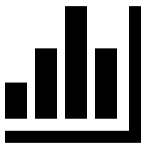
PERATION LYTICS AND ESTIGATING TRIC SPIKE

MAINAK GHOSH
mainakghosh623@gmail.com

DESCRIPTION

On a daily basis, there were numerous operations taking place in an organization. Operation analytics is such an analysis that is associated with the end-to-end operations of a company. It helps companies find out the exact scenario. It is very important for an organization to know if it is growing or declining. Better automation, improved communication among cross-functional teams, and more efficient workflows are the results. Matric Spike Investigation is a much-needed study to figure out the problems of other existing teams in an organization.



Approach

I'm going to take a really precise approach. In the beginning, I need to create a database; thus, I need to do the tasks. I was given one table and three CSV files, which I must first convert into a database before running my query.

STEP: Create a new schema in the connected server

--> Table Data Import Wizerd

--> Choose File

--> Create New Table

--> Import Data File

After building a database, I must execute the query in accordance with the management's requirements and provide a thorough report to convert the data into a particular piece of information.

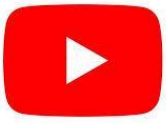
Tech-Stack Used











1.My SQL Workbench: Importing Database, Running all SQL Quaries
2. MS Excel: To extract the data in details, Deeply understanding the provided data.
3. Google Drive: To save the project.
4. MS Word: Making the project via ms word.

5. Youtube: For gaining additional knowledge, taking help when stuck with any quaries.

INSIGHT

CASE STUDY-1: JOB DATA

TASK:

1. Estimating the number of jobs that will be examined each hour and day in November 2020
2. Calculating the throughput's rolling 7-day average
3. calculating the proportion of each language in use over the previous 30 days
4. Displaying duplicates from the table

1.Number of jobs reviewed:

Estimating the number of jobs that will be examined each hour and day in November 2020. It is important to be aware of the precise number of jobs assessed throughout time.

QUERY

```
SELECT

ds,COUNT(job_id) AS jobs_per_day,

SUM(time_spent)/3600 AS hours_spent

FROM operations.`project-1_table`

GROUP BY  ds
```

2.Throughput:

Calculating the throughput's rolling 7-day average

QUERY

```
WITH CTE_Table AS (

SELECT ds, COUNT(job_id) AS num_jobs, SUM(time_spent) AS total_time

FROM operation.`sql project-1 table`

WHERE event IN('transfer','decision')

GROUP BY ds)

SELECT ds, ROUND (1.0*

SUM(num_jobs) OVER (ORDER BY ds ROWS BETWEEN 6 PRECEDING AND CURRENT ROW) /
SUM(total_time) OVER (ORDER BY ds ROWS BETWEEN 6 PRECEDING AND CURRENT ROW),2

) AS throughput_7d

FROM CTE_Table
```

3. **Percentage share of each language:**

It is critical to understand each language's share of various contents. I am calculating the proportion of each language in use over the course of 30 days.

QUERY

```
SELECT language,

num_jobs,

(100.0* num_jobs/total_jobs) AS percentage_of_language

FROM

(SELECT language, count(distinct job_id) as num_jobs

FROM operation.`sql project-1 table`

GROUP BY language)a

CROSS JOIN

(SELECT COUNT(DISTINCT job_id) AS total_jobs

FROM operation.`sql project-1 table`) b;
```

4. Duplicate Rows:

In that table, there are rows that have the same value. I need to show duplicates from the table.

QUERY

```
WITH CTE_Table AS (

SELECT *, ROW_NUMBER() OVER (PARTITION BY job_id) AS rownum

FROM operations.`project-1_table`)

Select *

FROM CTE_Table

WHERE rownum > 1
```

CASE STUDY-2: Investigating Metric Spike

TASK:

1. Calculating the weekly user engagement
2. Calculating the user growth for products
3. Calculating the weekly retention of users-sign up cohort
4. Calculating the weekly engagement per device
5. Calculating the email engagement metrics

1.USER ENGAGEMENT:

I must determine the weekly user engagement in order to gauge a user's activities. The user's perception of a product or service's quality will be the major focus of the analysis.

QUERY

```
SELECT

round('week', occurred_at) as week,

count(distinct user_id) as weekly_active_users

FROM operation.events

where event_type = 'engagement' and event_name = 'login'

group by 1
```

2.USER GROWTH:

I must know how many customers a product will eventually draw. I need to figure out the product's user growth in order to know this.

QUERY

```
SELECT

round('day', created_at) AS day, COUNT(*) AS all_users,

COUNT(CASE WHEN activated_at IS NOT NULL THEN u.user_id ELSE NULL END)

AS activated_users

FROM operation.users

WHERE created_at >= '2021-04-01' AND created_at < '2021-04-30'

GROUP BY day

ORDER BY 1
```

3.WEEKLY RETENTION:

QUERY

```
SELECT round('week', A.occurred_at) AS "week",

AVG(A.age_at_event) AS "Average age during week",

COUNT(DISTINCT CASE WHEN A.user_age > 70

THEN A.user_id ELSE NULL END) AS "10+ weeks",

COUNT(DISTINCT CASE WHEN A.user_age < 70 AND A.user_age >=63

THEN A.user_id ELSE NULL END) AS "9 weeks",

COUNT(DISTINCT CASE WHEN A.user_age < 63 AND A.user_age >=56

THEN A.user_id ELSE NULL END) AS "8 weeks",

COUNT(DISTINCT CASE WHEN A.user_age < 56 AND A.user_age >=49

THEN A.user_id ELSE NULL END) AS "7 weeks",

COUNT(DISTINCT CASE WHEN A.user_age < 49 AND A.user_age >=42

THEN A.user_id ELSE NULL END) AS "6 weeks",

COUNT(DISTINCT CASE WHEN A.user_age < 42 AND A.user_age >=35

THEN A.user_id ELSE NULL END) AS "5 weeks",

COUNT(DISTINCT CASE WHEN A.user_age < 35 AND A.user_age >=28

THEN A.user_id ELSE NULL END) AS "4 weeks",

COUNT(DISTINCT CASE WHEN A.user_age < 28 AND A.user_age >=21

THEN A.user_id ELSE NULL END) AS "3 weeks",

COUNT(DISTINCT CASE WHEN A.user_age < 21 AND A.user_age >=14

THEN A.user_id ELSE NULL END) AS "2 weeks",

COUNT(DISTINCT CASE WHEN A.user_age < 14 AND A.user_age >=7
```

```
THEN A.user_id ELSE NULL END) AS "1 weeks",
COUNT(DISTINCT CASE WHEN A.user_age < 7 AND A.user_age >=63
THEN A.user_id ELSE NULL END) AS "Less than a week"
FROM ( SELECT events.occurred_at, users.user_id, round("week",users.activated_at) AS
activation_week,
EXTRACT(day from events.occurred_at - users.activated_at) AS age_at_event,
EXTRACT(day FROM '2014-09-01' - users.activated_at) AS user_age
FROM operation.users users
JOIN operation.events events ON events.user_id = users.user_id
AND events.event_type = 'engagement'
AND events.event_name= 'login'
AND events.occurred_at >= '2014-05-01'
AND events.occurred_at < '2014-09-01'
WHERE users.activated_at IS NOT NULL ) A
GROUP BY 1
ORDER BY 1
```

4.WEEKLY ENGAGEMENT:

I have to compute the weekly engagement per device to determine a user's level of activity. Knowing if a user consistently values quality in a good or service is key.

QUERY

```
SELECT round('week', occurred_at) AS week,

COUNT(DISTINCT events.user_id) AS weekly_active_users,

COUNT(DISTINCT CASE WHEN e.device IN('macbook pro','lenovo thinkpad','macbook air','dell inspiron notebook','asus chromebook','dell inspiron desktop','acer aspire notebook','hp pavilion desktop','acer aspire desktop','mac mini') THEN events.user_id ELSE NULL END) AS computer,

COUNT(DISTINCT CASE WHEN events.device IN('iphone 5','samsung galaxy s4','nexus 5','iphone 5s','iphone 4s','nokia lumia 635','htc one','samsung galaxy note','amazon fire phone') THEN events.user_id ELSE NULL END) AS phone,

COUNT(DISTINCT CASE WHEN events.device IN('ipad air','nexus 7','ipad mini','nexus 10','kindle fire','windows surface','samsung galaxy tablet') THEN events.user_id ELSE NULL END) AS tablet

FROM operation.events events

WHERE events.event_type = 'engagement' AND events.event_name = 'login'

GROUP BY 1

ORDER BY 1
```

5.EMAIL ENGAGEMENT:

Users using the email service here.
Calculating the email engagement numbers is what I have to do.

QUERY

```
SELECT

ROUND('week', occurred_at) AS week,

COUNT(CASE WHEN email_events.action = 'sent weekly digest' THEN email_events.user_id ELSE NULL END) AS weekly_emails,

COUNT(CASE WHEN email_events.action = 'sent reengagement email' THEN email_events.user_id ELSE NULL END) AS reengagement_emails,

COUNT(CASE WHEN e.action = 'email open' THEN email_events.user_id ELSE NULL END) AS email_opens,

COUNT(CASE WHEN e.action = 'email clickthrough' THEN email_events.user_id ELSE NULL END) AS email_clickthroughs

FROM  operation.email_events  email_events

GROUP BY 1
```

RESULTS

I ran all the mentioned SQL queries above. With this questionnaire, I obtained some results that will undoubtedly aid the organisation in properly analysing them and moving forward in the right direction.

