Project Description:

Operation Analytics is the analysis done for the complete end to end operations of a company. With the help of this, the company then finds the areas on which it must improve upon. You work closely with the ops team, support team, marketing team, etc and help them derive insights out of the data they collect.

Being one of the most important parts of a company, this kind of analysis is further used to predict the overall growth or decline of a company's fortune. It means better automation, better understanding between cross-functional teams, and more effective workflows.

Investigating metric spike is also an important part of operation analytics as being a Data Analyst you must be able to understand or make other teams understand questions like- Why is there a dip in daily engagement? Why have sales taken a dip? Etc. Questions like these must be answered daily and for that its very important to investigate metric spike.

Approach

First, I examined the data set, including all of the tables and columns, to gain a sense of the information accessible to me. Then I examined and comprehended all of the case studies and summarized my method to solving these case studies. Then I compose the query.

Tech-Stack Used

MySQL Workbench- Version - 8.0.32

MySQL is a popular open-source relational database management system. The purpose of using MySQL is to store, organize, and manage large amounts of data efficiently and effectively

Insights-

- **Throughput:** It is the no. of events happening per second.

Your task: Let's say the above metric is called throughput. Calculate 7 day rolling average of throughput? For throughput, do you prefer daily metric or 7-day rolling and why?

Result-

```
9
10  SELECT ds, AVG(throughput) OVER (ORDER BY ds ROWS BETWEEN 6 PRECEDING AND CURRENT ROW) AS rolling_avg
11  FROM (
12    SELECT ds, COUNT(DISTINCT job_id) AS throughput
13    FROM job.jp
14    GROUP BY ds
15  ) AS throughput_by_day
16  ORDER BY ds
```

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

ds	rolling_avg
25-11-2020	1.0000
26-11-2020	1.0000
27-11-2020	1.0000
28-11-2020	1.2500
29-11-2020	1.2000
30-11-2020	1.3333

Insights-

- **Number of jobs reviewed:** Amount of jobs reviewed over time.

Your task: Calculate the number of jobs reviewed per hour per day for November 2020?

Result-

```
13  SELECT COUNT(DISTINCT job_id)/(24*3600) AS Rev
14  FROM job.JP
15  WHERE
16  EXTRACT(MONTH FROM ds) = 11;
```

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

Rev
0.0000

Insights-

- **Percentage share of each language:** Share of each language for different contents.

Your task: Calculate the percentage share of each language in the last 30 days

Result-

```
12 •  SELECT language, COUNT(*) * 100.0 / (SELECT COUNT(*) FROM job.jp) AS percentage_share
13    FROM job.jp
14    GROUP BY language;
15
16
```

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

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

Insights-

- **Duplicate rows:** Rows that have the same value present in them.

Your task: Let's say you see some duplicate rows in the data. How will you display duplicates from the table?

Result-

```
13 •  SELECT ds, job_id, actor_id, event, language, time_spent, org, COUNT(*) as duplicate
14    FROM job.jp
15    GROUP BY ds, job_id, actor_id, event, language, time_spent, org
16    HAVING COUNT(*) > 1;
```

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

ds	job_id	actor_id	event	language	time_spent	org	duplicate

CASE STUDY-2

Insights-

1. **User Engagement:** To measure the activeness of a user. Measuring if the user finds quality in a product/service.

Your task: Calculate the weekly user engagement?

```
41 •  SELECT COUNT(DISTINCT user_id) AS users, week
42    FROM(
43    SELECT user_id, EXTRACT(week FROM occurred_at) AS week
44    FROM job2.e
45    WHERE event_type = 'engagement') AS a
46    GROUP BY week;
```

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

users	week
578	NULL

Insights-

2. **User Growth:** Amount of users growing over time for a product.

Your task: Calculate the user growth for product?

```
49 •  SELECT Year, Week, u_active,
50    SUM(u_active) OVER(ORDER BY year, week
51    ROWS BETWEEN UNBOUNDED PRECEDING AND CURRENT ROW) AS
52    u_growth
53    FROM(
54    SELECT
55    EXTRACT(year FROM activated_at) AS year,
56    EXTRACT(week FROM activated_at) AS week,
57    COUNT(DISTINCT s) AS u_active
58    FROM job2.u
59    WHERE state = 'active'
60    GROUP BY week, year
61    ORDER BY year, week
62    ) AS d;
63
```

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

Year	Week	u_active	u_growth
NULL	NULL	9381	9381

Insights-

3. **Weekly Retention:** Users getting retained weekly after signing-up for a product.
Your task: Calculate the weekly retention of users-sign up cohort?

```sql
SELECT user_id,
SUM(CASE WHEN retained_week >= 1 THEN 1 ELSE 0 END) AS weekly_retention
FROM
(SELECT
c.user_id,
c.signup_week,
e.engagement_week,
e.engagement_week - c.signup_week AS retained_week
FROM
(SELECT
user_id,
EXTRACT(week FROM occurred_at) AS signup_week
FROM job2.e
WHERE event_name = 'complete_signup') AS c
INNER JOIN
(SELECT
```

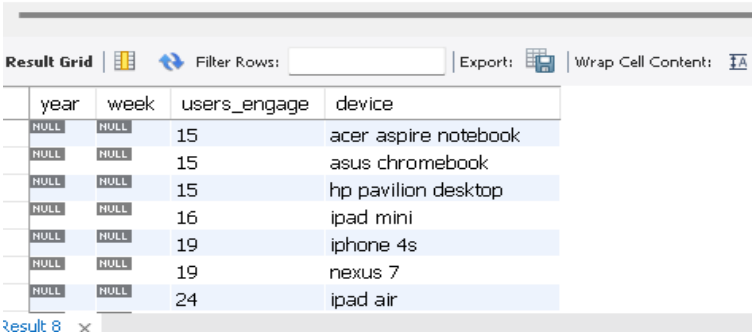
Result Grid | Filter Rows: | Export: | Wrap Cell Content:

user_id	weekly_retention
12865	0
12867	0
12872	0
12874	0
12875	0
12879	0
12880	0

Insights-

4. **Weekly Engagement:** To measure the activeness of a user. Measuring if the user finds quality in a product/service weekly.
Your task: Calculate the weekly engagement per device?

```sql
SELECT
EXTRACT(year FROM occurred_at) AS year,
EXTRACT(week FROM occurred_at) AS week,
COUNT(DISTINCT user_id) AS users_engage,
device
FROM job2.e
GROUP BY 1,2,4
ORDER BY 1,2,3;
```



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

year	week	users_engage	device
NULL	NULL	15	acer aspire notebook
NULL	NULL	15	asus chromebook
NULL	NULL	15	hp pavilion desktop
NULL	NULL	16	ipad mini
NULL	NULL	19	iphone 4s
NULL	NULL	19	nexus 7
NULL	NULL	24	ipad air

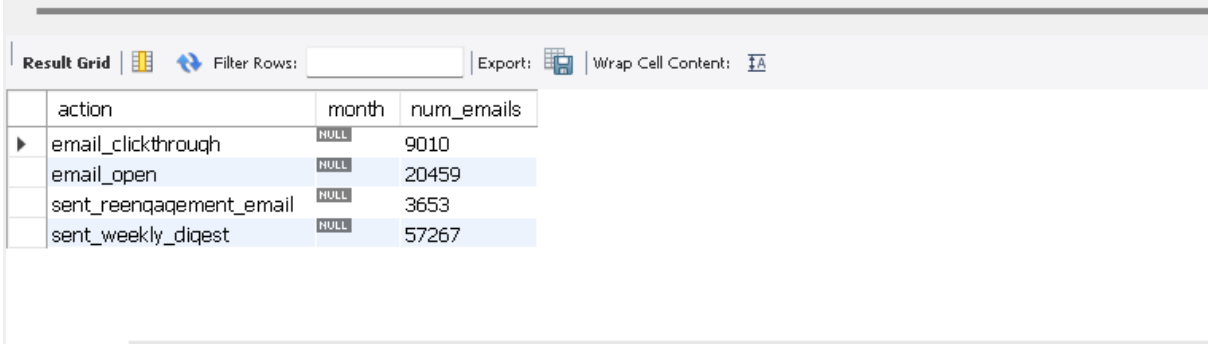
Result 8

Insights-

5. **Email Engagement:** Users engaging with the email service.

Your task: Calculate the email engagement metrics?

```sql
SELECT action, EXTRACT(MONTH FROM occurred_at) AS month, count(action) as num_emails
FROM job2.ee
GROUP BY action, month
ORDER BY action, month
```



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

action	month	num_emails
email_clickthrough	NULL	9010
email_open	NULL	20459
sent_reengagement_email	NULL	3653
sent_weekly_digest	NULL	57267