# Operation Analytics and Investigating Metric Spike

## DESCRIPTION:

Operation Analytics is the analysis done for the complete end-to-end operations of a company. With the help of this analysis, the company finds the areas in which it must do something to improve. In this project, I will work closely with the ops team, support team, marketing team, etc., and help them derive insights from the data they collect. As one of the most important parts of a company, this 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 spikes is also an important part of operation analytics as Data Analysts. I must 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, so it is very important to investigate metric spikes.

## Project Approach Used:

According to the project's scenario, I am working for a company like Microsoft, designated as Data Analyst Lead, and provided with different data sets and tables from which I must derive certain insights and answer the questions asked by different departments. This project is completely based on Structured Query Language (SQL). To create a complete database containing different tables, SQL queries were provided. To answer all the questions asked, I used SQL Basic to Advanced functions that generated useful insights. This report gives a better understanding of that in the 'insight.' segment.

## Tech-Stack Used:

- MySQL Workbench with the latest released Version 8.0 CE (Community Edition) by Oracle Corporation – For Importing Database, Running SQL Queries to get insights
- Excel 2016 by Microsoft Corporation – For extracting & manipulating data
- Word 2016 by Microsoft Corporation – For creating a project report

I used it because MySQL Workbench is a visual database design tool that combines SQL development, administration, database design, creation, and maintenance into a single development environment.

# Project Insights

## I. Case Study 1 (Job Data)

### A. Number of jobs reviewed:

Here, I need to calculate the number of jobs reviewed per hour per day for November 2020 from the given database.

For fulfilling the required query, I selected ds and used the COUNT function on job_id and named that column jobs_per_day. Then I used the SUM function on the total time spent (I assumed that the time spent was in minutes) and divided it by 60 to get time in hours and used table opsanalytics.jobs_data (Table I created in MYSQL using the given dataset), and lastly, used the WHERE function to narrow down the search results for the month of November.

**SQL Statement:**

```
SELECT
ds,
COUNT(job_id) AS jobs_per_day,
SUM(time_spent)/60 AS hours_spent
FROM opsanalytics.jobs_data
WHERE ds >='01-11-2020' AND ds <='30-11-2020' GROUP BY ds;
```

**Result:**

| ds | jobs_per_day | hours_spent |
|---|---|---|
| 30-11-2020 | 2 | 0.6667 |
| 29-11-2020 | 1 | 0.3333 |
| 28-11-2020 | 2 | 0.5500 |
| 27-11-2020 | 1 | 1.7333 |
| 26-11-2020 | 1 | 0.9333 |
| 25-11-2020 | 1 | 0.7500 |

**Conclusion:**

Following are the number of jobs reviewed per hour per day for the month of November 2020.

## B. Throughput or the no. of events happening per second.

Here I need to calculate the 7-day rolling average of throughput, and for throughput, do I prefer daily metric or 7-day rolling and why?

We know that a rolling average is a calculation that lets us analyze data points by creating a series of averages based on different subsets of a data set called a moving average or running average.

A 7-day moving average is a short-term trend indicator. It is simply the average closing values of the last seven days. I prefer a 7-day rolling average because it is reliable, and it is a simple indicator of changing the price/production of services or goods.

**SQL Statement:**

```
SELECT x.*,
AVG(dailyusage) over (partition by job_id order by job_id, ds rows
between 6 preceding and current row) AS rolling_avg
FROM (select job_id, ds, sum(time_spent) as dailyusage
FROM opsanalytics.jobs_data
GROUP BY job_id, ds) x;
```

**Result:**

| job_id | ds | dailyusage | rolling_avg |
|---|---|---|---|
| 20 | 25-11-2020 | 45 | 45.0000 |
| 21 | 30-11-2020 | 15 | 15.0000 |
| 22 | 30-11-2020 | 25 | 25.0000 |
| 23 | 26-11-2020 | 56 | 56.0000 |
| 23 | 28-11-2020 | 22 | 39.0000 |
| 23 | 29-11-2020 | 20 | 32.6667 |
| 25 | 28-11-2020 | 11 | 11.0000 |

**Conclusion:**

A 7-day rolling average is what I prefer because rolling averages can be used to uncover long-term trends that are occasionally masked by volatility.

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

Here I need to calculate the percentage share of each language in the last 30 days.

**SQL Statement:**

```
SELECT ds, language, event,
COUNT(*) as lang_usage,
COUNT(*) * 100.0 / sum(COUNT(*)) OVER () AS percentage
FROM opsanalytics.jobs_data
WHERE ds >='01-11-2020' and ds <='30-11-2020'
GROUP BY language ORDER BY event;
```

**Result:**

| ds | language | event | lang_usage | percentage |
|---|---|---|---|---|
| 29-11-2020 | Persian | decision | 3 | 37.50000 |
| 28-11-2020 | Hindi | decision | 1 | 12.50000 |
| 27-11-2020 | French | decision | 1 | 12.50000 |
| 30-11-2020 | English | skip | 1 | 12.50000 |
| 30-11-2020 | Arabic | transfer | 1 | 12.50000 |
| 25-11-2020 | Italian | transfer | 1 | 12.50000 |

**Conclusion:**

Following is the percentage share of each language for events, where we can conclude Persian is the most used language, and the rest, like Hindi, French, etc., have an equal share with a percentage share of 37.5% and 12.5%, respectively.

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

Here I need to display duplicates from the table.

**SQL Statement:**

```
SELECT job_id, actor_id,
COUNT(*)
FROM opsanalytics.jobs_data
GROUP BY job_id, actor_id
HAVING COUNT(*) > 1;
```

**Result:**

| job_id | actor_ID | COUNT(*) |
|---|---|---|

**Conclusion:**

Here all the rows have unique values. No rows were found that have the same value present in them.
Hence, there are no duplicate values to be displayed.

## II. Case Study 2 (Investigating metric spike)

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

Here I need to calculate the weekly user engagement.

**SQL statement:**

```
SELECT
EXTRACT(WEEK FROM occurred_at) AS weeknum,
COUNT(DISTINCT user_id) AS no_of_distinct_user
FROM `opsanalytics`.`table-2 events`
GROUP BY weeknum
```

**Conclusion:**

Following is the weakly user engagement.

**Result:**

| | A | B |
|---|---|---|
| 1 | weeknum | no_of_distinct_user |
| 2 | 17 | 85 |
| 3 | 18 | 194 |
| 4 | 19 | 208 |
| 5 | 20 | 195 |
| 6 | 21 | 208 |
| 7 | 22 | 230 |
| 8 | 23 | 224 |
| 9 | 24 | 252 |
| 10 | 25 | 245 |
| 11 | 26 | 230 |
| 12 | 27 | 249 |
| 13 | 28 | 114 |
| 14 | 29 | 18 |
| 15 | 30 | 7 |
| 16 | 31 | 7 |
| 17 | 32 | 1 |
| 18 | 33 | 2 |
| 19 | 34 | 1 |
| 20 | | |

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

Here I need to calculate the user growth for the product.

**SQL statement:**

```
SELECT
activated_at, weeknum, num_active_user,
SUM(num_active_user),
OVER(ORDER BY activated_at, weeknum ROWS BETWEEN UNBOUNDED
PRECEDING AND CURRENT ROW) AS cum_active_users
FROM
(SELECT EXTRACT(year from activated_at) AS year, EXTRACT(week from
activated_at) AS weeknum,
COUNT(DISTINCT user_id) AS num_active_user
FROM `opsanalytics`.`table-1 users` a
WHERE state='active'
GROUP BY year, weeknum
ORDER BY year, weeknum) a;
```

**Result:**

| year | weeknum | no_of distiner_user | cum_active_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 |

**Conclusion:** Following is the output for the query having the amount of users growing over time for a product.

## C. Weekly Retention:

Here I need to find users getting retained weekly after signing up for a product.

**SQL statement:**

```
SELECT week (occurred_at) AS week,
COUNT(CASE WHEN e.event_type = 'engagement' THEN e.user_id
ELSE NULL END) AS engagement,
COUNT(CASE WHEN e.event_type = 'signup_flow' THEN e.user_id
ELSE NULL END) AS signup
FROM `opsanalytics`.`table-2 events` e
GROUP BY 1
ORDER BY 1
```

**Conclusion:**

Following mentioned above are the metrics of the users getting retained weekly after signing up for a product, and the engagements related to it.

**Result:**

| week | engagement | signup |
|---|---|---|
| 17 | 617 | 72 |
| 18 | 1498 | 163 |
| 19 | 1552 | 185 |
| 20 | 1580 | 176 |
| 21 | 1612 | 183 |
| 22 | 1870 | 196 |
| 23 | 1795 | 196 |
| 24 | 2014 | 229 |
| 25 | 1798 | 207 |
| 26 | 1963 | 201 |
| 27 | 1988 | 222 |
| 28 | 1010 | 89 |
| 29 | 147 | 0 |
| 30 | 75 | 0 |
| 31 | 77 | 0 |
| 32 | 6 | 0 |
| 33 | 7 | 0 |
| 34 | 2 | 0 |

**D. Weekly Engagement:** To measure the activeness of a user. Measuring if the user finds quality in a product/service weekly.

Here I need to calculate the weekly engagement per device.

**SQL Statement:**

```
SELECT week (occurred_at) AS week,
COUNT(DISTINCT e.user_id) AS weekly_users,
COUNT(DISTINCT CASE WHEN e. device IN ('macbook pro', 'aceraspire notebook',
'acer aspire desktop','lenovo thinkpad', 'mac mini', 'dell inspirondesktop',
'dell inspiron notebook','windows surface','macbook air','asus chromebook','hppavilion
desktop')
THEN e.user_id ELSE NULL END) AS computer,
COUNT(DISTINCT CASE WHEN e. device IN ('iphone 5s',
'nokia lumia635','amazon fire phone','iphone 4s','htc one','iphone 5','samsung galaxy s4')
THEN e.user_id ELSE NULL END) AS phone,
COUNT(DISTINCT CASE WHEN e. device IN ('kindle fire',
'samsunggalaxy note','ipad mini','nexus 7','nexus 10','samsumg galaxy tablet','nexus 5',
'ipad air')
THEN e.user_id ELSE NULL END) AS tablet

FROM `opsanalytics`.`table-2 events` e
WHERE e. event_type = 'engagement'
AND e. event_name = 'login'
GROUP BY 1
ORDER BY 1
```

**Result:**

| week | weekly_users | computer | phone | tablet |
|---|---|---|---|---|
| 17 | 85 | 35 | 29 | 14 |
| 18 | 194 | 107 | 35 | 36 |
| 19 | 208 | 95 | 59 | 39 |
| 20 | 195 | 98 | 47 | 32 |
| 21 | 208 | 99 | 56 | 40 |
| 22 | 230 | 120 | 47 | 47 |
| 23 | 224 | 111 | 53 | 36 |
| 24 | 252 | 126 | 67 | 43 |
| 25 | 245 | 126 | 53 | 38 |
| 26 | 230 | 115 | 65 | 32 |
| 27 | 249 | 118 | 77 | 40 |
| 28 | 114 | 56 | 28 | 21 |
| 29 | 18 | 10 | 3 | 3 |
| 30 | 7 | 5 | 1 | 0 |
| 31 | 7 | 2 | 3 | 1 |
| 32 | 1 | 0 | 1 | 0 |
| 33 | 2 | 2 | 0 | 0 |
| 34 | 1 | 0 | 1 | 0 |

**Conclusion:**

Following is the weakly engagement per device, where users are using devices like computers, phone and tablets.

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

Here I need to calculate the email engagement metrics.

**SQL statement:**

```
SELECT
EXTRACT(WEEK FROM occurred_at) AS weeknum,
COUNT(DISTINCT e.user_id) AS weekly_users,
COUNT(CASE WHEN e.action = 'sent_weekly_digest' THEN e.user_id ELSE NULL
END) AS 'weekly_emails',
COUNT(CASE WHEN e.action = 'email_open' THEN e.user_id ELSE NULL END) AS
'email_opens',
COUNT(CASE WHEN e.action = 'email_clickthrough' THEN e.user_id ELSE NULL
END) AS 'email_click through'
FROM `opsanalytics`.`table-3 email_events` e
GROUP BY 1
```

**Conclusion:**

Following are the email metrics, where I found 18 rows of the metrics like weekly users, weekly emails, email opens, email clickthrough etc.

**Result:**

| weeknum | weekly_users | weekly_emails | email_opens | email_click through |
|---|---|---|---|---|
| 17 | 510 | 510 | 129 | 55 |
| 18 | 1462 | 1462 | 430 | 160 |
| 19 | 1462 | 1462 | 431 | 189 |
| 20 | 1462 | 1462 | 440 | 172 |
| 21 | 1462 | 1462 | 451 | 153 |
| 22 | 1462 | 1462 | 408 | 170 |
| 23 | 1462 | 1462 | 446 | 182 |
| 24 | 1462 | 1462 | 459 | 192 |
| 25 | 1462 | 1462 | 410 | 168 |
| 26 | 1462 | 1462 | 416 | 176 |
| 27 | 1462 | 1462 | 444 | 175 |
| 28 | 1462 | 1462 | 443 | 181 |
| 29 | 1462 | 1462 | 442 | 163 |
| 30 | 1461 | 1461 | 448 | 172 |
| 31 | 1461 | 1461 | 422 | 97 |
| 32 | 1461 | 1461 | 439 | 99 |
| 33 | 1461 | 1461 | 453 | 110 |

ult 11

**Result:**

I accomplished a great deal while working on the Operation Analytics and Investigating Metric Spike Report project. During the project, I gained knowledge about various techniques utilized for investigating metric spikes, and I also learned about the tools utilized for analyzing data. This project has provided me with practical knowledge about these concepts, enhancing my understanding of operation analytics and data analysis. Furthermore, the project has improved my problem-solving and analytical abilities.

DRIVE LINK:

https://drive.google.com/file/d/1m186RQGVljZULBtQ9ReMMtxUjTZijS7N/view?usp=share_link