Operation Analytics and Investigating Metric Spike

Project Description:

The project is about Operation Analytics and Investigating Metric Spike, where Operation Analytics is crucial process that involves analyzing a company's end-to-end operations. This analysis helps identify areas for improvement within the company. As a Data Analyst, we will work closely with various teams, such as operations, support, and marketing, helping them derive valuable insights from the data they collect.

The main goal is to use advanced SQL skills to analyze the data and provide valuable insights that can help improve the company's operations and understand sudden changes in key metrics.

Also it is provided with various datasets and tables, and the task will be to derive insights from this data to answer questions posed by different teams within the company per below.

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

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

3) Find out the the percentage share of each language in the last 30 days.

4) Identify duplicate rows in the data.

5) Measure the activeness of users on a weekly basis.

6) Analyze the growth of users over time for a product.

7) Analyze the retention of users on a weekly basis after signing up for a product.

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

9) Analyze how users are engaging with the email service.

Approach:

At first the data provided will need to store in database. So I used MYSQL Workbench, where I created database and loaded that data into database. Then I started analysing the data with all tables one-by-one, where I got clear understanding of dataset like Daily Active Users, Metrics, Throughput Analysis, 7-day rolling average etc. This involves understanding and explaining sudden changes in key metrics, such as a dip in daily user engagement or a drop in sales.

When it comes to perform the tasks, tried to understand these Key metrics, user engagement so that I can write queries accordingly to get desired outputs for all the tasks.

Tech-Stack Used:

I used MySQL Workbench-Version 8.0.34 software. It is hassle-free and easy to use RDBMS software. And queries were executed correctly. Also whenever got error it was clearly provided details of error with error code where I tried to diagnose the errors and corrected the queries and executed them.

Insights:

Working on Operation Analytics and Investigating Metric Spike helps me to understand the role of Data Analytics.

How Data Analyst has to provide valuable insights that can help improve the company's operations by analyzing company's end to end operations and identify the areas of improvement within the company for the growth of the company.

Result:

The project helps me to understand the new definitions like Throughput Analysis, 7-day rolling average, Daily Metrics. Also get familiar with new keywords like week, day, month etc.

As a result it leads me to one step ahead to understand the Analyst role.

Tasks:

A) Jobs Reviewed Over Time:

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

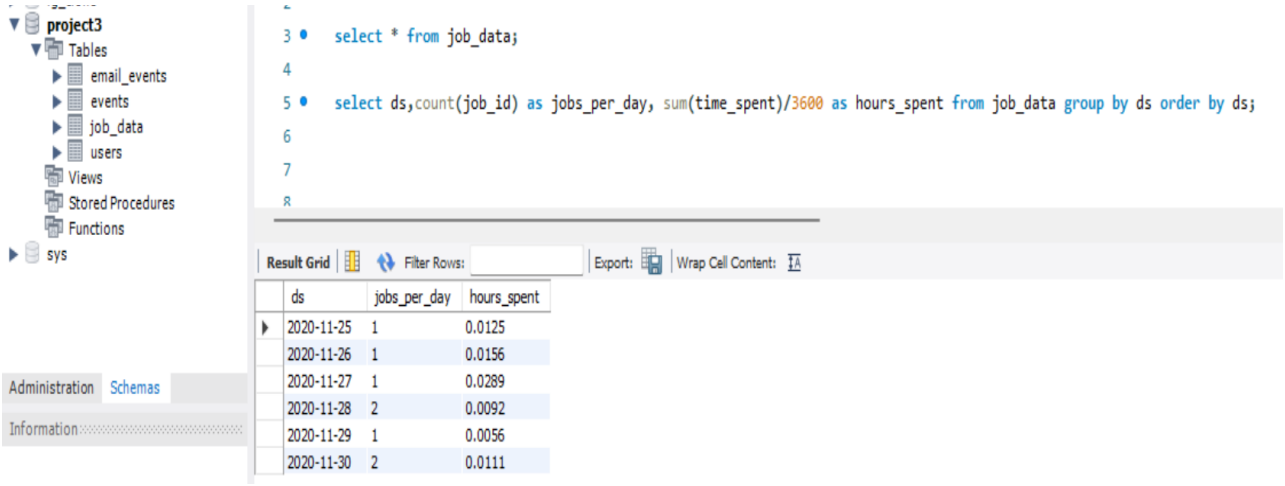
Your Task: Write an SQL query to calculate the number of jobs reviewed per hour for each day in November 2020.

Query:

```
select ds,count(job_id) as jobs_per_day, sum(time_spent)/3600 as hours_spent from job_data group by ds order by ds;
```

Output:

ds	jobs_per_day	hours_spent
2020-11-25	1	0.0125
2020-11-26	1	0.0156
2020-11-27	1	0.0289
2020-11-28	2	0.0092
2020-11-29	1	0.0056
2020-11-30	2	0.0111



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.

Throuhput - If the density of the data is larger, we use the daily metric, and if the density is low the 7-day rolling works well. It also depends upon the anomaly in the data set. Because in 7 days metric the view is broader similar to the daily metric view becomes narrower.

Query:

```sql
WITH CTE AS (
SELECT ds, COUNT(job_id) AS jobs, SUM(time_spent) AS times
FROM job_data GROUP BY ds)
SELECT ds, SUM(jobs) OVER (ORDER BY ds ROWS BETWEEN 6 PRECEDING AND CURRENT ROW) / SUM(times)
OVER (ORDER BY ds ROWS BETWEEN 6 PRECEDING AND CURRENT ROW) AS throughput_7d_rolling_avg FROM CTE;
```

Output:

ds	throughput_7d_rolling_avg
2020-11-25	0.0222
2020-11-26	0.0198
2020-11-27	0.0146
2020-11-28	0.0210
2020-11-29	0.0233
2020-11-30	0.0268

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.

Query:

```sql
select language, (count(*) * 100 / (select COUNT(*) from job_data)) as percentage_share
from job_data group by language order by language desc;
```

Output:

language	percentage_share
Persian	37.5000
Italian	12.5000
Hindi	12.5000
French	12.5000
English	12.5000
Arabic	12.5000

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.

Query:

```
select job_id, actor_id,event,language, row_number() over(partition by  job_id,language order by actor_id) as duplicate_rows
from job_data;
```

Output:

job_id	actor_id	event	language	duplicate_rows
11	1007	decision	French	1
20	1003	transfer	Italian	1
21	1001	skip	English	1
22	1006	transfer	Arabic	1
23	1003	decision	Persian	1
23	1004	skip	Persian	2
23	1005	transfer	Persian	3
25	1002	decision	Hindi	1

Case Study 2: Investigating Metric Spike:

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.

Query:

```
select count(distinct user_id) as users, week(occurred_at) as weeks from events group by 2;
```

Output:

	users	weeks
▶	663	17
	1068	18
	1113	19
	1154	20
	1121	21
	1186	22
	1232	23
	1275	24
	1264	25
	1302	26
	1372	27
	1365	28
	1376	29
	1467	30
	1299	31
	1225	32
	1225	33
	1204	34
	104	35

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.

Query:

```
with cte as (
select week(activated_at) as week, year(activated_at) as year,
count(distinct user_id) as active_users
from users
group by week,year)
select week, year, active_users,
sum(active_users) over (order by year,week rows between unbounded preceding and current row) as user_growth
from cte
group by week,year
order by year,week;
```

Output:

week	year	active_users	user_growth
0	2013	23	23
1	2013	30	53
2	2013	48	101
3	2013	36	137
4	2013	30	167
5	2013	48	215
6	2013	38	253
7	2013	42	295
8	2013	34	329
9	2013	43	372
10	2013	32	404
11	2013	31	435
12	2013	33	468
13	2013	39	507
14	2013	35	542
15	2013	43	585
16	2013	46	631
17	2013	49	680
18	2013	44	724
19	2013	57	781
20	2013	39	820
21	2013	49	869
22	2013	54	923
23	2013	50	973
12	2014	148	4931
13	2014	167	5098
14	2014	162	5260
15	2014	164	5424
16	2014	179	5603
17	2014	170	5773
18	2014	163	5936
19	2014	185	6121
20	2014	176	6297
21	2014	183	6480
22	2014	196	6676
23	2014	196	6872
24	2014	229	7101
25	2014	207	7308
26	2014	201	7509
27	2014	222	7731
28	2014	215	7946
29	2014	221	8167
30	2014	238	8405
31	2014	193	8598
32	2014	245	8843
33	2014	261	9104
34	2014	259	9363
35	2014	18	9381

C. Weekly Retention Analysis:

o 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.

Query:

```sql
select Weeks, sum(case when week_num = 0 then 1 else 0 end) as week_0,
sum(case when week_num = 1 then 1 else 0 end) as week_1,
sum(case when week_num = 2 then 1 else 0 end) as week_2,
sum(case when week_num = 3 then 1 else 0 end) as week_3,
sum(case when week_num = 4 then 1 else 0 end) as week_4,
sum(case when week_num = 5 then 1 else 0 end) as week_5,
sum(case when week_num = 6 then 1 else 0 end) as week_6,
sum(case when week_num = 7 then 1 else 0 end) as week_7,
sum(case when week_num = 8 then 1 else 0 end) as week_8,
sum(case when week_num = 9 then 1 else 0 end) as week_9,
sum(case when week_num = 10 then 1 else 0 end) as week_10,
sum(case when week_num = 11 then 1 else 0 end) as week_11,
sum(case when week_num = 12 then 1 else 0 end) as week_12
from (select a.user_id, week, weeks, (week-weeks) as week_num from
(select user_id, week(occurred_at) as week from events group by 1,2) a,
(select user_id, min(week(occurred_at)) as weeks from events group by user_id)b
where a.user_id = b.user_id ) as week_with_num
group by weeks
order by weeks;
```

Output:

weeks	week_0	week_1	week_2	week_3	week_4	week_5	week_6	week_7	week_8	week_9	week_10	week_11	week_12
17	663	472	324	251	205	187	167	146	145	145	136	131	132
18	596	362	261	203	168	147	144	127	113	122	106	118	127
19	427	284	173	153	114	95	91	81	95	82	68	65	63
20	358	223	165	121	91	72	63	67	63	65	67	41	40
21	317	187	131	91	74	63	75	72	58	48	45	39	35
22	326	224	150	107	87	73	63	60	55	48	41	39	31
23	328	219	138	101	90	79	69	61	54	47	35	30	0
24	339	205	143	102	81	63	65	61	38	39	29	0	0
25	305	218	139	101	75	63	50	46	38	35	2	0	0
26	288	181	114	83	73	55	47	43	29	0	0	0	0
27	292	199	121	106	68	53	40	36	1	0	0	0	0
28	274	194	114	69	46	30	28	3	0	0	0	0	0
29	270	186	102	65	47	40	1	0	0	0	0	0	0
30	294	202	121	78	53	3	0	0	0	0	0	0	0
31	215	145	76	57	1	0	0	0	0	0	0	0	0
32	267	188	94	8	0	0	0	0	0	0	0	0	0
33	286	202	9	0	0	0	0	0	0	0	0	0	0
34	279	44	0	0	0	0	0	0	0	0	0	0	0
35	18	0	0	0	0	0	0	0	0	0	0	0	0

D. Weekly Engagement Per Device:

o Objective: Measure the activeness of users on a weekly basis per device.

Your Task: Write an SQL query to calculate the weekly engagement per device.

Query:

```
select week(occurred_at) as weeks, device, count(distinct user_id) as count
from events
where event_type = 'engagement'
group by 1,2;
```

Output:

weeks	device	count
17	acer aspire desktop	9
17	acer aspire notebook	20
17	amazon fire phone	4
17	asus chromebook	21
17	dell inspiron desktop	18
17	dell inspiron notebook	46
17	hp pavilion desktop	14
17	htc one	16
17	ipad air	27
17	ipad mini	19
17	iphone 4s	21
17	iphone 5	65
17	iphone 5s	42
17	kindle fire	6
17	lenovo thinkpad	86
17	mac mini	6
17	macbook air	54
17	macbook pro	143
17	nexus 10	16
35	hp pavilion desktop	1
35	htc one	2
35	ipad mini	2
35	iphone 4s	6
35	iphone 5	2
35	iphone 5s	3
35	kindle fire	3
35	lenovo thinkpad	16
35	mac mini	2
35	macbook air	10
35	macbook pro	17
35	nexus 10	2
35	nexus 5	4
35	nexus 7	2
35	nokia lumia 635	2
35	samsung galaxy note	1
35	samsung galaxy s4	6
35	windows surface	3
27	ipad mini	35
27	iphone 4s	67
27	iphone 5	163
27	iphone 5s	83
27	kindle fire	25
27	lenovo thinkpad	202
27	mac mini	15
27	macbook air	142
27	macbook pro	302
27	nexus 10	37
27	nexus 5	84
27	nexus 7	40
27	nokia lumia 635	31
27	samsumg galaxy tablet	15
27	samsung galaxy note	15
27	samsung galaxy s4	116
27	windows surface	33
28	acer aspire desktop	30
28	acer aspire notebook	49

E. Email Engagement Analysis:

o Objective: Analyze how users are engaging with the email service.

Your Task: Write an SQL query to calculate the email engagement metrics.

Query:

```
select sum(case when actions = 'email_open' then 1 else 0 end)/sum(case when actions = 'delivered_emails' then 1 else 0 end)*100 as email_open_rate,
sum(case when actions = 'total_measured_clicks' then 1 else 0 end)/sum(case when actions = 'delivered_emails' then 1 else 0 end)*100 as email_clickthrough_rate
from ( select
    case
    when action In ('sent_weekly_digest', 'sent_reengagement_email') then 'delivered_emails'
    when action in ('email_open') then 'email_open'
    when action in ('email_clickthrough') then 'total_measured_clicks'
    end as actions
    from email_events) as email_metrics;
```

Output:

	email_open_rate	email_clickthrough_rate
▶	33.5834	14.7899