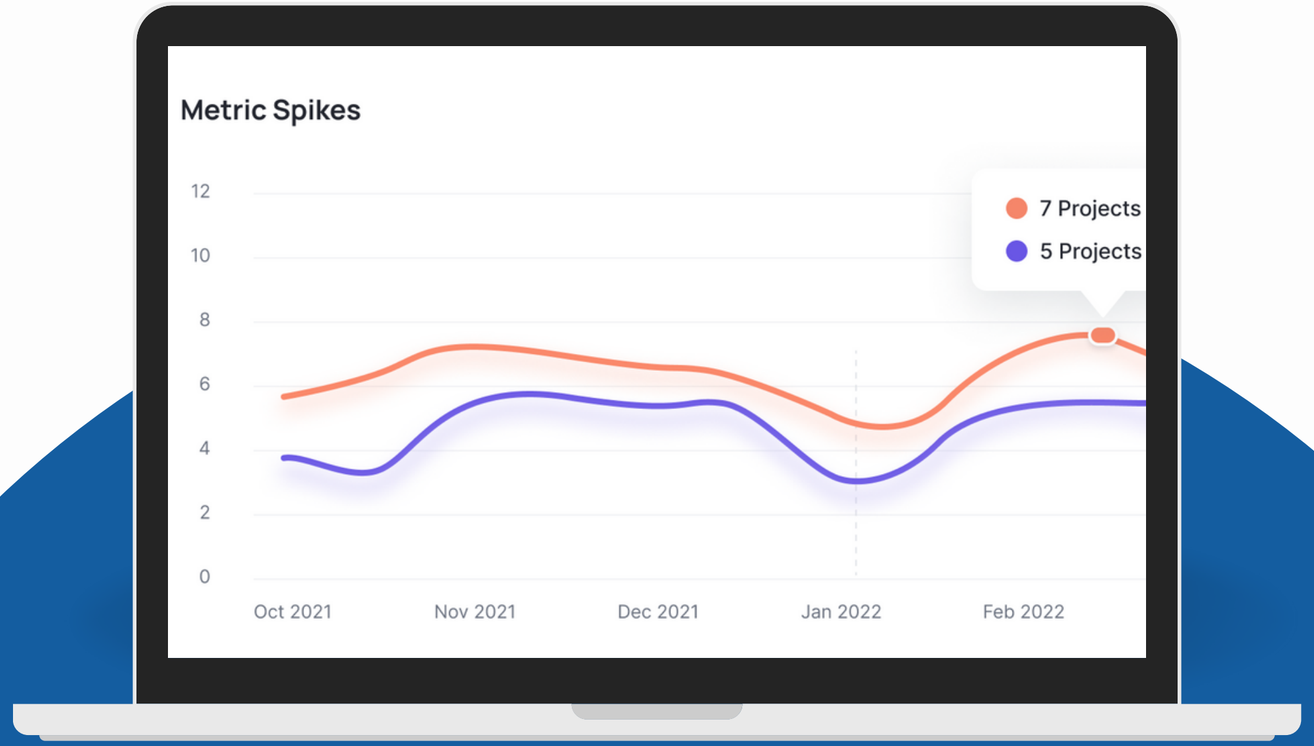
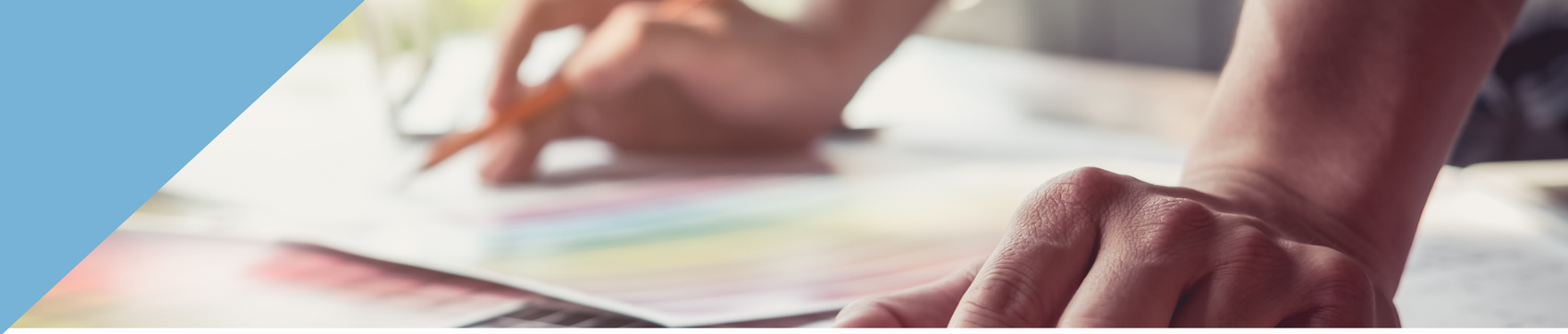
OPERATION ANALYTICS AND INVESTIGATING METRIC SPIKE

By : Iswariya S



Agenda



01 PROJECT DESCRIPTION

02 APPROACH

03 TECH-STACK USED

04 INSIGHTS

05 RESULT

06 CONCLUSION

PROJECT DESCRIPTION

The Operation Analytics and Investigating Metric Spike Data Analytics project is a strategic initiative designed to empower organizations with enhanced insights into their operational data. Focused on real-time monitoring and analysis, the project employs advanced data analytics techniques to detect and investigate metric spikes efficiently. By leveraging cutting-edge technologies and algorithms, the system aims to provide actionable intelligence for prompt decision-making. The project encompasses the development of a comprehensive analytics platform that integrates seamlessly with existing systems, ensuring a streamlined approach to data collection, processing, and interpretation. With a primary goal of optimizing operational performance, this project is poised to deliver valuable outcomes for businesses seeking to proactively address issues, improve efficiency, and make informed strategic decisions based on a thorough understanding of their metrics and data patterns.

APPROACH

The project employs advanced data analytics to monitor real-time operational metrics, detect irregularities, and derive insights for informed decision-making.

- **Established the database structure and tables**

Developed a database and subsequently constructed tables based on the provided structure and interconnections.

- **Conducted an in-depth analysis**

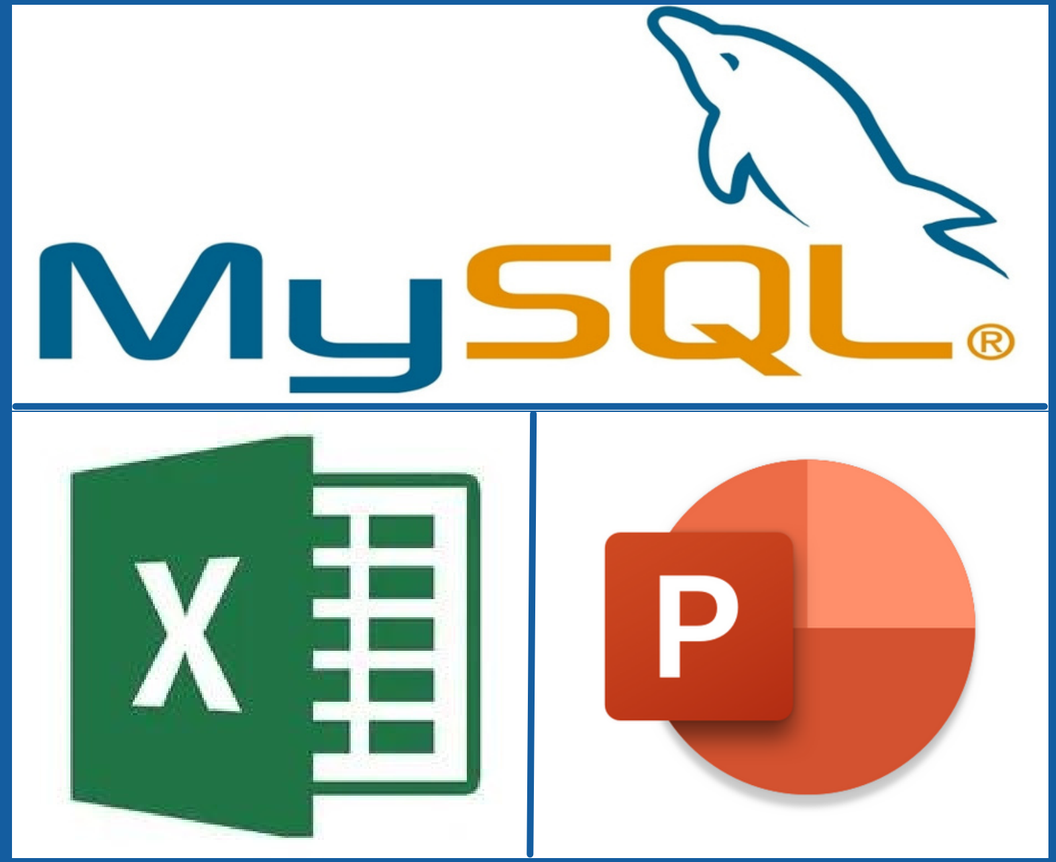
Employed SQL to thoroughly analyze the dataset, addressing the posed questions. After reviewing the provided dataset, tables were created to compute diverse queries.

- **Integrated Data for Informed Decision-Making**

Combined data fragments, organized the table structure for extracting business insights, retrieved the necessary outcomes, and thereby generated valuable insights for the company, enabling informed and strategic decision-making.

TECH-STACK USED

The project encompasses MySQL Community Server for database management, MS Excel for data analysis, and MS PowerPoint for effective insights presentation, providing a well-rounded toolkit for streamlined data analytics and decision-making.



1

Used MySQL Community server - (GPL Version 8.0.34 Connector Version C++ 8.0.34 For creating database)

2

Microsoft Excel - (For import and Export of data)

3

Microsoft PowerPoint - (For Presentation)

INSIGHTS AND KNOWLEDGE

Case Study 1: Job Data Analysis

1.Jobs Reviewed Over Time:
- Insights: Identify peak review periods.
- Knowledge: Forecast job review trends.

2.Throughput Analysis:
- Insights: Assess efficiency and detect bottlenecks.
- Knowledge: Improve process efficiency.

3.Language Share Analysis:
- Insights: Understand language distribution.
- Knowledge: Tailor services based on language popularity.

4.Duplicate Rows Detection:
- Insights: Ensure data integrity.
- Knowledge: Handle duplicate records for cleaner analysis.

Case Study 2: Investigating Metric Spike

1.Weekly User Engagement:
- Insights: Recognize user behavior patterns.
- Knowledge: Optimize engagement during peak periods.

2.User Growth Analysis:
- Insights: Track and understand user acquisition.
- Knowledge: Sustain and accelerate user growth.

3.Weekly Retention Analysis:
- Insights: Evaluate user retention factors.
- Knowledge: Improve retention through engagement.

4.Weekly Engagement Per Device:
- Insights: Analyze user behavior across devices.
- Knowledge: Optimize for specific platforms.

5.Email Engagement Analysis:
- Insights: Assess email communication effectiveness.
- Knowledge: Improve email engagement for better user interaction.

RESULT

Case Study 1: Job Data Analysis

A. Jobs Reviewed Over Time

SQL query to calculate the number of jobs reviewed per hour for each day in November 2020.

```
#TASK-1(Jobs Reviewed Over Time)
select * from job_data;
select avg(t) as 'avg jobs reviewed per day per hour',
avg(p) as 'avg jobs reviewed per day per second'
from
(select
ds,
((count(job_id)*3600)/sum(time_spent)) as t,
((count(job_id))/sum(time_spent)) as p
from
job_data
where
month(ds)=11
group by ds) a;
```

avg jobs reviewed per day per hour	avg jobs reviewed per day per second
126.18048333	0.03505000

B. Throughput Analysis

SQL query to calculate the 7-day rolling average of throughput (number of events per second).

- If the goal is to understand short-term variations and daily patterns, using the "Daily Throughput" metric is appropriate.
- If the goal is to identify broader trends and patterns while reducing the impact of daily noise, the "7-Day Rolling Average" is a better choice.

It's often beneficial to use both metrics in conjunction to gain a comprehensive understanding of throughput trends.

```sql
SELECT ROUND(COUNT(event)/SUM(time_spent), 2) AS "Weekly Throughput" FROM job_data;

SELECT ds AS Dates, ROUND(COUNT(event)/SUM(time_spent), 2) AS "Daily Throughput" FROM job_data
GROUP BY ds ORDER BY ds;
```

For Daily Throughput :

Dates	Daily Throughput
2020-11-25	0.02
2020-11-26	0.02
2020-11-27	0.01
2020-11-28	0.06
2020-11-29	0.05
2020-11-30	0.05

For Weekly Throughput :

Weekly Throughput
0.03

C. Language Share Analysis

SQL query to calculate the percentage share of each language over the last 30 days.

```sql
SELECT language AS Languages, ROUND(100 * COUNT(*)/TOTAL, 2) AS Percentage, sub.total
FROM job_data
CROSS JOIN (SELECT COUNT(*) AS total FROM job_data) AS sub
GROUP BY language, sub.total;
```

Languages	Percentage	total
English	12.50	16
Arabic	12.50	16
Persian	37.50	16
Hindi	12.50	16
French	12.50	16
Italian	12.50	16

D. Duplicate Rows Detection

SQL query to display duplicate rows from the job_data table.

```sql
SELECT actor_id, COUNT(*) AS Duplicates FROM job_data
GROUP BY actor_id HAVING COUNT(*) > 1;
```

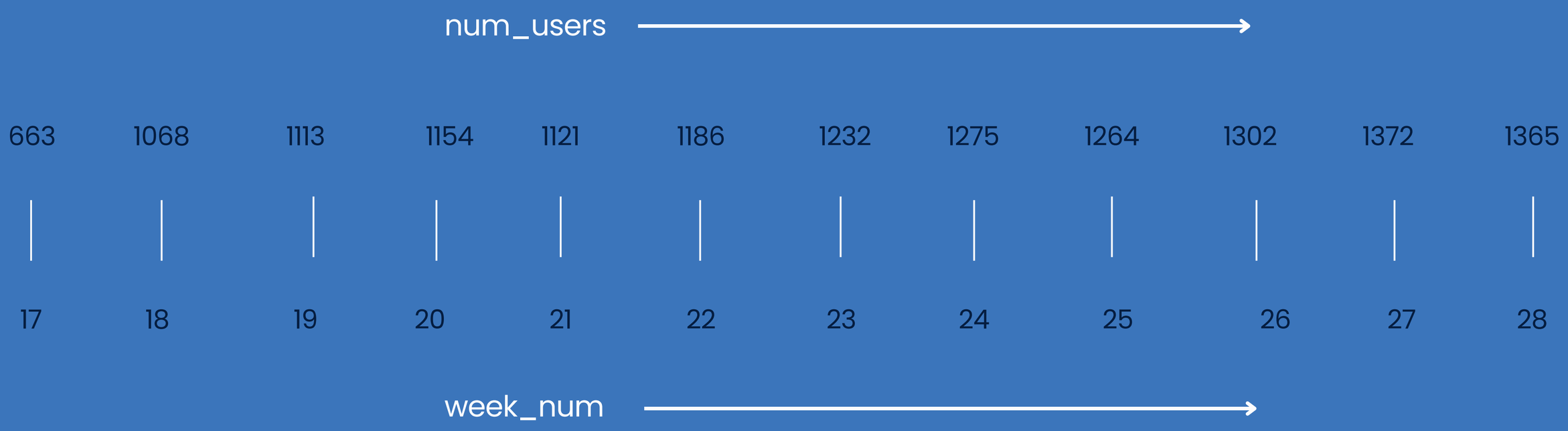
actor_id	Duplicates
1001	2
1006	2
1003	4
1005	2
1002	2
1007	2
1004	2

Case Study 2: Investigating Metric Spike

A. Weekly User Engagement

SQL query to calculate the weekly user engagement.

```sql
select extract(week from occured_at) as week_number,
count(distinct user_id) as active_user
from events_tb1
where event_type='engagement'
group by week_number
order by week_number
```



B. User Growth Analysis

SQL query to calculate the user growth for the product.

```
select year, week_num, num_users, sun(num_users)
 over (order by year, week_num) as cum_users
 from (
 select extract(year from created_at) as year, extract(week from created_at) as week_num, count(distinct user_id) as num_users
 from users_tb1
 where state='active'
 group by year, week_num
 order by year, week_num)sub
```

year	week_num	num_users	cum_users
2013	0	23	23
2013	1	30	53
2013	2	48	101
2013	3	36	137
2013	4	30	167
2013	5	48	215
2013	6	38	253
2013	7	42	295
2013	8	34	329
2013	9	43	372
2013	10	32	404
2013	11	31	435
2013	12	33	468

C. Weekly Retention Analysis

SQL query to calculate the weekly retention of users based on their sign-up cohort.

```sql
with ctel as (
select distinct user_id,
Extract (week from occured at) as signup_week
from events_tb1
where event_type = 'signup_flow"
and event_name = 'complete_signup and extract (week from occurred at) = 18 ),
cte2 as (select distinct user_id,
Extract (week from occurred at) as engagement_week
from events_tb1
where event_type = 'engagement')
select count(user_id) total_engaged_users,
sum(case when retension_week> 8 then 1 else end) as retained_users
from (select a.user_id, a.signup_week,
b.engagement_week, b.engagement_week-a.signup_week as retension_week
from ctel a
LEFT JOIN cte2 b
on a.user_id - b.user_id
order by a.user_id) sub
```

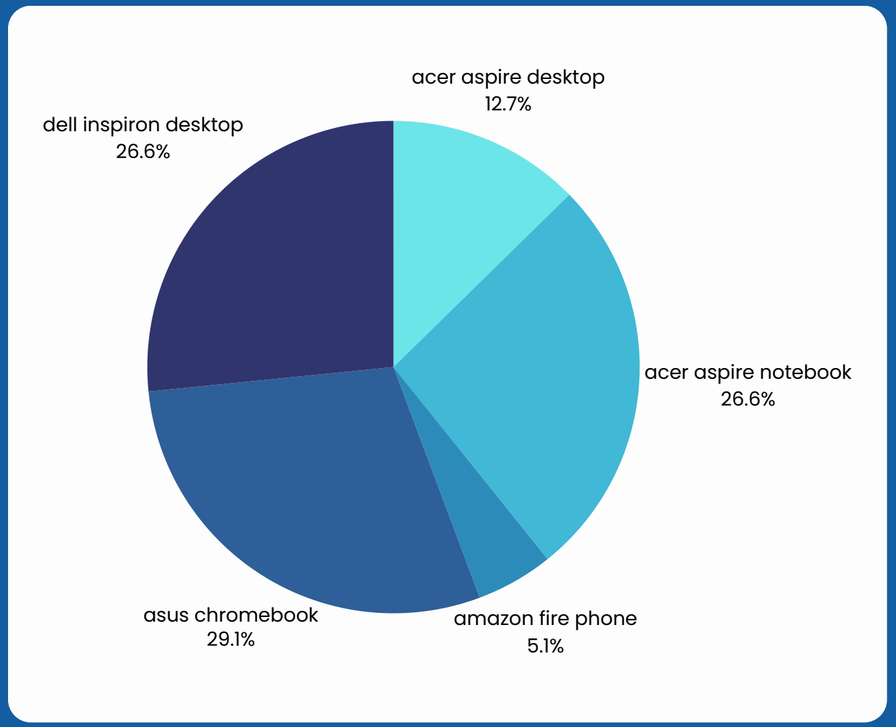
total_engaged_users	retained_users
317	236

D. Weekly Engagement Per Device

SQL query to calculate the weekly engagement per device.

```sql
with cte as (select extract (year from occurred_at)||'-'||extract(week from occurred_at) as weeknum
device, count(distinct user_id) as usercnt
from events_tb1
where event_type = 'engagement'
group by weeknum, device
order by weeknum)
select weeknum, device, usercnt
from cte
```

weeknum	device	usercnt
2014-18	acer aspire desktop	10
2014-18	acer aspire notebook	21
2014-18	amazon fire phone	4
2014-18	asus chromebook	23
2014-18	dell inspiron desktop	21



E. Email Engagement Analysis

SQL query to calculate the email engagement metrics.

```
select
100 * sum(case when email_cat = 'email_open' then 1 else 0 end)/
sum(case when email_cat = 'email sent' then 1 else 0 end) as email_open_rate,
100 * sum(case when email_cat = 'email clicked' then 1 else 0 end)/
sun(case when email_cat = 'email sent' then 1 else 0 end) as email_click_rate
from (select*,
case
when action in ('sent_weekly_digest', 'sent_reengagement_email') then 'email_sent'
when action in ('email_open') then 'email_open'
when action in ('email_clickthrough') then 'email_clicked'
end as email_cat
from email_events) sub
```

31.1921	10.4745
email_open_rate	email_click_rate

CONCLUSION

In this project, I learned advanced SQL techniques like Windows Functions, improved my SQL skills, and discovered how to ask the right questions for business growth. I gained insights into finding valuable information, understanding areas for improvement in companies, and investigating changes in metrics.