PROJECT 03: OPERATION ANALYTICS & INVESTIGATING METRIC SPIKE

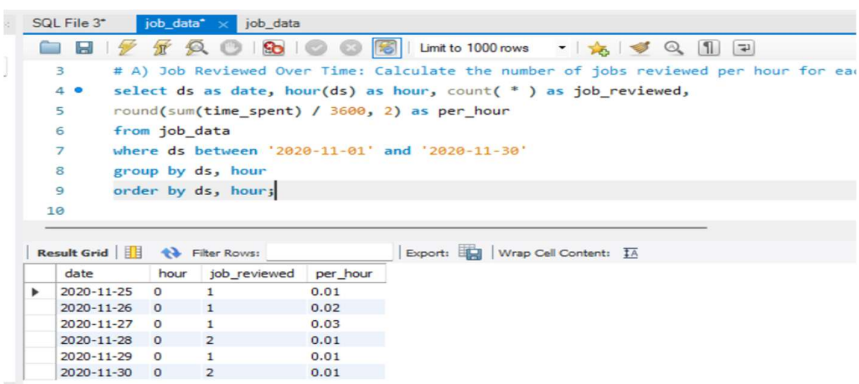
Case Study 1: Job Data Analysis

A) **Job Reviewed Over Time:** Calculate the number of jobs reviewed per hour for each day in november 2020.

Code:

```
select ds as date, hour(ds) as hour, count( * ) as job_reviewed,
round(sum(time_spent) / 3600, 2) as per_hour
from job_data
where ds between '2020-11-01' and '2020-11-30'
group by ds, hour
order by ds, hour;
```

Output:



date	hour	job_reviewed	per_hour
2020-11-25	0	1	0.01
2020-11-26	0	1	0.02
2020-11-27	0	1	0.03
2020-11-28	0	2	0.01
2020-11-29	0	1	0.01
2020-11-30	0	2	0.01

Result: In this query we calculate the number of jobs reviewed per hour for each day in November 2020.

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

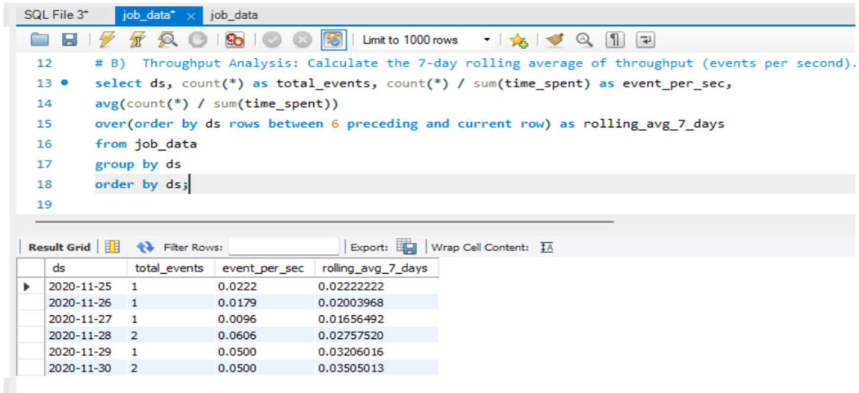
Code:

```
select ds, count( * ) as total_events, count( * ) / sum(time_spent) as event_per_sec,
avg(count( * ) / sum(time_spent))
over(order by ds rows between 6 preceding and current row) as rolling_avg_7_days
from job_data
```

group by ds

order by ds;

Output:



ds	total_events	event_per_sec	rolling_avg_7_days
2020-11-25	1	0.0222	0.02222222
2020-11-26	1	0.0179	0.02003968
2020-11-27	1	0.0096	0.01656492
2020-11-28	2	0.0606	0.02757520
2020-11-29	1	0.0500	0.03206016
2020-11-30	2	0.0500	0.03505013

Result: In this query we calculates the number of events per second for each day and computes a 7-day rolling average of throughput.

C) **Language Share Analysis**: Write an SQL query to calculate the percentage share of each language over the last 30 days.

Code:

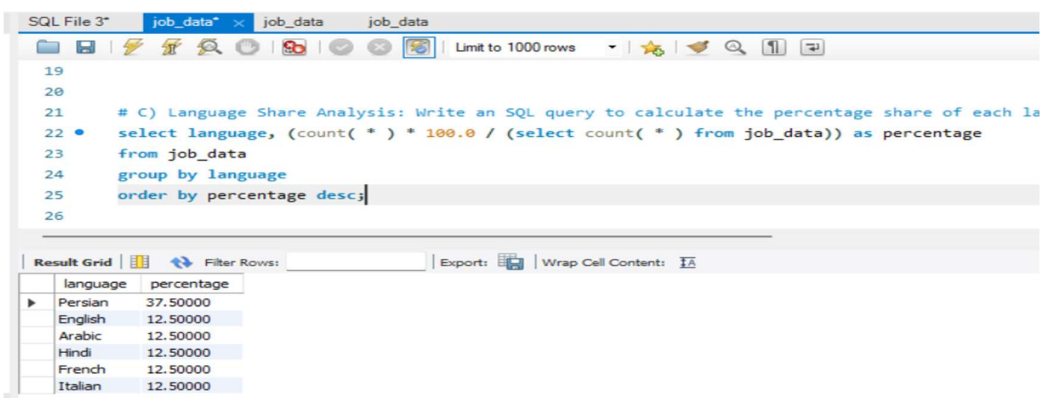
select language, (count(*) * 100.0 / (select count(*) from job_data)) as percentage

from job_data

group by language

order by percentage desc;

Output:



language	percentage
Persian	37.50000
English	12.50000
Arabic	12.50000
Hindi	12.50000
French	12.50000
Italian	12.50000

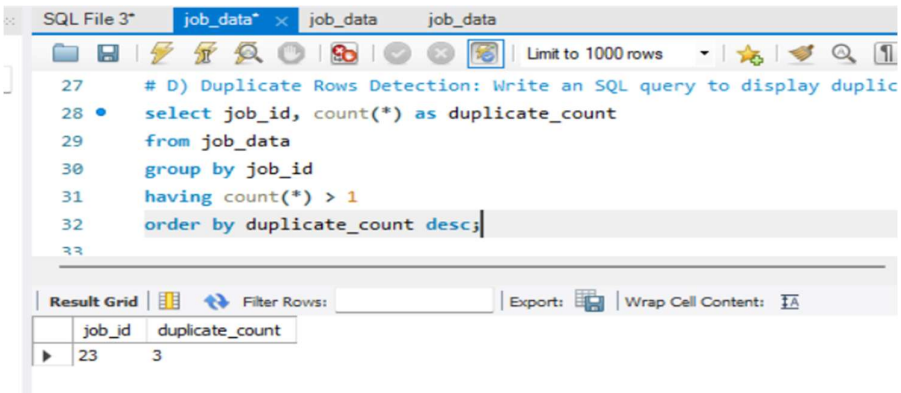
Result: In this query we calculate the percentage of jobs associated with each language over the total job count in the last 30 days, helping to understand language distribution in the dataset.

D) **Duplicate Rows Detection:** Write an SQL query to display duplicate rows from the job_data table.

Code:

```
select job_id, count( * ) as duplicate_count

from job_data

group by job_id

having count(*) > 1

order by duplicate_count desc;
```

Output:



Result: In this query we identifie duplicate job id's from the job_data table by counting how many times each job_id appears, helping to detect repeated entries.

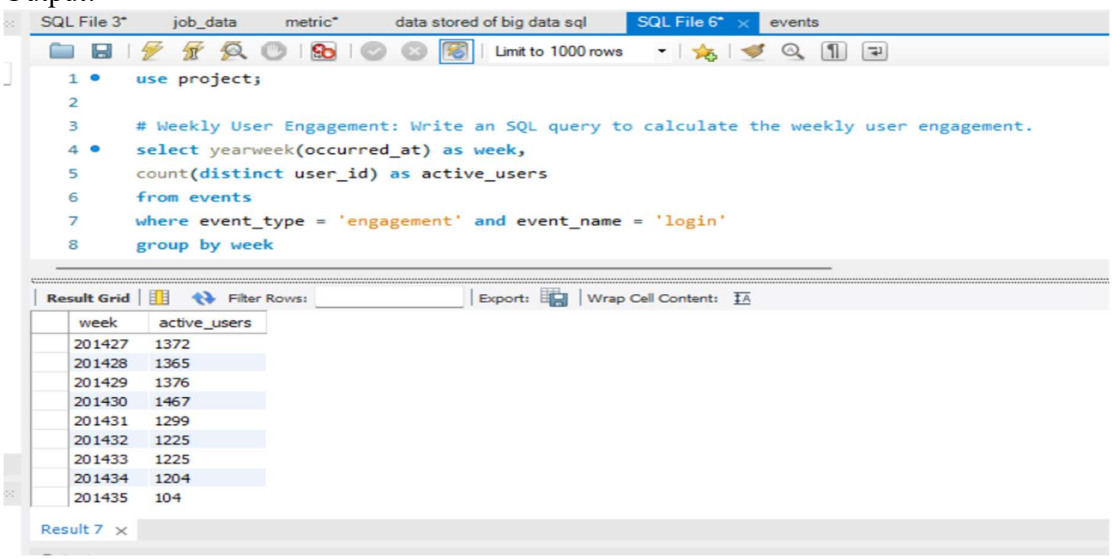
Case Study 2: Investigating Metric Spike

A) **Weekly User Engagement:** Write an SQL query to calculate the weekly user engagement.

Code: select yearweek(occurred_at) as week,

```
count(distinct user_id) as active_users
from events
where event_type = 'engagement' and event_name = 'login'
group by week
order by week;
```

Output: :



Result:

Sr.No.	week	active_users
1	201417	663
2	201418	1068
3	201419	1113
4	201420	1154
5	201421	1121
6	201422	1186
7	201423	1232

8	201424	1275
9	201425	1264
10	201426	1302
11	201427	1372
12	201428	1365
13	201429	1376
14	201430	1467
15	201431	1299
16	201432	1225
17	201433	1225
18	201434	1204
19	201435	104

Insights: This query helps track how many unique users log in each week. If the number of active users is increasing, more users are attached to the platform. A decline in active users may indicate that users are losing interest or facing issues. Looking at different weeks, we can understand how user activity changes over time

B) **User Growth Analysis:** Write an SQL query to calculate the user growth for the product.

Code: select date(created_at) as day,

count(user_id) as total_users,

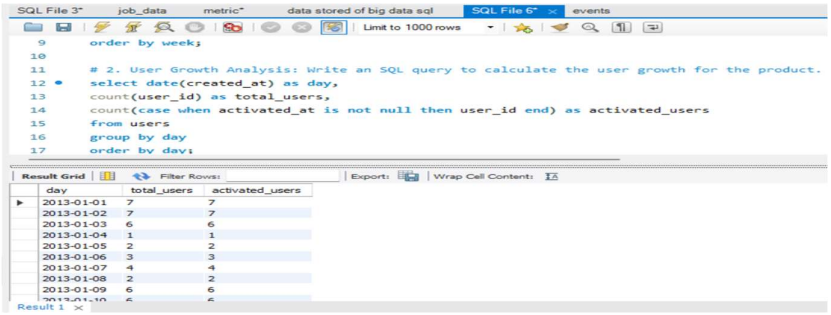
count(case when activated_at is not null then user_id end) as activated_users

from users

group by day

order by day;

Output:



Result:

day	total_users	activated_users
01-01-2013	7	7
02-01-2013	7	7
03-01-2013	6	6
04-01-2013	1	1
05-01-2013	2	2
06-01-2013	3	3
07-01-2013	4	4
08-01-2013	2	2
09-01-2013	6	6

Insights: It calculates the query of how many users sign up each day and how many their accounts are activated. If the total users continue to grow, the product is attracting new people. If the number of active users is low, many users sign up but do not use the product. A high activation rate means that users find the product useful after sign up.

C) **Weekly Retention Analysis:** Write an SQL query to calculate the weekly retention of users based on their sign-up cohort.

Code: select yearweek(u.created_at) as week, yearweek(e.occurred_at) as activity_week,

count(distinct e.user_id) as retained_users

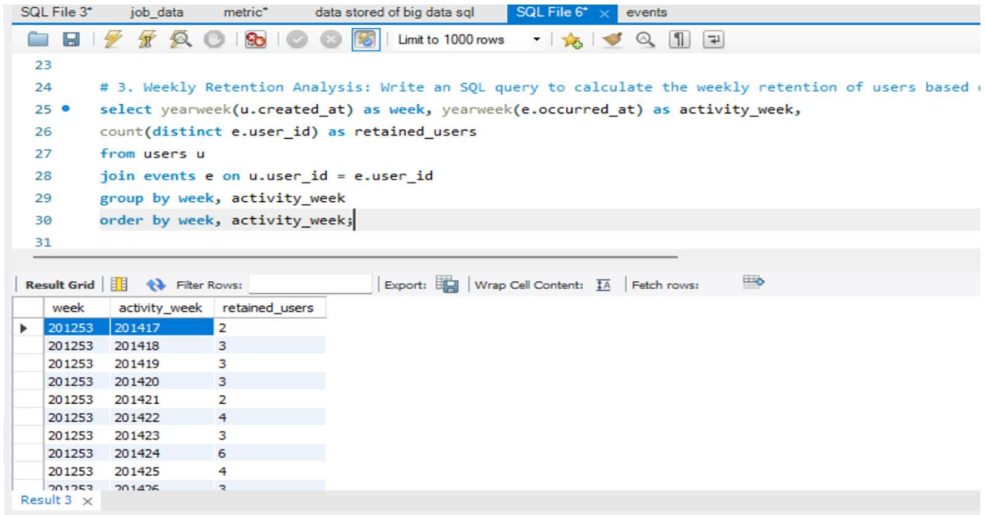
from users u

join events e on u.user_id = e.user_id

group by week, activity_week

order by week, activity_week;

Output:



Output:

week	activity_week	retained_users
201253	201417	2
201253	201418	3
201253	201419	3
201253	201420	3
201253	201421	2
201253	201422	4
201253	201423	3
201253	201424	6
201253	201425	4

Insights: It checks the query how many users return to the platform in weeks after signing up. If many users keep coming back, it shows good retention.

If retention falls rapidly, it may mean that users lose interest or face problems. Looking at various groups of users who join the same week helps us understand whether new users live longer or leave quickly.

D) **Weekly Engagement Per Device:** Write an SQL query to calculate the weekly engagement per device.

Code: select year(str_to_date(occurred_at, '%Y-%m-%d %H:%i:%s')) as year,

week(str_to_date(occurred_at, '%Y-%m-%d %H:%i:%s')) as week_number, device,

count(distinct user_id) as active_users

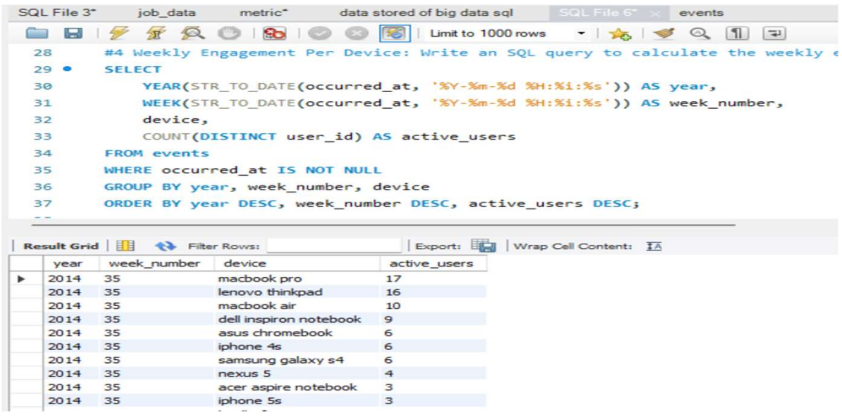
from events

where occurred_at is not null

group by year, week_number, device

order by year desc, week_number desc, active_users desc;

Result:



Output:

year	week_number	device	active_users
2014	35	macbook pro	17
2014	35	lenovo thinkpad	16
2014	35	macbook air	10
2014	35	dell inspiron notebook	9
2014	35	asus chromebook	6
2014	35	iphone 4s	6
2014	35	samsung galaxy s4	6
2014	35	nexus 5	4
2014	35	acer aspire notebook	3

Insights: This question measures how energetic customers are on exceptional gadgets every week. If greater users are lively on a certain device, it approch that tool is greater popular for engagement. A high range of cell customers shows the cellular revel in is ideal, even as a low range would possibly imply it needs development

E) **Email Engagement Analysis:** Write an SQL query to calculate the email engagement metrics.

Code: select date_format(str_to_date(occurred_at, '%d-%m-%Y %H:%i'), '%Y-%u') as week_number, action,

count(distinct user_id) as unique_users,

count(*) as total_events

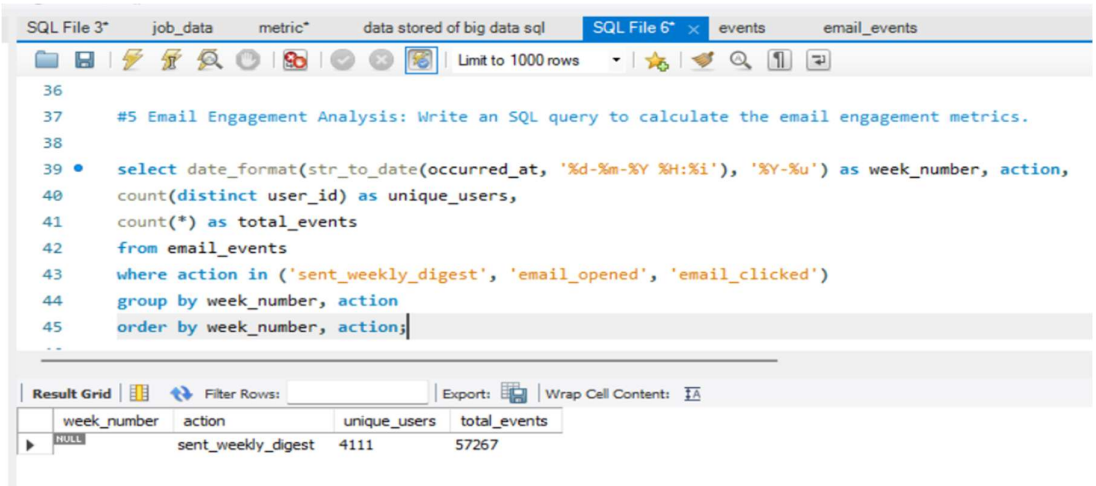
from email_events

where action in ('sent_weekly_digest', 'email_opened', 'email_clicked')

group by week_number, action

order by week_number, action;

Result:



Insights: It tracks the query how users interact with email, such as they open or click on them. If emails are low, it may mean that theme lines are not attractive**.**

Project Description

This project is about analyzing a company's operational data using SQL. It focuses on understanding user engagement, job reviews and e -post interactions to identify trends and areas of improvement. By examining important calculations such as job flow, user storage and e -post engagement, we aim to provide valuable insights that can help optimize business operations. The project also includes detecting anomalies, such as sudden spikes or drops in the user activity, and securing computer courses by identifying duplicate records.

Approach

we created a database in MySQL Workbench and imported the included CSV data sets. We examined the table structures and the ratio of different data points. Using SQL questions, we performed various analyzes, such as tracking job assessments over time, calculating storage speeds and measuring user engagement on a weekly basis. We used aggregations, joints, date features and rolling average to extract meaningful insights. Each query was designed to answer a specific business question, and helped us understand trends and patterns in the data.

Tech-Stack used

The project was carried out using the MySQL Workbench, which allowed us to write and run SQL questions effectively. We used CSV files as the primary data source, which was imported to MySQL for analysis. SQL techniques as a group of, join, count, avg, date features and rolling average were used to extract insight from the data. These techniques helped to segment the data, track trends and ensure accurate calculations.

Insights

From the analysis we observed different patterns in user behavior and job reviews. The flow analysis showed fluctuations in transit speed, with a few days experiencing higher activity than others. The user growth analysis revealed a steady increase in new users, with varying activation speed over time. Weekly engagement analysis indicated trends in user activity across different devices. In addition, the measurements highlighted the types of e -post messages that had higher open and click prices. Identifying duplicated rows helped improve data placement and reliability.

Result

This project helped us understand how users interact with the platform, how effectively work is reviewed, and how commitment patterns change over time. The findings can help the company improve user storage strategies, optimize job review processes and improve the marketing efforts for e -mail. By utilizing this insight, companies can make data -driven decisions to improve their general performance and operational efficiency.