OPERATION & METRIC ANALYTICS

Name :- Yash Shinde
Email :- yashpradeepshinde@gmailcom

TABLE OF CONTENT

PROJECT DESCRIPTION

PROJECT DESCRIPTION

- This project involves taking on the role of a senior data analyst and completely involves me in various datasets and spreadsheets related to the company's operations.
- As a data analyst, I will have to work closely with various teams within the company, including operations, support and marketing, to gain valuable insights from the data they have collected.
- The main focus of this project is to investigate spikes in metrics, understand sudden changes in key metrics such as daily user engagement or sales drops, and provide daily answers to related questions.
- Using my advanced SQL skills, I will analyze data and provide actionable insights to improve the company's operations and understand sudden metrics changes. By working on this project, I will play a key role in data-driven decision-making and help my organization optimize its overall performance and efficiency.

APPROACH

APPROACH

- My approach through this project would be first creating a database for a project
- Then solve the questions through SQL queries using the MySQL Workbench.
- I will provide a detailed explanation of the query along with the results of each query.
- I will perform my analysis using the following list of points.

Case Study 1: Job Data Analysis:-

1. Jobs Reviewed Over Time
2. Throughput Analysis
3. Language Share Analysis
4. Duplicate Rows Detection

Case Study 2: Investigating Metric Spike

1. Weekly User Engagement
2. User Growth Analysis
3. Weekly Retention Analysis
4. Weekly Engagement Per Device
5. Email Engagement Analysis

TECH STACK USED

- MySQL Workbench
- Microsoft PowerPoint

ANALYSIS

Case Study 1: Job Data Analysis
Case Study 2: Investigating Metric Spike

Job Data Analytics

JOBS REVIEWED OVER TIME:

OBJECTIVE: CALCULATE THE NUMBER OF JOBS REVIEWED PER HOUR FOR EACH DAY IN NOVEMBER 2020.
TASK: WRITE AN SQL QUERY TO CALCULATE THE NUMBER OF JOBS REVIEWED PER HOUR FOR EACH DAY IN NOVEMBER 2020.

THROUGHPUT ANALYSIS:

OBJECTIVE: CALCULATE THE 7-DAY ROLLING AVERAGE OF THROUGHPUT.
TASK: WRITE AN SQL QUERY TO CALCULATE THE 7-DAY ROLLING AVERAGE OF THROUGHPUT. ALSO, EXPLAIN WHY YOU PREFER USING THE DAILY METRIC OR THE 7-DAY ROLLING AVERAGE FOR THROUGHPUT.

LANGUAGE SHARE ANALYSIS:

OBJECTIVE: CALCULATE THE PERCENTAGE SHARE OF EACH LANGUAGE IN THE LAST 30 DAYS.
TASK: WRITE AN SQL QUERY TO CALCULATE THE PERCENTAGE SHARE OF EACH LANGUAGE OVER THE LAST 30 DAYS.

DUPLICATE ROWS DETECTION:

OBJECTIVE: IDENTIFY DUPLICATE ROWS IN THE DATA.
TASK: WRITE AN SQL QUERY TO DISPLAY DUPLICATE ROWS FROM THE JOB_DATA TABLE.

Job Data Analytics

JOBS REVIEWED OVER TIME:

CREATE AN SQL QUERY TO CALCULATE THE NUMBER OF JOBS REVIEWED PER HOUR FOR EACH DAY IN NOVEMBER 2020.

- Steps for finding the number of jobs reviewed :

  Step 1) I have selected `operation & metric analytics` database for accessing data.

  Step 2) To calculate a number of jobs reviewed per hour we just have to calculate the total number of job_id from job_data and then divide it by 24*30.

  Step 3) Using SELECT and COUNT commands I have executed the following Query.

  Step 4) It is giving an output of 0.0111 jobs.

Query/Program :

```
SELECT
   COUNT(job_id) / (24 * 30) AS
No_of_jobs_reviewed
FROM
   job_data;
```

Output/Result Table :

No_of_jobs_reviewed
0.0111

Job Data Analytics

THROUGHPUT ANALYSIS : CREATE AN SQL QUERY TO CALCULATE THE 7-DAY ROLLING AVERAGE OF THROUGHPUT. ALSO, EXPLAIN WHY YOU PREFER USING THE DAILY METRIC OR THE 7-DAY ROLLING AVERAGE FOR THROUGHPUT.

- Steps for finding the 7-day rolling average of throughput :

  Step 1) I have selected `operation & metric analytics` database for accessing data.

  Step 2) Then I created a temporary table (i.e. Common Table Expression) named jobs_by_date. In that table, I have counted total number of jobs as jobs_reviewed from the "job_data" table and grouped them using the dates of "ds" column.

  Step 3) Then I have used that temporary table in the following command along with window function to make a third column named "rolling_average_throughput_7days".

  Step 4) In window function I have used AVG aggregate function to find the rolling average of the 7 days.

  Step 5) Then inside the OVER command I used ORDER BY review_date to order the final result in the ascending order of the date.

Job Data Analytics

THROUGHPUT ANALYSIS : CREATE AN SQL QUERY TO CALCULATE THE 7-DAY ROLLING AVERAGE OF THROUGHPUT. ALSO, EXPLAIN WHY YOU PREFER USING THE DAILY METRIC OR THE 7-DAY ROLLING AVERAGE FOR THROUGHPUT.

Query/Program :

```
WITH jobs_by_date AS (
    SELECT
        ds as review_date,
        COUNT(job_id) AS jobs_reviewed
    FROM job_data
    GROUP BY ds
)
SELECT
    review_date,
    jobs_reviewed,
    AVG(jobs_reviewed) OVER (ORDER BY review_date) AS rolling_average_throughput_7days
FROM
    jobs_by_date
ORDER BY
    review_date;
```

Job Data Analytics

THROUGHPUT ANALYSIS : CREATE AN SQL QUERY TO CALCULATE THE 7-DAY ROLLING AVERAGE OF THROUGHPUT. ALSO, EXPLAIN WHY YOU PREFER USING THE DAILY METRIC OR THE 7-DAY ROLLING AVERAGE FOR THROUGHPUT.

Output/Result Table :

review_date	jobs_reviewed	rolling_average_throughput_7days
11/25/2020	1	1
11/26/2020	1	1
11/27/2020	1	1
11/28/2020	2	1.25
11/29/2020	1	1.2
11/30/2020	2	1.3333

Job Data Analytics

LANGUAGE SHARE ANALYSIS: CALCULATE THE PERCENTAGE SHARE OF EACH LANGUAGE IN THE LAST 30 DAYS. WRITE AN SQL QUERY TO CALCULATE THE PERCENTAGE SHARE OF EACH LANGUAGE OVER THE LAST 30 DAYS.

- Steps for finding the percentage share of each language :

  Step 1) I have selected `operation & metric analytics` database for accessing data.

  Step 2) Then I created a temporary table (i.e. Common Table Expression) named language_total.

  In that table, I have counted total number of occurrences of each language as "total" from "job_data" table and grouped them using the "language" column.

  Step 3) Then I used that temporary table in the following command for calculation of percentage.

  Step 4) Formula used for percentage is "(total / (SELECT COUNT(*) FROM job_data)) * 100"

Query/Program :

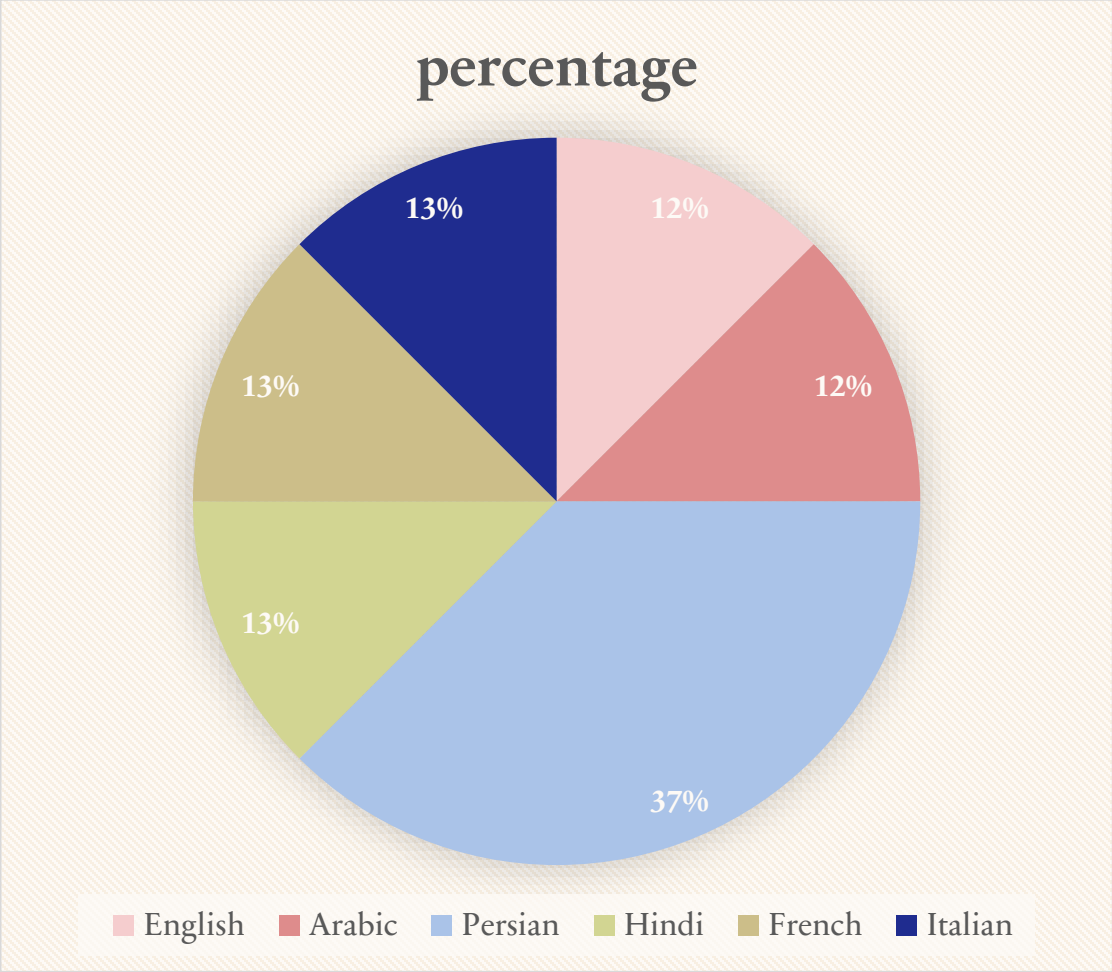
```
with language_total as (
select job_id, language,
count(language) as total
from job_data
group by language)

SELECT
   job_id,  language, total,
   (total / (SELECT COUNT(*) FROM job_data)) *
100 AS percentage
FROM
   language_total;
```

Job Data Analytics

LANGUAGE SHARE ANALYSIS: CALCULATE THE PERCENTAGE SHARE OF EACH LANGUAGE IN THE LAST 30 DAYS. WRITE AN SQL QUERY TO CALCULATE THE PERCENTAGE SHARE OF EACH LANGUAGE OVER THE LAST 30 DAYS.

Output/Result Table :

job_id	language	total	percentage
21	English	1	12.5
22	Arabic	1	12.5
23	Persian	3	37.5
25	Hindi	1	12.5
11	French	1	12.5
20	Italian	1	12.5



Job Data Analytics

DUPLICATE ROWS DETECTION: IDENTIFY DUPLICATE ROWS IN THE DATA. WRITE AN SQL QUERY TO DISPLAY DUPLICATE ROWS FROM THE JOB_DATA TABLE.

- Steps for finding the duplicate rows :

  Step 1) I have selected all columns from "job_data" table where all columns are present in the same table which is selected using the derived table and 'IN' command.

  Step 2) Then I created a derived table which has all columns grouped by all columns from table "job_data" as we want to check if there is any duplicate row in the table.

  Step 3) Then by using "HAVING COUNT(*)>1" we have selected only those rows which are duplicate.

  Step 4) In table job_data there are no duplicate rows so we get empty table as output.

Query/Program :

```
SELECT *
FROM job_data
WHERE (ds, job_id, actor_id, event, language, time_spent, org) IN (
   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
);
```

Job Data Analytics

DUPLICATE ROWS DETECTION: IDENTIFY DUPLICATE ROWS IN THE DATA. WRITE AN SQL QUERY TO DISPLAY DUPLICATE ROWS FROM THE JOB_DATA TABLE.

Output/Result Table :

ds	job_id	actor_id	event	language	time_spent	org

- In the job_data table there are no duplicate rows, so we get an empty table as output.

Investigating Metric Spike

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.

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.

WEEKLY RETENTION ANALYSIS:

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.

WEEKLY ENGAGEMENT PER DEVICE:

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.

EMAIL ENGAGEMENT ANALYSIS:

OBJECTIVE: ANALYZE HOW USERS ARE ENGAGING WITH THE EMAIL SERVICE.
YOUR TASK: WRITE AN SQL QUERY TO CALCULATE THE EMAIL ENGAGEMENT METRICS.

Investigating Metric Spike

WEEKLY USER ENGAGEMENT: MEASURE THE ACTIVENESS OF USERS ON A WEEKLY BASIS. WRITE AN SQL QUERY TO CALCULATE THE WEEKLY USER ENGAGEMENT.

- Steps for finding the weekly user engagement :

  Step 1) I have selected `operation & metric analytics` database for accessing data.

  Step 2) I have extracted week number from "occurred_at" column using the WEEK function as week_number.

  Step 3) Then I counted the number of distinct user_id from the events table using COUNT function.

  Step 4) By using GROUP BY clause grouped distinct user ids in each week. As we want to measure the activeness of users on a weekly basis.

Query/Program :

```
SELECT
   WEEK(occurred_at) AS week_number,
   COUNT(DISTINCT user_id) AS active_users
FROM events
GROUP BY week_number
ORDER BY  week_number ;
```

Investigating Metric Spike

Output/Result Table :

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

Investigating Metric Spike

USER GROWTH ANALYSIS: ANALYZE THE GROWTH OF USERS OVER TIME FOR A PRODUCT. WRITE AN SQL QUERY TO CALCULATE THE USER GROWTH FOR THE PRODUCT.

- Steps for finding the user growth analysis :

  Step 1) User growth of users over time for a product means a number of users signed up over time.

  Step 2) I have extracted rows from table “events” where (event_name = "complete_signup") which gives us the entries for only user signups from all the events from events table.

  Step 3) Then I have created an derived table named weekly signup to count the total number of sign ups each week.

  Step 4) Then by using SELECT command on the table weekly_signup. I have created a new table where I have added new table named “cumulative_signup”.

  Step 5) To calculate cumulative signup I have used an window function “OVER()”.

Investigating Metric Spike

USER GROWTH ANALYSIS: ANALYZE THE GROWTH OF USERS OVER TIME FOR A PRODUCT. WRITE AN SQL QUERY TO CALCULATE THE USER GROWTH FOR THE PRODUCT.

Query/Program :

```
with signup as(
select occurred_at, event_name from events where event_name = "complete_signup")

SELECT
   week_number,
   total_signup,
   SUM(total_signup) OVER (ORDER BY week_number) AS cumulative_signup
FROM
(SELECT
   WEEK(occurred_at) AS week_number,
   COUNT(event_name) as total_signup
FROM signup
GROUP BY week_number
ORDER BY  week_number) as  weekly_signup;
```

Investigating Metric Spike

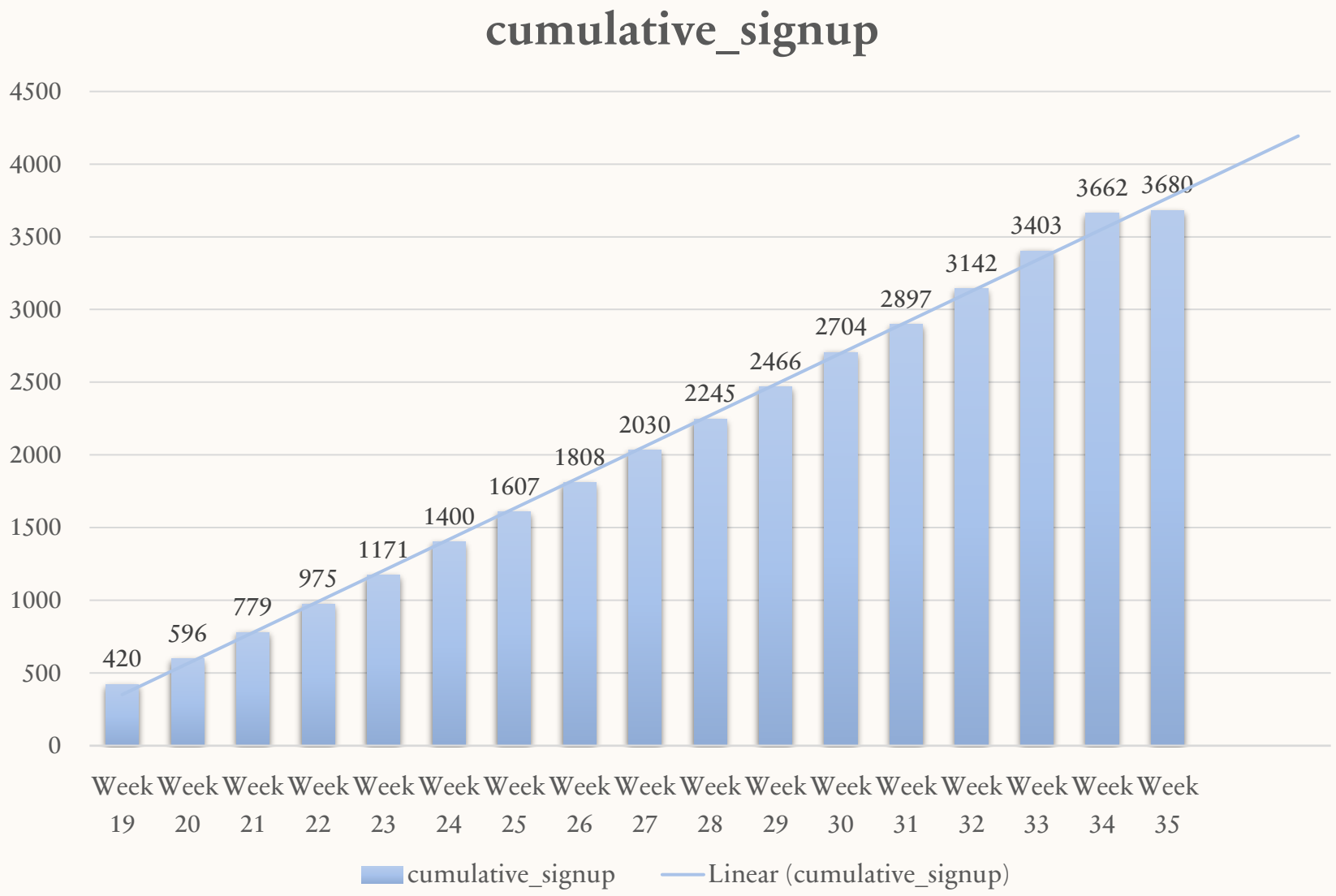
Output/Result Table :

week_number	total_signup	Cumulative_signup
17	72	72
18	163	235
19	185	420
20	176	596
21	183	779
22	196	975
23	196	1171
24	229	1400
25	207	1607
26	201	1808
27	222	2030
28	215	2245
29	221	2466
30	238	2704
31	193	2897
32	245	3142
33	261	3403
34	259	3662
35	18	3680

- This data clearly shows us user growth for the product is constantly increasing.
- That is shown graphically in the next slide through the bar chart on the week_number and Cumulative_signup column.

Investigating Metric Spike

Output/Result Table :



Investigating Metric Spike

WEEKLY RETENTION ANALYSIS: ANALYZE THE RETENTION OF USERS ON A WEEKLY BASIS AFTER SIGNING UP FOR A PRODUCT. WRITE AN SQL QUERY TO CALCULATE THE WEEKLY RETENTION OF USERS BASED ON THEIR SIGN-UP COHORT.

Query/Program :

```
with activities as(
select occurred_at, event_name from events )
SELECT
   week_number,
   total_activities,
   SUM(total_activities) OVER (ORDER BY week_number) AS cumulative_activities
FROM
(SELECT
   WEEK(occurred_at) AS week_number,
   COUNT(event_name) as total_activities
FROM activities
GROUP BY week_number
ORDER BY  week_number) as  weekly_activities;
```

Investigating Metric Spike

Output/Result Table :

week_number	total_activities	cumulative_activities
17	8091	8091
18	17504	25595
19	17409	43004
20	18087	61091
21	17334	78425
22	18609	97034
23	18476	115510
24	19281	134791
25	18849	153640
26	19262	172902
27	20103	193005
28	20991	213996
29	20288	234284
30	21771	256055
31	18749	274804
32	16857	291661
33	16406	308067
34	16386	324453
35	802	325255

- This data clearly shows us user retention is increasing.

Investigating Metric Spike

WEEKLY ENGAGEMENT PER DEVICE: MEASURE THE ACTIVENESS OF USERS ON A WEEKLY BASIS PER DEVICE. WRITE AN SQL QUERY TO CALCULATE THE WEEKLY ENGAGEMENT PER DEVICE.

- Steps for finding the weekly engagement per device :

  Step 1) I have selected `operation & metric analytics` database for accessing data.

  Step 2) I have extracted week number from "occurred_at" column using the WEEK function as week_number.

  Step 3) Then I counted the number of distinct user_id from the events table using COUNT function.

  Step 4) By using WHERE clause selected event_type as 'engagement'.

  Step 5) Finally combined the result using the GROUP BY clause on week_num and device.

Query/Program :

```
SELECT
week(occurred_at) as week_num,
device,
COUNT(distinct user_id) as no_of_users
FROM
events
where event_type = 'engagement'
GROUP by week_num, device
order by week_num;
```

Investigating Metric Spike

WEEKLY ENGAGEMENT PER DEVICE: MEASURE THE ACTIVENESS OF USERS ON A WEEKLY BASIS PER DEVICE. WRITE AN SQL QUERY TO CALCULATE THE WEEKLY ENGAGEMENT PER DEVICE.

Output/Result Table :

- Output has 492 rows thus I have provided the output csv file in the following drive link.

Drive Link:-
https://drive.google.com/file/d/1vZ5FddRzskdvGVb0qUUvbh5-N7eKAI6E/view?usp=sharing

Investigating Metric Spike

EMAIL ENGAGEMENT ANALYSIS: ANALYZE HOW USERS ARE ENGAGING WITH THE EMAIL SERVICE & WRITE AN SQL QUERY TO CALCULATE THE EMAIL ENGAGEMENT METRICS.

- Steps for finding the weekly engagement per device :

  Step 1) I have selected `operation & metric analytics` database for accessing data.

  Step 2) Then I have classified the actions into three categories: email_sent, email_opened, and email_clicked.

  Step 3) That categorization will be performed using the CASE, WHEN, THEN functions.

  Step 4) Then, I have calculated the email_opening_rate by summing up the occurrences of email_opened events and dividing the result by the sum of email_sent events. The final value is multiplied by 100.0 to represent the percentage accurately.

  Step 5) Similarly, I have computed the email_clicking_rate by summing up the occurrences of email_clicked events and dividing the result by the sum of email_sent events. The outcome will also be multiplied by 100.0 to express the rate as a percentage.

Investigating Metric Spike

Query/Program :

```
SELECT
  100.0 * COUNT(CASE WHEN email = 'email_opened' THEN 1 END) / COUNT(CASE WHEN email =
'email_sent' THEN 1 END) AS email_opening_percentage,
  100.0 * COUNT(CASE WHEN email = 'email_clicked' THEN 1 END) / COUNT(CASE WHEN email =
'email_sent' THEN 1 END) AS email_clicking_percentage
FROM (
  SELECT
    *,
    CASE
      WHEN action IN ('sent_weekly_digest', 'sent_reengagement_email') THEN 'email_sent'
      WHEN action IN ('email_open') THEN 'email_opened'
      WHEN action IN ('email_clickthrough') THEN 'email_clicked'
    END AS email
  FROM
    email_events
) email_action;
```

Output/Result Table :

email_opening_percentage	email_clicking_percentage
33.58339	14.78989

CONCLUSION

CONCLUSION

- Thus, I have explored provided Job data and Investigated metric spike analytics.
- Given all the required insights into the Marketing and Investees Matrices.
- I have learned to handle the database on the MySQL Workbench.
- I have learned to gain insights by using Queries.
- All the respective queries and their output is attached to this report.
- GitHub Repository and drive links are given as follows.

GitHub Repository:-
https://github.com/ShindeYash/Operation_and_Metric_Analytics.git

Drive Link:-
https://drive.google.com/drive/folders/1AG1LqrLAbPbGbaLKe0rd3Cpz24Ndh9w9?usp=sharing

THANK YOU

Yash Shinde
yashpradeepshinde@gmail.com