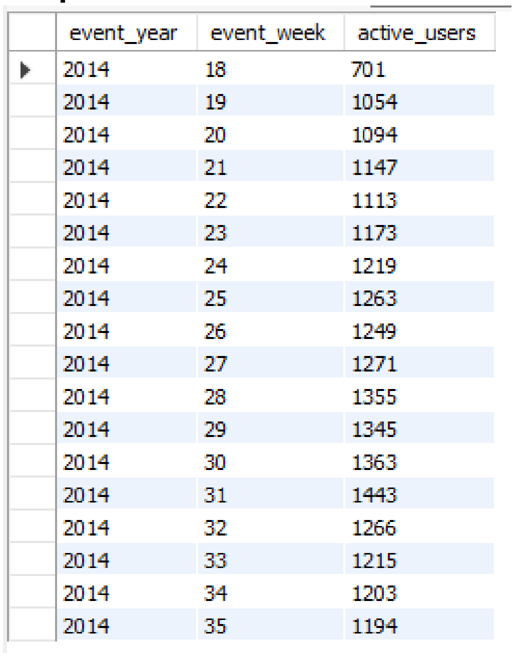
SQL Tasks:

Weekly User Engagement

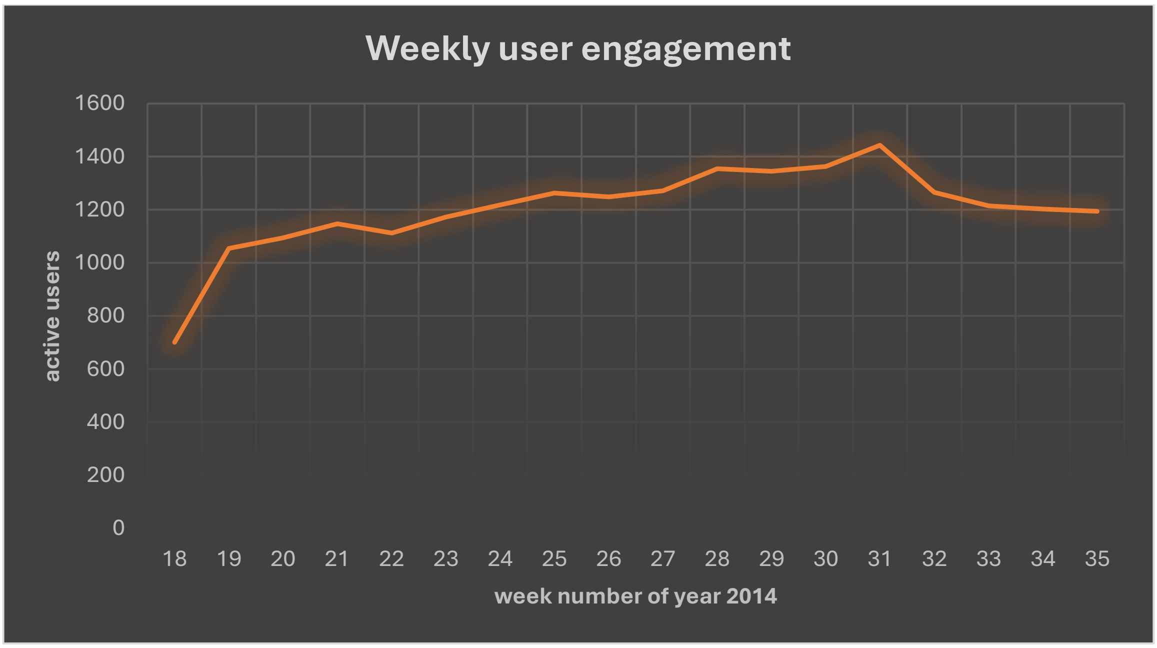
Objective: Measure the activeness of users on a weekly basis.
Task: Write an SQL query to calculate the weekly user engagement.

```sql
use project3;
SELECT YEAR(occurred_at) AS event_year,
WEEK(occurred_at,1) AS event_week,
COUNT(DISTINCT user_id) AS active_users
FROM events
GROUP BY event_year, event_week
ORDER BY event_year, event_week;
```

Output



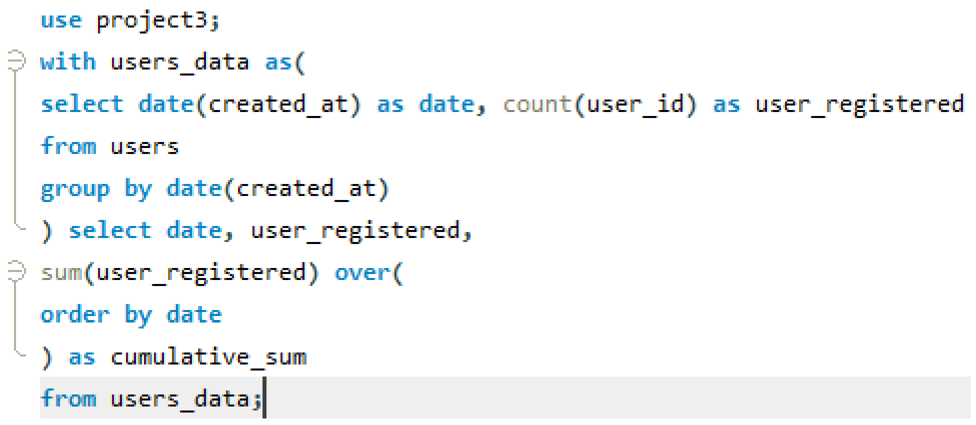
event_year	event_week	active_users
2014	18	701
2014	19	1054
2014	20	1094
2014	21	1147
2014	22	1113
2014	23	1173
2014	24	1219
2014	25	1263
2014	26	1249
2014	27	1271
2014	28	1355
2014	29	1345
2014	30	1363
2014	31	1443
2014	32	1266
2014	33	1215
2014	34	1203
2014	35	1194



Insight: Tracking weekly user engagement helps identify patterns in user activity, highlighting peak engagement periods and potential drop-offs.
Interpretation: A steady or increasing number of active users indicates strong engagement, while a decline may signal issues with user retention or content relevance.

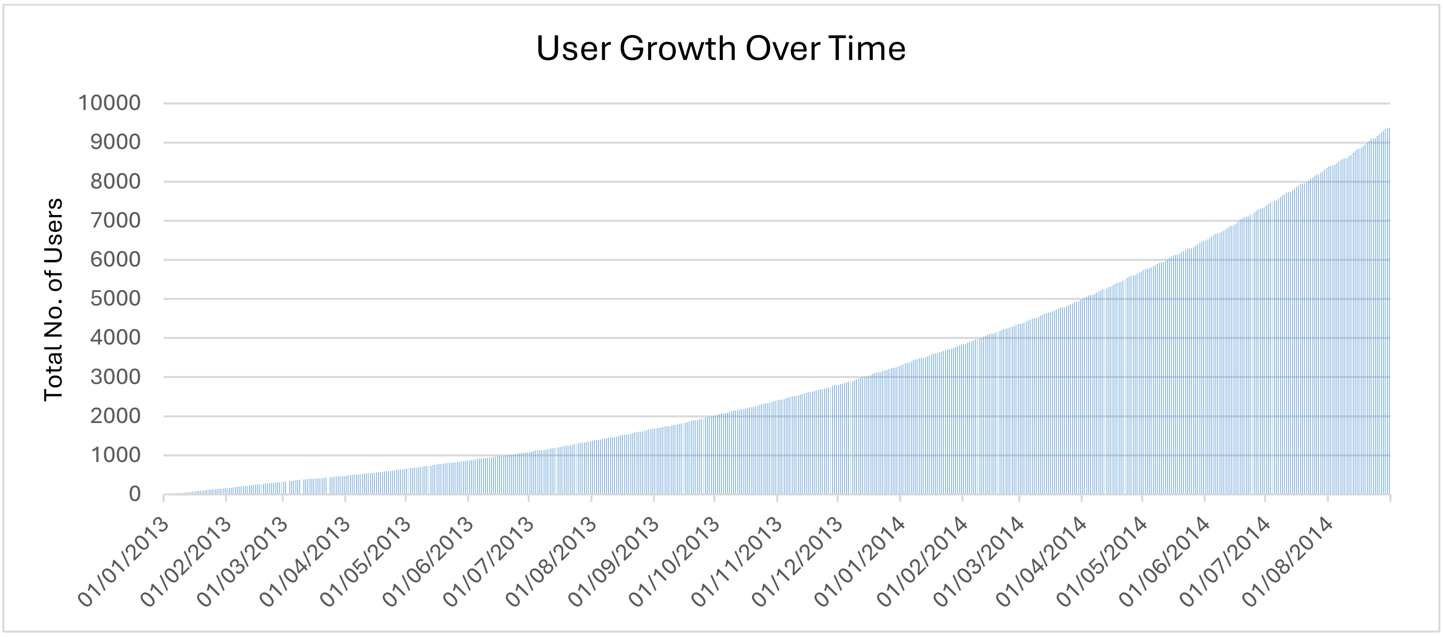
User Growth Analysis

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



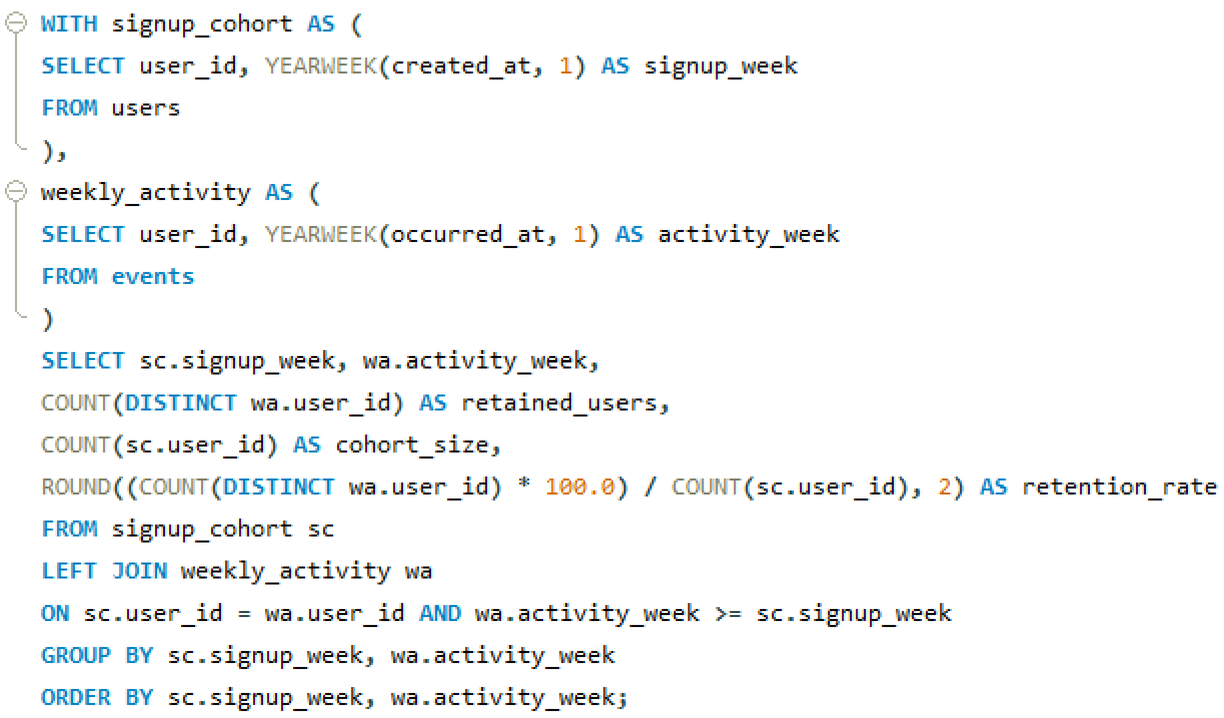
```
use project3;
with users_data as(
select date(created_at) as date, count(user_id) as user_registered
from users
group by date(created_at)
) select date, user_registered,
sum(user_registered) over(
order by date
) as cumulative_sum
from users_data;
```

Output
Check output using below link-
https://drive.google.com/file/d/1k3DjJQZH9sYwyiWRHbH5yDxZ0BV1-OI7/view?usp=drive_link



Insight: User growth trends reveal how effectively new users are being acquired over time.
Interpretation: A consistent increase suggests successful acquisition strategies, while fluctuations may indicate external factors or campaign effectiveness.

Weekly Retention Analysis
<u>Objective:</u> Analyze the retention of users on a weekly basis after signing up for a product.
<u>Task:</u> Write an SQL query to calculate the weekly retention of users based on their sign-up cohort.

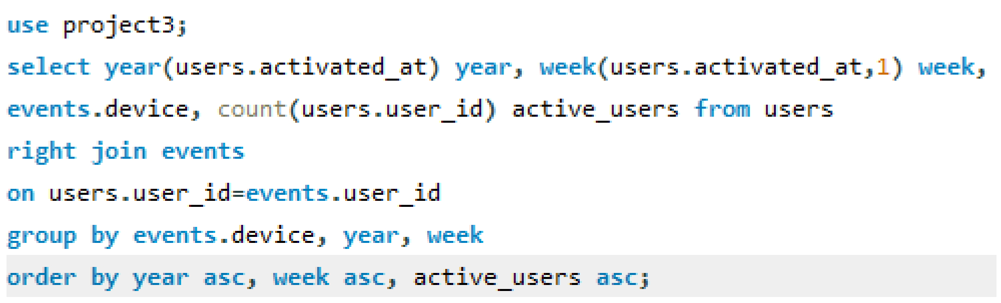


```
WITH signup_cohort AS (
SELECT user_id, YEARWEEK(created_at, 1) AS signup_week
FROM users
),
weekly_activity AS (
SELECT user_id, YEARWEEK(occurred_at, 1) AS activity_week
FROM events
)
SELECT sc.signup_week, wa.activity_week,
COUNT(DISTINCT wa.user_id) AS retained_users,
COUNT(sc.user_id) AS cohort_size,
ROUND((COUNT(DISTINCT wa.user_id) * 100.0) / COUNT(sc.user_id), 2) AS retention_rate
FROM signup_cohort sc
LEFT JOIN weekly_activity wa
ON sc.user_id = wa.user_id AND wa.activity_week >= sc.signup_week
GROUP BY sc.signup_week, wa.activity_week
ORDER BY sc.signup_week, wa.activity_week;
```

Output
Check output using below link-
https://drive.google.com/file/d/140KKNhvGe4PLStIpmdI9ggpvNArCxLQX/view?usp=drive_link

Insight: Tracks how many users return each week after signing up.
Interpretation: Helps understand user retention trends, indicating product stickiness and user engagement over time.

Weekly Engagement Per Device
<u>Objective:</u> Measure the activeness of users on a weekly basis per device.
<u>Task:</u> Write an SQL query to calculate the weekly engagement per device.

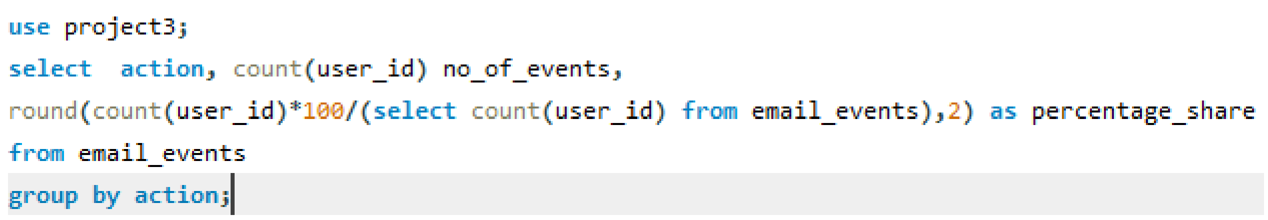


```
use project3;
select year(users.activated_at) year, week(users.activated_at,1) week,
events.device, count(users.user_id) active_users from users
right join events
on users.user_id=events.user_id
group by events.device, year, week
order by year asc, week asc, active_users asc;
```

Output
Check output using below link-
https://drive.google.com/file/d/1TRnqTOVossslUagWdJ3gDYehwxrmvbUH/view?usp=drive_link

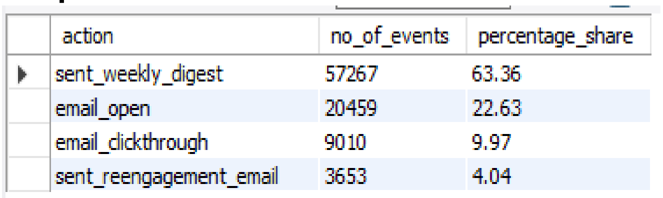
Insight: Measures user activity across different devices on a weekly basis.
Interpretation: Helps identify preferred devices, optimize user experience, and improve platform accessibility.

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

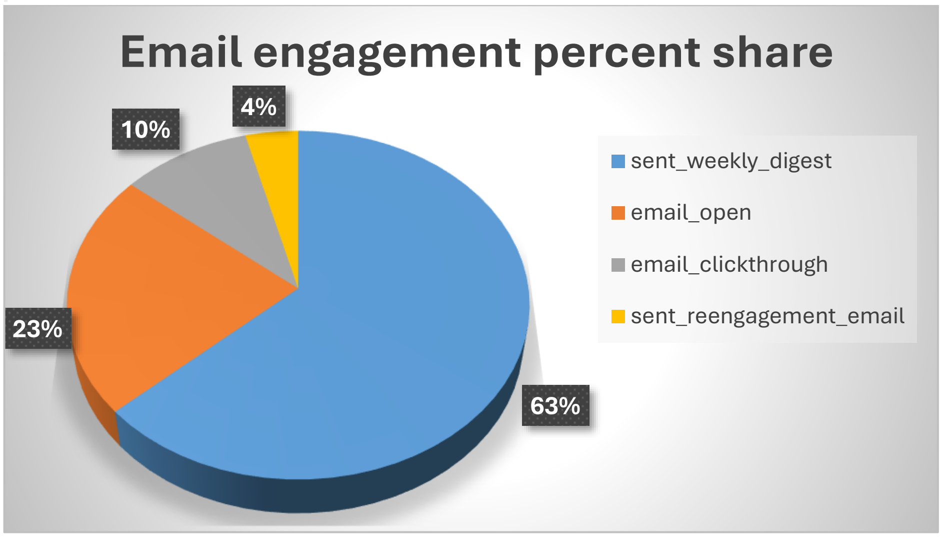


```
use project3;
select  action, count(user_id) no_of_events,
round(count(user_id)*100/(select count(user_id) from email_events),2) as percentage_share
from email_events
group by action;
```

Output



	action	no_of_events	percentage_share
▸	sent_weekly_digest	57267	63.36
	email_open	20459	22.63
	email_clickthrough	9010	9.97
	sent_reengagement_email	3653	4.04



Insight: Analyses user interaction with emails, including opens, clicks, and responses.
Interpretation: Helps optimize email campaigns, improve engagement rates, and enhance communication strategies.

Insights

- User activity fluctuates across time, with peak engagement during specific hours and devices playing a key role.
- The 7-day rolling average smooths fluctuations, providing clearer trends. Growth analysis shows user expansion trends, while retention drops over time, emphasizing the importance of onboarding.
- Language distribution highlights the need for localization, and duplicate data issues stress the importance of clean data.
- Email engagement insights suggest optimizing content and targeting for better interaction.

These findings help refine user strategies, improve efficiency, and enhance overall engagement.

Results

- Successfully analyze job reviews, throughput trends, language distribution, and detected duplicate records.
- Measured user engagement, retention, and email interaction to assess activity levels and trends.
- Identified peak job review times and user activity patterns across different devices.
- Provided recommendations to improve review efficiency, user retention, and email engagement strategies.

Useful links

To know how we import data into mySQL and change column formats go to below link-
https://drive.google.com/file/d/151Fr5HzkxxniyaJ42v1Rk3vyF-KtOtdY/view?usp=drive_link

For detailed steps watch trainity video using below link-
https://drive.google.com/file/d/12bZ_pTKQ4XaSDxpQGPsjpOz32t5r4sAE/view?usp=drive_link

link for job_data_final dataset which we used for case study 1-
https://drive.google.com/file/d/1fu8Jnx2KvDqEaMCHmh8KBhLrzHp-31ck/view?usp=drive_link