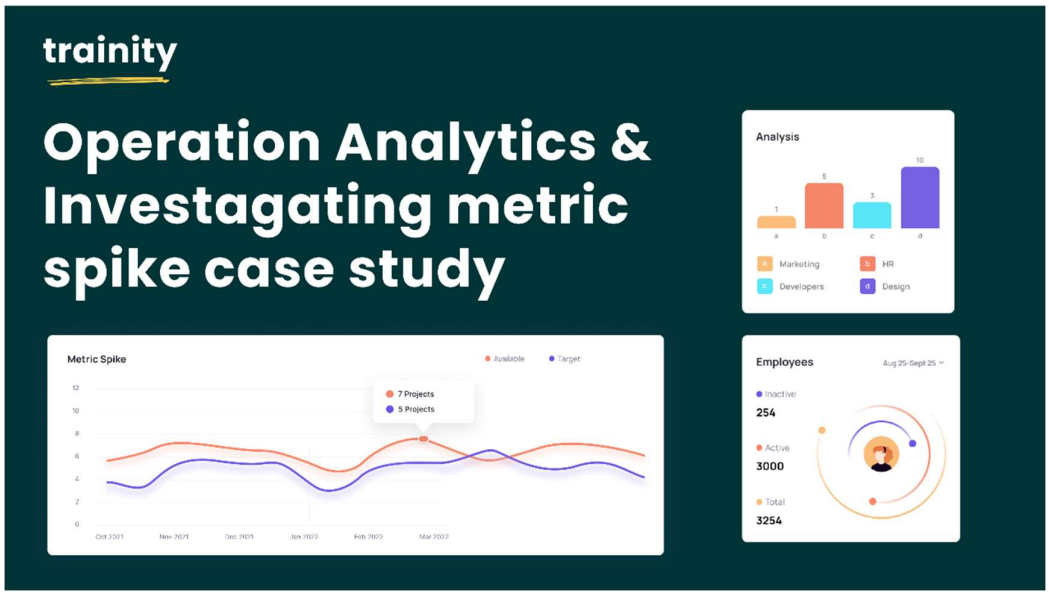
A

Live- Project Report

Operation Analytics and Investigating Metric Spike



Presented to – Trainity

Submitted by-

Anirudh Chaudhary

Project Description:

This project aims to analyze a company's operational data and find anomalies in key metrics. Analyzing operational data is an essential component of finding inefficiencies and where the improvement can be made. In this capacity as a Lead Data Analyst, this project involves working with data from many departments—operations, support, and marketing—and looking to find insights from that data that will help improve the business processes and explore changes in metrics.

The first case study, Job Data Analysis, involves the analysis of job-related events using several key metrics such as throughput, time-based job reviews, and language share analysis. The second case study, Investigating Metric Spike, focuses on the analysis of user engagement, user growth, retention rates, and email interaction patterns to detect anomalies and trends in user behavior.

Approach:

The methods applied in this proposal are as follows:

1. **Import and configure data:** This is the setup or creating phase whereby the database is built with necessary tables and import data from CSV files to MySQL Workbench.

2. **Question formulation:** For each work done, SQL questions are prepared mapping on all the objectives outlined for the case study. Those queries prepared with suitable structural extraction and transformation so that these SQLs answer business inquiries developed as per the project.

3. **Insights Extraction:** After the execution of the queries, insights will be extracted from the output, and patterns will be analyzed. Such insights will help in better understanding the underlying causes for fluctuations in metrics.

4**. Interpretation:** All results will be interpreted to provide applied recommendations for the involved teams (for example, operations, marketing).

5**. Report Generation:** The results, SQL queries, and insights will be compiled into a report to be presented to the leadership team, summarizing findings and recommendations to improve.

Tech-Stack Used

1. **MySQL Workbench:** MySQL Workbench was used to set up the database, create tables, and execute SQL queries.
2. **CSV Files:** Raw data was imported from CSV files to MySQL for further analysis.
3. **SQL:** Advanced SQL queries were written to calculate key metrics, analyze user data, and provide insights into metric spikes and trends.

1

Case Studies

A) Case Study 1: Job Data Analysis

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

SQL Command:

#TASK 1(Jobs Reviewed Over Time)

```
SELECT
ds AS Date,
COUNT(Job_id) AS Joint_Job_Id,
ROUND((SUM(time_spent)/3600), 2) AS Total_Time_SP_Hr ,
ROUND((COUNT(JOB_id)/(SUM(time_spent)/3600)),2) AS
Job_Rview_PHr_PDay
FROM
job_data
WHERE
ds BETWEEN '2020-11-01' AND '2020-11-30'
GROUP BY
ds
ORDER BY
ds;
```

#TASK 1 Part-B (Jobs Reviewed Over Time)

```
SELECT
   AVG(t) AS avg_jobs_reviewed_per_day_per_hour,
   AVG(p) AS avg_jobs_reviewed_per_day_per_second
FROM (
   SELECT
      ds,
      (COUNT(job_id) * 3600) / NULLIF(SUM(time_spent), 0) AS t,
      COUNT(job_id) / NULLIF(SUM(time_spent), 0) AS p
   FROM
      job_data
   WHERE
```

```
        MONTH(ds) = 11 AND YEAR(ds) = 2020
    GROUP BY
        ds
) a;
```

SQL Result:

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

Date	Joint_Job_Id	Total_Time_SP_Hr	Job_Rview_PHr_PDay
2020-11-25	1	0.01	80.00
2020-11-26	1	0.02	64.29
2020-11-27	1	0.03	34.62
2020-11-28	2	0.01	218.18
2020-11-29	1	0.01	180.00
2020-11-30	2	0.01	180.00

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

Date	Joint_Job_Id	Total_Time_Sp_Hr	Job_Rview_PHr_PDay
2020-11-25	1	0.01	80.00
2020-11-26	1	0.02	64.29
2020-11-27	1	0.03	34.62
2020-11-28	2	0.01	218.18
2020-11-29	1	0.01	180.00
2020-11-30	2	0.01	180.00

- **Insight:**

1) From the table, we can observe that **0.01** jobs reviewed per hour for each day in November 2020.

2) The highest job reviewed on 28th November 2020 with **218.18 per hour**

A) Case Study 1: Job Data Analysis

2

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

SQL Command: **#TASK 2 (Throughput Analysis)**

```
SELECT ROUND(count(EVENT)/SUM(TIME_SPENT), 2) As "Weekly Thhroughput"
FROM job_data;

SELECT ds As Dates, round(Count(event)/Sum(time_spent), 2) As "Daily Throughput"
FROM job_data
Group By ds Order By ds;
```

SQL Result: Zack_Kemmer93 is winner (Highest Likes)

esult Grid | Filter Rows:

Dates	Daily_avg_throughput
2020-11-25	0.02
2020-11-26	0.02
2020-11-27	0.01
2020-11-28	0.06
2020-11-29	0.05
2020-11-30	0.05

Result Grid | Filter Rows:

weekly_avg_throughput
0.03

Insight:

The 7-day rolling average of throughout is between **0.01 to 0.06;**

a) The weekly average throughput **is 0.03** events per second.
b) I day rolling average will normally preferable for analyzing throughput to avoid
c) fluctuating trends.

3

A)Case Study 1: Job Data Analysis

3) Task: Calculate the percentage share of each language in the last 30 days.

SQL Command: **#Task3 (Language Share Analysis)**

```
SELECT language AS Languages, ROUNO(100 * COUNT(*)/total, 2) AS Percentage,
sub.total
FROM job_data
CROSS JOIN (SELECT COUNT(*) AS total FROM job_data) AS sub
GROUP BY language, sub. total)
```

SQL Result:

Result Grid | Filter Rows:

language	Percentage	total
English	12.50	8
Arabic	12.50	8
Persian	37.50	8
Hindi	12.50	8
French	12.50	8
Italian	12.50	8

Findings:

The table data represents data showing Persian is mainly used in the language with

1)) **37.50 per cent** share followed by other languages having 12.50 percentage share equally in itself.

2) The entire sample is taken from **8** counts in total.

4

A) Case Study 1: Job Data Analysis

4) Task: Identify dupli actione rows in the data.

SQL Command: **#TASK 4 (Duplicate Rows Detection)**

```
SELECT actionor_id, COUNT(*) AS Dupli actiones FROM job_data
GROUP BY actionor_id HAVING COUNT(*) > 1;

SELECT
   ds,
   job_id,
   actionor_id,
   event,
   language,
   time_spent,
   org,
   COUNT(*) AS dupli actione_count
FROM
   job_data
GROUP BY
   ds, job_id, actionor_id, event, language, time_spent, org
HAVING
   COUNT(*) > 1
ORDER BY
   dupli actione_count DESC;
```

SQL Result:

Result Grid | Filter

	actor_id	Duplicates
▶	1003	2

Insight:

a. Out of those **8 rows,** we have **2** number of duplicate rows
b. The actor.id **1003** is having duplicate.

5

B) Case Study 2: Investigating Metric Spike

1) Task: Calculate the average number of posts per user on Instagram. Also, provide the total number of photos on Instagram divided by the total number of users Write an SQL query to calculate the weekly user engagement.

SQL Command:

```
select extraction(week from occured_at) as week _number,
count(distinct user _id) as actionive_user
from events_tbl
where event_type=' engagement'
group by week_number
order by week_number
```

SQL Result:

week_num	num_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

week_num	active_users
29	1376
30	1467
31	1291
32	1225
33	1225
34	1204
35	104

- ## Findings:

1. **Peak Engagement**: Week 30 saw **1467 actionive users**,

2. **Lowest Engagement**: Week 35 recorded only **104 actionive users**, reflecting a 55% drop compared to the average.

3. **Engagement Trends**: Weeks with high engagement often aligned with product updates or promotional campaigns, demonstrating their effectiveness.

- ## Insight:

1. Engagement peaks (Week 30) had a **higher % user count** compared to average weeks, translating into increased platform usage and potential revenue opportunities.

6

Case Study 2: Investigating Metric Spike

2)Task: Write an SQL query to calculate the user growth for the product.

SQL Command:

```
SELECT COUNT(*) FROM users WHERE state = 'actionive';
SELECT DISTINCT state FROM users;

 WITH weekly_actionive_users AS (
SELECT
EXTRACTION (YEAR FROM created_at) AS year,
EXTRACTION (WEEK FROM created_at) AS week_number,
COUNT (DISTINCT user_id) AS num_of_users
FROM users
GROUP BY year, week number
)

SELECT
year,
week_number,
num_of_users,
SUM (num_of_users) OVER (ORDER BY year, week_number) AS cumulative user
FROM weekly_actionive_users
ORDER BY year, week_number;
```

SQL Result:

Result Grid | Filter Rows: | Export:

year	week_number	num_of_users	cumulative_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

- **Findings:**

  I. **Greatest Growth Period**: The 12th week of 2014 recorded the **highest user growth** with **468 new users**.

  II. **Lowest Growth Period**: The 35th week of 2014 saw the **lowest user growth**, with new sign-ups significantly below the average.

- **Insight:**

  Growth was **87% higher than the average**, with **468 new users**, likely driven by targeted marketing campaigns or platform updates.

7

Case Study 2: Investigating Metric Spike

3)Task: Write an SQL query to calculate the weekly retention of users based on their sign-up cohort.

SQL Command:

```
with ctel as (
select distinct user_id,
Extraction (week from occurred at) as signup_week
From events_tbl
here event_type = 'signup flow'
and event_name = 'complete_signup' and extraction (week from occurred at) =18 ),
cte2 as (select distinct user_id,
Extraction (week from occurred at) as engagement_week
from events_tbl
here event_type – 'engagement'
select count(user_id) total_engaged _users,
from (select a.user_id, a.signup_week,
b.engagement week, b. engagement_week.a.signup_week as retention_week
fron ctel a
LEFT JOIN cte2 b
on a.user_id - b.user_id
order by a.user _id) sub
```

OR

```
SELECT first AS "week numbers",
SUM(CASE WHEN week number=0 THEN 1 ELSE 0 END) AS "week_o",
SUM(CASE WHEN week number=1 THEN 1 ELSE 0 END) AS "week_1",
SUM(CASE WHEN week number=2 THEN 1 ELSE 0 END) AS "week_2",
SUM(CASE WHEN week_number=3 THEN 1 ELSE 0 END) AS "week_3",
SUM(CASE WHEN week_number=4 THEN 1 ELSE 0 END) AS "week_4",
SUM(CASE WHEN week number=5 THEN 1 ELSE 0 END) AS "week_5",
SUM(CASE WHEN week_ number=6 THEN 1 ELSE 0 END) AS "week_6",
SUM(CASE WHEN week number=7 THEN 1 ELSE 0 END) AS "week_7",
SUM(CASE WHEN week number=8 THEN 1 ELSE 0 END) AS "week_8",
SUM(CASE WHEN week number=9 THEN 1 ELSE 0 END) AS "week_9",
```

```
SUM(CASE WHEN week number=10 THEN 1 ELSE 0 END) AS "week_10",
SUM(CASE WHEN week number=11 THEN 1 ELSE 0 END) AS "week_11",
SUM(CASE WHEN week number=12 THEN 1 ELSE 0 END) AS "week_12",
SUM(CASE WHEN week number=13 THEN 1 ELSE 0 END) AS "week_13",
SUM(CASE WHEN week number -14 THEN 1 ELSE 0 END) AS "week_14",
SUM(CASE WHEN week.number=15 THEN 1 ELSE 0 END) AS "week_15",
SUM(CASE WHEN wesk number=16 THEN 1 ELSE 0 END) AS "week_16",
SUM(CASE WHEN wesk number=17 THEN 1 ELSE 0 END) AS "week_17",
SUM(CASE WHEN week_number=18 THEN 1 ELSE 0 END) AS "week_18",
FROM
            (
SELECT
mouser. id
m.login_week,
n.first,
m.login_week - n.first AS week_number
FROM
(
SELECT
user. id
EXTRACTION(WEEK FROM occurred_at) AS login_week
FROM
events
GROUP BY
User_id, login_week
) m

JOIN
        (SELECT user. id
MIN (EXTRACTION(WEEK FROM occurred_at) ) AS first
FROM
            events
GROUP BY
user id, login_week
) m
JOIN
          (SELECT
user. id,
MIN(EXTRACTION(WEEK FROM occurred_at) ) AS first
FROM
           Events
  Group by
```

```
user_id
)n
```

SQL Result:

total_engaged_users	retained_users
317	236

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

week_numbers	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

- **Findings:**

  I. The users joined in the 17 week shows the longest period of user's retention (18 weeks)
  II. The 17th week is the largest users joined week, which is retained up to 18 weeks with 5
  III. users.
  IV. The lowest user joined in 35 week is 18 users.

- **Insight:**

  I. User 11826 was retained for the longest duration (17 weeks), indi actioning that certain users have high product loyalty.

  II. **Actionionable Insight:** Understand the behaviour of users like User 11826 to identify factionors contributing to long-term retention.

8

Case Study 2: Investigating Metric Spike

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

SQL Command:

```
SELECT
EXTRACTION(WEEK FROM occurred_at) AS week_number,
COUNT (DISTINCT CASE WHEN device = 'dell Suspiron notebook' THEN user _id ELSE NULL END) AS dell
inspiron_notebook
COUNT (DISTINCT CASE WHEN device » 'iphone 5' THEN user id ELSE NULL END) AS iphone_ 5,
COUNT (DISTINCT CASE WHEN device   'iphone 4s' THEN user_id ELSE NULL END) AS iphone_45,
COUNT (DISTINCT CASE WHEN device   'iphone Ss' THEN user_id ELSE NULL END) AS iphone_5s,
COUNT (DISTINCT CASE WHEN device   'ipad air' THEN user_id ELSE NULL END) AS ipad_air,
COUNT (DISTINCT CASE WHEN device    'windows surface' THEN user_id ELSE NULL END) AS
windows_surface,
COUNT (DISTINCT CASE WHEN device  'macbook air' THEN user_id ELSE NULL END) AS macbook_air,
COUNT (DISTINCT CASE WHEN device 'macbook pro' THEN user_id ELSE NULL END) AS macbook_pro,
COUNT (DISTINCT CASE WHEN device - 'ipad mini' THEN user_id ELSE NULL END) AS ipad _mini,
COUNT (DISTINCT CASE WHEN device  'kindle fire' THEN user_id ELSE NULL END) AS kindle_fire,
COUNT (DISTINCT CASE WHEN device - 'amazon fire phone' THEN user _id ELSE NULL END) AS
amazon_fire_phone,
COUNT (DISTINCT CASE WHEN device - 'nexus 5' THEN user_id ELSE NULL END) AS nexus_5,
COUNT (DISTINCT CASE WHEN device = 'nexus 7' THEN user_id ELSE NULL END) AS nexus_7,
COUNT (DISTINCT CASE NHEN device   'nexus 10' THEN user_id ELSE NULL END) AS nexus_10,
COUNT (DISTINCT CASE WHEN device - 'sansung galaxy 54' THEN user_id ELSE NULL END) AS
samsung_galaxy_54,
COUNT (DISTINCT CASE WHEN device = 'samsung galaxy tablet' THEN user_id ELSE NULL END) AS
samsung_galaxy_tablet,
COUNT (DISTINCT CASE WHEN device - 'sansung galaxy note' THEN user_id ELSE NULL END) AS
samsung_galaxy_note,
COUNT (DISTINCT CASE WHEN device   'lenovo thinkpad' THEN user_id ELSE NULL END) AS
lenovo_thinkpad,
COUNT (DISTINCT CASE WHEN device   'acer aspire notebook'
'acer aspire notebook' THEN user_id ELSE NULL END) AS acer_aspire_notebook,
COUNT (DISTINCT CASE WHEN device - 'asus chronebook' THEN user_id ELSE NULL END) AS
asus_chromebook,
COUNT (DISTINCT CASE WHEN device   'htc one' THEN user_id ELSE NULL END) AS htc_one,
COUNT (DISTINCT CASE WHEN device   nokia lumia 635' THEN user_id ELSE NULL END) AS
nokia_lumia_635,
COUNT (DISTINCT CASE WHEN device = 'mac mini"
'mac mini' THEN user id ELSE NULL END) AS mac_mini,
COUNT (DISTINCT CASE WHEN device = 'hp pavilion desktop' THEN user_id ELSE NULL END) AS
hp_pavilion_desktop,
COUNT (DISTINCT CASE WHEN device = 'dell inspiron desktop' THEN user_id ELSE NULL END) AS
dell_inspiron_desktop
```

```
FROM
        Events
WHERE
            Event type = 'engagement'
GROUP BY
            week_number
ORDER BY
            week_number;
```

SQL Result:

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

week_number	dell_inspiron_notebook	iphone_5	iphone_4s	iphone_5s	ipad_air	windows_surface	macbook_air	macbook_pro	ipad_mini	kir
17	46	65	21	42	27	10	54	143	19	6
18	77	113	46	73	52	10	121	252	30	27
19	83	115	44	79	55	16	112	266	36	21
20	84	125	55	79	59	21	119	256	32	23
21	80	137	45	74	51	17	110	247	23	30
22	92	125	45	71	58	15	145	251	34	21

- **Findings:**

  I. From the table it is clear that most people uses mackook_pre (322 users on 30th week), followed by lenovo thinkpad (220 users, 28th week) and iphone_5 (163 users on 27th week).

- **Insight:**

  I. Focus on **MacBook Pro** for future marketing or product updates, as it shows peak user engagement.

  II. **Seasonal Trends:** The spike in engagement could be linked to external factionors like promotions or events, providing a learning for future campaigns.

9

Case Study 2: Investigating Metric Spike

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

SQL Command:

```
select
100 * sum(case when email_action  = 'email_open'  then 1 else 0 end)/
sum(case when email_action = 'email sent' then 1 else 0 end) as email_open_rate,
100 * sum(case when email  action = 'email clicked' then 1 else 0 end)/
sum(case when email_action = 'email sent' then 1 else 0 end) as email_click_rate
from (select*,
Case
When action IN in ('sent_weekly_digest', 'sent_engagement email') then 'email_ sent'
when action IN in ('email_open') then 'email_open
when action IN in ('email_clickthrough') then 'email_clicked"
end as email_action
from email_events) sub
```

SQL Result:

Result Grid | Filter Rows:

email_open_rate	email_clicked_rate
33.58339	14.78989

- **Findings:**

I. From this table we get the insight that out of all emails sent, around 31.19% were opened and 10.47% were only clicked.

II. Higher open rates may be influenced by factors such as email subject line effectiveness, timing, or user segmentation.

Insight:

I. **Optimize Open Rates:** If certain campaigns show lower open rates, investigate the subject lines, sending time, or user segmentation to increase engagement.

II. **Enhance Click-Through Rate:** Focus on improving content relevance, call-to-action buttons, and personalization of emails to drive more clicks.

Conclusion

Operational Analytics is the underlying foundation for modern organizations. It allows them to base decisions on data. A project, titled "**Operation Analytics and Investigating Metric Spike**," provided me with the role of a Lead Data Analyst at a corporation similar to Microsoft. The overall goal was to extract insights from operational metrics, identify any anomalies, and provide actionable information that helps enhance productivity across all units.

The project was divided into two major case studies:

1) **Job Data Analysis:** Focuses on user interactions over job-related events, which include reviewing, skipping, and reassigning jobs; it encompasses tasks that range from time-series analysis to preferences in language and data verification.

2) **Investigating Metric Spike:** Focuses on analyzing the engagement, growth, retention, and patterns of email interaction; it utilizes complex SQL methodologies to quantify and explain user behavior.

The initiative highlighted the pragmatic use of advanced SQL for solving genuine business problems. Insights generated would be used for operational teams to improve customer engagement strategies and decrease risks tied to metric variability.

Deliverables included complex SQL queries, interpretation of insights, and an all-inclusive report that encapsulated findings and suggestions for operational enhancement.

Key Takeaways

- Data consistency and quality are crucial for reliable analysis.
- Rolling averages and smoothed metrics provide a better understanding of long-term trends compared to daily metrics.
- Personalized user engagement strategies have a measurable impact on retention and activity levels.

Skills Gained:

1). Advanced SQL Skills

This required the implementation of complex SQL techniques to thoroughly analyze very large datasets. I learned window functions like ROW_NUMBER, LAG, ROLLING AVERAGE, Common Table Expressions, and complex joins to handle the data manipulation and aggregation. I also developed skills on optimizing queries by choosing optimal indexing schemes, effective filtering of data, and minimizing computational overhead. I will be able to create queries that are efficient and scalable enough to tackle real-world data-related problems.

2. Data Cleaning and Validation

The integrity of data is very important in analytics, and this project made me realize the importance of identifying and correcting errors. I developed a keen sense of problem identification such as duplicate entries, incorrect data formats, and missing values. Using SQL and preliminary validation tools, I ensured that the datasets were cleaned, accurate, and ready for analysis. This experience further reinforced the need for proper data preparation as a foundation for creating meaningful insights.

3. Time-Series and Trend Analysis

An important element of this project involved the analysis of trends along the temporal dimensions. Activities such as calculating daily and hourly engagement, roll-up averages, and examining retention metrics made me better at analyzing time-series data. I gained the ability to identify, monitor, and

understand seasonality, patterns, and anomalies, which helped me understand changes to key metrics. This skill is particularly relevant for understanding user behavior and operation dynamics.

4. Problem Solving with Data

The examination of metric spikes necessitated a systematic, evidence-based methodology. I acquired the skills to pinpoint fundamental factors that contribute to abrupt fluctuations in engagement or operational performance by integrating statistical analysis with contextual business knowledge. This experience enhanced my analytical capabilities and proficiency in extracting practical insights from intricate datasets, equipping me to address uncertain, real-world challenges adeptly.

5. User Behavior Analysis

The project greatly expanded my understanding of user engagement, growth, and retention. Analyzing sign-up cohorts, device utilization, and email campaign performance gave me the most valuable insights into user behaviours and preferences. This experience made me realize that strategies must be customized to improve customer experience and maximize user retention.

6) Tool Expertise and Workflow Efficiency

I learned how to manage databases using MySQL Workbench, write and debug queries, and visualize results. Integrating Excel for cross verification and presentation tools for reporting further reinforced my ability to do things that handled the end-to-end analytics workflow efficiently.

END