Operation Analytics and Investigating Metric Spike

Case Study 1: Job Data Analysis

Project Description:
This project is about analysing a job dataset containing records of organizational activities, that includes job events, language, time spent, and organizational units. In this project SQL is used to perform the analysis. The main goal of this project is to uncover the patterns and key operational metrics to support business decisions and optimize processes.

Approach:
Step 1: Imported the dataset from drive to MS-Excel and removed the empty records.
Step 2: Created a new database named operation_metrics_case1 and a table named job_data, as there were less records to analyse. I populated the dataset with more records.
Step 3: used SQL queries to answer all the defined business problems

Tech-Stack used:

- Software and Version: MySQL workbench 8.0.43.
- Why MySQL workbench: Provides a better interface to write, execute, and visualize SQL queries efficiently. Its tabular output and ease of use make query execution and result interpretation much simpler, improving overall productivity during analysis.

Tasks Performed:

a) Jobs Reviewed over Time:

1. Calculate the number of jobs reviewed per hour for each day in November 2020.

Query:

```sql
SELECT
    ds,
    COUNT(job_id) AS total_jobs,
    COUNT(job_id) / 24 AS job_per_hour
FROM
    job_data
WHERE
    ds BETWEEN '2020-11-01' AND '2020-11-30'
GROUP BY ds
ORDER BY ds;
```

Output:

| Result Grid | Filter Rows: | Export: | Wrap Cell Conte

ds	total_jobs	job_per_hour
2020-11-25	1	0.0417
2020-11-26	1	0.0417
2020-11-27	1	0.0417
2020-11-28	2	0.0833
2020-11-29	1	0.0417
2020-11-30	2	0.0833

b) Throughput Analysis:

1. Calculate the 7-day rolling average of throughput (number of events per second).

Query:

```sql
with throughput as (
select ds,count(event)/sum(time_spent) as event_per_second
from job_data
group by ds
)
select ds,event_per_second,avg(event_per_second)over( order by ds Rows between 6 preceding and current row)as `7-day_rolling_average`
from throughput
order by ds;
```

Output:

ds	event_per_second	7-day_rolling_average
2020-11-25	0.0222	0.02220000
2020-11-26	0.0179	0.02005000
2020-11-27	0.0096	0.01656667
2020-11-28	0.0606	0.02757500
2020-11-29	0.0500	0.03206000
2020-11-30	0.0500	0.03505000
2020-12-25	0.0200	0.03290000
2020-12-26	0.0164	0.03207143
2020-12-27	0.0092	0.03082857
2020-12-28	0.0465	0.03610000
2020-12-29	0.0400	0.03315714
2020-12-30	0.0400	0.03172857
2021-01-24	0.0182	0.02718571
2021-01-25	0.0152	0.02650000
2021-01-26	0.0088	0.02541429
2021-01-27	0.0377	0.02948571
2021-01-28	0.0333	0.02760000
2021-01-29	0.0333	0.02664286
2021-02-23	0.0167	0.02331429
2021-02-24	0.0141	0.02272857
2021-02-25	0.0084	0.02175714
2021-02-26	0.0317	0.02502857
2021-02-27	0.0286	0.02372857
2021-02-28	0.0286	0.02305714

c) Language Share Analysis:

1. Calculate the percentage share of each language in the last 30 days.

   Query:

```
select language ,count(*)*100.0/sum(count(*)) over() as percent_share_of_lang
from job_data
where ds>=date_sub((select max(ds)from job_data),interval 30 day)
group by language
order by percent_share_of_lang desc;
```

Output:

language	percent_share_of_lang
Arabic	42.85714
persian	28.57143
Hindi	21.42857
English	7.14286

d) Duplicate Rows Detection:

1. Identify duplicate rows in the data.

   Query:

```sql
select ds ,job_id,actor_id,event,language,time_spent,org
from job_data
group by ds ,job_id,actor_id,event,language,time_spent,org
having count(*)>1;
```

Output:

	ds	job_id	actor_id	event	language	time_spent	org

Insights:

a) Jobs Reviewed over Time

- In November 2020, job reviews per hour ranged between 0.0417 and 0.0833.
- Nov 28 and Nov 30 had the highest job review rate of 0.0833 jobs/hours.
- Other days maintained a rate of 0.0417 jobs/hour.

b) Throughput Analysis

- Daily throughput fluctuated between 0.0084 and 0.0606 events/sec and with peaks on Nov 28 ,2020(0.0606) and Dec 28,2020(0.0465).
- 7-day rolling average smoothed these spikes, averaging it around 0.02-0.03 events/sec.

c) Language Share Analysis:

- Arabic language dominated with 42.86% of total jobs in the last 30 days.
- Persian language accounted for 28.57% and Hindi contributed 21.43%.
- English had the lowest share at 7.14%, making it a minor area that could be focused on.

d) Duplicate Rows Detection:

- The query returned no results, indicating that there are no exact duplicate rows in the job_data table.
- However, there are individual duplicate values in some columns (example, multiple rows may share the same job_id and actor_id)

Results:

- Calculated job review rates per hour, identifying peak activity days in November 2020
- While solving the second question, I was not knowing what throughput was, then I developed a clear understanding of throughput and learned how to calculated it effectively. I manually practiced calculating throughput before implementing it in SQL, which strengthened my understanding of both the concept and its practical application.
- We can automate throughput monitoring by building a dashboard to track daily and rolling throughput trends in real-time, making performance issues easier to spot.
- We need to address language imbalance by allocating more resources to underrepresented languages to ensure better localization coverage.

Case Study 1: Investigating Metric Spike

Project Description:
The project focuses on analysing a dataset containing records of user activities that include user details, platform events, and email interactions. In this project SQL is used to perform the analysis. The main objective of this project is to identify patterns in user engagements, and highlight key operational metrics that can support data-driven decision making and improve organizational strategies.

Approach:
Step 1: Imported the dataset into MS-Excel from drive and removed the empty records.
Step 2: Saved the cleaned dataset into the directory Program data → MySQL → Uploads. This method allows importing large datasets directly without relying on the MYSQL import wizard.
Step 3: Created a database named operation_metrics_case_study2 and structured it with three tables named users, events, email_events.

Step 4: used SQL queries to answer all the defined business problems.

Tech-Stack used:

- Software and Version: MySQL workbench 8.0.43.
- Why MySQL workbench: Provides a better interface to write, execute, and visualize SQL queries efficiently. Its tabular output and ease of use make query execution and result interpretation much simpler, improving overall productivity during analysis.

Tasks Performed:

a) Weekly User Engagement:

1. Measure the activeness of users on a weekly basis.

   Query:

```sql
with user_events as (
select user_id ,occurred_at
from `events`
union
select user_id ,occurred_at
from email_events
)
select
extract(year from user_events.occurred_at )as year_,
extract(week from user_events.occurred_at )as week_,
count(distinct user_events.user_id)as active_users
from user_events
join users on user_events.user_id=users.user_id
group by week_,year_
order by week_,year_;
```

   Output:

Result Grid | Filter Rows:

year_	week_	active_users
2014	17	1370
2014	18	2921
2014	19	3013
2014	20	3085
2014	21	3149
2014	22	3266
2014	23	3354
2014	24	3467
2014	25	3565
2014	26	3657
2014	27	3772
2014	28	3884
2014	29	3990
2014	30	4122
2014	31	4189
2014	32	4315
2014	33	4470
2014	34	4580
2014	35	114

b) User Growth Analysis:

1. Analyse the growth of users over time for a product.
   Query:

```
select
   count(distinct user_id)total_users,
   year ( created_at)as year_,
   week(created_at,1)week_,
   sum(count(user_id))over(order by year(created_at),week(created_at,1))cumulative_sum_of_users_over_time
from users
group by year_,week_;
```

Output:

total_users	year_	week_	cumulative_sum_of_users_over_time
26	2013	1	26
29	2013	2	55
47	2013	3	102
36	2013	4	138
30	2013	5	168
48	2013	6	216
41	2013	7	257
39	2013	8	296
33	2013	9	329
43	2013	10	372
33	2013	11	405
32	2013	12	437
33	2013	13	470
40	2013	14	510
35	2013	15	545
42	2013	16	587
48	2013	17	635
48	2013	18	683
45	2013	19	728
55	2013	20	783
41	2013	21	824
49	2013	22	873
51	2013	23	924
51	2013	24	975
46	2013	25	1021
57	2013	26	1078
57	2013	27	1135
52	2013	28	1187
71	2013	29	1258
66	2013	30	1324
69	2013	31	1393
66	2013	32	1459
73	2013	33	1532
71	2013	34	1603
79	2013	35	1682
65	2013	36	1747
71	2013	37	1818
84	2013	38	1902

c) Weekly Retention Analysis:

1. Analyse the retention of users on a weekly basis after signing up for a product.

Query:

```
with user_signup as(
select users.user_id,
yearweek(users.created_at,1) signup_week
from users
),
user_activity as(
select `events`.user_id,
yearweek(`events`.occurred_at) activity_week
from `events`
)
select
user_signup.signup_week,
user_activity.activity_week,
count(distinct user_activity.user_id)as retained_user
from user_signup
join user_activity
on user_signup.user_id=user_activity.user_id
where user_activity.activity_week>=user_signup.signup_week
group by user_signup.signup_week,user_activity.activity_week
order by retained_user ;
```

Output:

signup_week	activity_week	retained_user
201320	201421	6
201320	201422	6
201320	201424	6
201320	201427	6
201320	201430	6
201320	201431	6
201320	201432	6
201323	201431	6
201324	201419	6
201324	201420	6
201324	201422	6
201324	201434	6
201325	201424	6
201325	201425	6
201326	201425	6
201326	201426	6
201327	201421	6
201327	201423	6
201327	201427	6
201327	201428	6
201328	201420	6
201328	201427	6
201328	201429	6
201329	201417	6
201329	201423	6
201329	201430	6
201331	201421	6
201331	201429	6
201332	201422	6
201332	201431	6
201333	201423	6
201333	201428	6
201334	201417	6
201335	201432	6
201336	201425	6
201336	201429	6
201336	201431	6

- Note: The full SQL output of the above query is large and not included in this document. A sample of the first few rows is shown for reference. A total of 1410 rows have been returned.

d) Weekly Engagement Per Device:

1. Measure the activeness of users on a weekly basis per device.

   Query:

```
with user_activity as (
select user_id,occurred_at,device
from `events`
union
select user_id,occurred_at,'email'as device
from email_events
)
select device,
year(occurred_at)as year_,
week(occurred_at,1)as week_,
count( distinct user_id) as active_users
from user_activity
group by week_,year_,device
order by active_users desc;
```

   Output:

device	year_	week_	active_users
email	2014	35	4309
email	2014	34	4209
email	2014	33	4061
email	2014	32	3953
email	2014	31	3883
email	2014	30	3748
email	2014	29	3675
email	2014	28	3557
email	2014	27	3461
email	2014	26	3340
email	2014	25	3272
email	2014	24	3143
email	2014	23	3047
email	2014	22	2945
email	2014	21	2876
email	2014	20	2801
email	2014	19	2724
email	2014	18	1006
macbook pro	2014	31	317
macbook pro	2014	32	317
macbook pro	2014	34	308
macbook pro	2014	33	307
macbook pro	2014	28	301
macbook pro	2014	29	295
macbook pro	2014	30	291
macbook pro	2014	35	290

- Note: The full SQL output of the above query is large and not included in this document. A sample of the first few rows is shown for reference. A total of 486 rows have been returned.

e) Email Engagement Analysis:

1. Analyse how users are engaging with the email service.
   Query:

```
SELECT
   YEAR(occurred_at) AS year_,
   WEEK(occurred_at) AS week_,
   SUM(CASE
       WHEN action = 'sent_weekly_digest' THEN 1
       ELSE 0
   END) total_email_sent,
   SUM(CASE
       WHEN action = 'email_open' THEN 1
       ELSE 0
   END) total_email_opened,
   SUM(CASE
       WHEN action = 'email_clickthrough' THEN 1
       ELSE 0
   END) total_email_clickthroughs,
   SUM(CASE
       WHEN action = 'email_open' THEN 1
       ELSE 0
   END) / COUNT(*) * 100 email_open_rate,
   SUM(CASE
       WHEN action = 'email_clickthrough' THEN 1
       ELSE 0
   END) / COUNT(*) * 100 email_clickthrough_rate
FROM
   email_events
GROUP BY year_ , week_
ORDER BY year_ , week_;
```

Output:

year_	week_	total_email_sent	total_email_opened	total_email_clickthroughs	email_open_rate	email_clickthrough_rate
2014	17	908	310	166	21.2766	11.3933
2014	18	2602	912	430	22.2385	10.4852
2014	19	2665	972	477	22.6732	11.1267
2014	20	2733	1004	507	22.6381	11.4318
2014	21	2822	1014	443	22.8224	9.9707
2014	22	2911	987	488	21.5596	10.6597
2014	23	3003	1075	538	22.3353	11.1781
2014	24	3105	1155	554	22.9167	10.9921
2014	25	3207	1096	530	21.7936	10.5389
2014	26	3302	1165	556	22.2243	10.6066
2014	27	3399	1228	621	22.4867	11.3715
2014	28	3499	1250	599	22.4780	10.7714
2014	29	3592	1219	590	21.7136	10.5094
2014	30	3706	1383	630	23.2437	10.5882
2014	31	3793	1351	445	23.2490	7.6579
2014	32	3897	1337	418	22.8469	7.1429
2014	33	4012	1432	490	23.1042	7.9058
2014	34	4111	1528	490	23.9124	7.6682
2014	35	0	41	38	32.2835	29.9213

Insights:

a) Weekly User Engagement:

- User engagement started at 1,370 active users (week 17,2014).
- It showed a steady growth trend over the weeks, crossing 3,000 + users by week 19 and reaching over 4500 users by week 34.
- The highest engagement was observed in week 34 with 4,580 active users and there was a drop in week 35(114 users), this may incomplete data or an anomaly.

b) User Growth Analysis:

- The platform started with 26 users in week 1(2013).
- By week 10 (2013), the cumulative user base had grown to 372 users.
- The growth trend continued strongly, with cumulative users surpassing 1,000 by week 25 and reaching 1,902 by week 38.
- This shows consistent week-on-week growth, with no major decline in sign-ups.

c) Weekly Retention Analysis:

- Retention was tracked by comparing signup week and activity week.

- Retention counts increased significantly from week 25(2014), primarily due to larger signup cohorts.
- While early cohorts had only 6-20 retained users.
- This indicated that as the user base grew, the absolute number of retained users also grew.

d) Weekly Engagement Per Device:

- Email is the leading engagement channel with consistent week-on-week growth from 1,006(week 18) to 4,309 users (week 35).
- Engagement via email grew more than 4x in just 17 weeks.
- Other devices (example., MacBook Pro, Lenovo, etc) show much lower engagement.
- This suggest that email should remain as the primary focus for driving engagement.

e) Email Engagement Analysis:

- Email volume grew from 908(week 17) to 411(week 34)
- Open rates stayed around 21-23% (peak :23.9% in week 34)
- Clickthrough rate dropped from 11.4% (week 17) to 7.7% (week 34).
- Week 35 shows very high open (32.3%) and clickthrough (29.9%) rates but with 0 email sent → possible anomaly.

Result:

- I was able analyse the weekly users, weekly growth, cumulative users.
- While working on retention analysis, I learned the importance of cohort-based retention percentages.
- While Email Engagement Analysis I learned about clickthrough rates, open rates and how to calculate it.
- we need investigate the week 35 anomaly(sudden drop to 114 users)to check if its due to data gap or real behaviour.