

# Trinity Assignment Three

## Operational Analytics & Investigating Metric Spikes

Aayush Damani

# Table of Contents

1. PROJECT OVERVIEW
2. APPROACH
3. TECH STACK USED
4. RESULTS AND INSIGHTS
5. GOOGLE DRIVE LINK

# Project Description

- The goal of this project is to leverage SQL skills to analyze Microsoft data from the point of view of a Lead Data Analyst.
- The analysis will focus on various aspects such as investigating metric spikes, such as a decrease in weekly user engagement, and operational analytics.
- The insights derived from this analysis will assist different teams within the organization, such as marketing, product, and operations, in making informed decisions.

# Project Overview

## Operational Analytics

**Jobs Reviewed Over Time:** Calculate the number of jobs reviewed per hour for each day in November 2020.

**Throughput Analysis:** Calculate the 7-day rolling average of throughput (number of events per second).

**Language Share Analysis:** Calculate the percentage share of each language in the last 30 days.

**Duplicate Rows Detection:** Identify duplicate rows in the data.

## Investigating Metric Spikes

**Weekly User Engagement:** Measure the activeness of users on a weekly basis.

**User Growth Analysis:** Analyze the growth of users over time for a product.

**Weekly Retention Analysis:** Analyze the retention of users on a weekly basis after signing up for a product.

**Weekly Engagement Per Device:** Measure the activeness of users on a weekly basis per device.

**Email Engagement Analysis:** Analyze how users are engaging with the email service.

# Approach

## **Data Preparation:**

- Set up the database using provided commands and files.
- Ensure data integrity for accurate analysis.

## **Analysis Execution:**

- Complete the tasks identified in the previous slide (Project Overview) in order to derive actionable insights.

## **Query Execution:**

- Write and validate SQL queries in MySQL Workbench.

## **Reporting:**

- Create a PowerPoint report with SQL queries, outputs, and insights.
- Include screenshots and summarize findings for leadership presentation.

# Tech Stack Used



Google Slides



# Jobs Reviewed Over Time

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

## SQL Code

```
## Task A: Jobs Reviewed Over Time
```

```
SELECT
```

```
    EXTRACT(DAY FROM STR_TO_DATE(ds, '%m/%d/%Y')) AS date_day,  
    (COUNT(job_id) / 24) AS no_of_jobs_reviewed_per_hour
```

```
FROM
```

```
    job_data
```

```
GROUP BY date_day;
```

## Results

date_day	no_of_jobs_reviewed_per_h...
30	0.0833
29	0.0417
28	0.0833
27	0.0417
26	0.0417
25	0.0417

## RESULTS

From the results provided in the previous slide, it is clear to see that there are 2 days where the number of jobs reviewed per hour were higher. However, the dataset only contained 6 days of data.

## INSIGHTS

There might be a correlation between certain days of the month and the number of jobs reviewed per hour, but the sample size is too small to draw a conclusion.



# Throughput Analysis

Write an SQL query to calculate the 7-day rolling average of throughput.

## SQL Code

```
16  ## Task B: Throughput Analysis
17
18  •  SELECT
19      event_date,
20      throughput_per_day,
21      AVG(throughput_per_day) OVER (
22          ORDER BY event_date
23          ROWS BETWEEN 6 PRECEDING AND CURRENT ROW
24      ) AS rolling_avg_throughput
25  ⊖ FROM (
26      SELECT
27          DATE(STR_TO_DATE(ds, '%m/%d/%Y')) AS event_date,
28          COUNT(*) AS events_per_day,
29          SUM(time_spent) AS total_time_spent,
30          COUNT(*) / NULLIF(SUM(time_spent), 0) AS throughput_per_day
31      FROM
32          job_data
33      GROUP BY
34          event_date
35      ) AS daily_throughput
36      ORDER BY
37          event_date;
```

## Results

event_date	throughput_per_day	rolling_avg_throughput
2020-11-25	0.0222	0.02220000
2020-11-26	0.0179	0.02005000
2020-11-27	0.0096	0.01656667
2020-11-28	0.0606	0.02757500
2020-11-29	0.0500	0.03206000
2020-11-30	0.0500	0.03505000

## RESULTS

From the results provided in the previous slide, it is clear to see that 28th November had the highest throughput per day, but 30th November had the highest rolling average throughput.

## INSIGHTS

While the rolling average throughput is extremely important in understanding the relativity of daily metrics as compared to the previous days, daily metrics can also be incredibly important in identifying which days had the highest throughput. Therefore, both metrics should be used together in order for the best

# Language Share Analysis

Write an SQL query to calculate the percentage share of each language over the last 30 days.

## SQL Code

```
## Task C: Language Share Analysis

SELECT
    language,
    COUNT(*) AS event_count,
    (COUNT(*) / (SELECT COUNT(*) FROM job_data)) * 100 AS percentage_share
FROM
    job_data
GROUP BY
    language
ORDER BY
    percentage_share DESC;
```

## Results

language	event_count	percentage_share
Persian	3	37.5000
English	1	12.5000
Arabic	1	12.5000
Hindi	1	12.5000
French	1	12.5000
Italian	1	12.5000

## RESULTS

From the results provided in the previous slide, it is clear to see that Persian holds the largest language share, while the rest of the shares are distributed equally amongst several languages.

## INSIGHTS

The jobs could be identified such that Persian is the primary language for that job.

# Duplicate Row Detection

Write an SQL query to display duplicate rows from the job\_data table.

## SQL Code

```
## Task D: Duplicate Count

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

## Results

[illegible]

## RESULTS

From the results provided in the previous slide, it is clear to see that there were no duplicate records found in the job\_data table.

## INSIGHTS

Since there are no duplicates, it is important to maintain this status as duplicates can affect the analysis of our data.

# Weekly User Engagement

Write an SQL query to calculate the weekly user engagement.

## SQL Code

```
## Task A: Weekly User Engagement
```

```
SELECT
    YEAR(STR_TO_DATE(occurred_at, '%d-%m-%Y %H:%i')) AS year,
    WEEK(STR_TO_DATE(occurred_at, '%d-%m-%Y %H:%i')) AS week,
    COUNT(event_type) AS user_engagement
FROM
    events
WHERE
    event_type = 'engagement'
GROUP BY
    YEAR(STR_TO_DATE(occurred_at, '%d-%m-%Y %H:%i')),
    WEEK(STR_TO_DATE(occurred_at, '%d-%m-%Y %H:%i'))
ORDER BY
    year,
    week;
```

## Results

year	week	user_engagement
2014	17	617
2014	18	1498
2014	19	1552
2014	20	1580
2014	21	1612
2014	22	1870
2014	23	1795
2014	24	2014
2014	25	539
2014	26	144
2014	27	46
2014	28	30
2014	29	26
2014	30	8
2014	31	14

## RESULTS

From the results provided in the previous slide, it is clear to see that the weekly user engagement was increasing steadily between weeks 17 and 24, after which the engagement drastically fell.

## INSIGHTS

This suggests that something happened in Week 24/25 that caused a major reduction in engagement, such as a software bug, or better promotions from competitors. This would need to be investigated further.



# User Growth Analysis

Write an SQL query to calculate the user growth for the product.

## SQL Code

```
SELECT year, month, new_active_user,  
       SUM(new_active_user) OVER (ORDER BY year, month ROWS BETWEEN UNBOUNDED PRECEDING AND CURRENT ROW)  
       AS cumulative_active_users  
FROM  
(  
  SELECT  
    YEAR(STR_TO_DATE(activated_at, '%d-%m-%Y %H:%i')) AS year,  
    MONTH(STR_TO_DATE(activated_at, '%d-%m-%Y %H:%i')) AS month,  
    COUNT(DISTINCT user_id) AS new_active_user  
  FROM users WHERE state = "active"  
  GROUP BY year, month) a;
```

## Results

year	month	new_active_user	cumulative_active_users
2013	1	160	160
2013	2	160	320
2013	3	150	470
2013	4	181	651
2013	5	214	865
2013	6	213	1078
2013	7	284	1362
2013	8	316	1678
2013	9	330	2008
2013	10	390	2398
2013	11	399	2797
2013	12	486	3283
2014	1	552	3835
2014	2	525	4360
2014	3	615	4975
2014	4	726	5701
2014	5	779	6480
2014	6	873	7353
2014	7	997	8350
2014	8	1031	9381

## RESULTS

From the results provided in the previous slide, it is clear to see that the number of new users joining has increased over the periods of 2013 and 2014. The growth rate has been steadily increasing every month, with the number of new users increasing every month.

## INSIGHTS

This suggests that the platform is growing and further factors should be investigated further in order to understand how we acquired these new customers successfully. Was the marketing spot on? Did the sales team do something different? Was there a promotional offer?

# Weekly Retention Rate

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

## SQL Code

```
## Task C: Weekly Retention Rate
```

```
SELECT WEEK(STR_TO_DATE(occurred_at, '%d-%m-%Y %H:%i')) AS weeks,  
COUNT(DISTINCT user_id) as no_of_users FROM events  
WHERE event_type="signup_flow" AND event_name="complete_signup"  
GROUP BY weeks  
ORDER BY weeks;
```

## Results

weeks	no_of_users
17	72
18	163
19	185
20	176
21	183
22	196
23	196
24	229
25	32

## RESULTS

From the results provided in the previous slide, it is clear to see that the number of users signing up and completing the sign up flow was steadily increasing until week 25.

## INSIGHTS

Week 25 must be investigated further as there might be a reason why the number of users that signed up and completed their sign up flow was so low.

# Weekly Engagement Per Device

Write an SQL query to calculate the weekly engagement per device.

## SQL Code

```
## Task D: Weekly Engagement Per Device
```

```
SELECT
```

```
    YEAR(STR_TO_DATE(occurred_at, '%d-%m-%Y %H:%i')) AS year,
```

```
    WEEK(STR_TO_DATE(occurred_at, '%d-%m-%Y %H:%i')) AS week,
```

```
    device, COUNT(DISTINCT user_id) AS user_count
```

```
FROM events WHERE event_type = "engagement"
```

```
GROUP BY year, week, device
```

```
ORDER BY year, week, device;
```

## Results

year	week	device	user_count
2014	17	acer aspire desktop	2
2014	17	acer aspire notebook	2
2014	17	amazon fire phone	1
2014	17	asus chromebook	3
2014	17	dell inspiron desktop	1
2014	17	dell inspiron notebook	4
2014	17	hp pavilion desktop	2
2014	17	htc one	2
2014	17	ipad air	1
2014	17	ipad mini	3
2014	17	iphone 4s	3
2014	17	iphone 5	11
2014	17	iphone 5s	5
2014	17	lenovo thinkpad	8
2014	17	mac mini	1
2014	17	macbook air	4
2014	17	macbook pro	13
2014	17	nexus 5	4
2014	17	nexus 7	4
2014	17	nokia lumia 635	2
2014	17	samsung galaxy tablet	2
2014	17	samsung galaxy note	1
2014	17	samsung galaxy s4	7
2014	18	acer aspire desktop	4
2014	18	acer aspire notebook	4
2014	18	amazon fire phone	2
2014	18	asus chromebook	4
2014	18	dell inspiron desktop	3
2014	18	dell inspiron notebook	12
2014	18	hp pavilion desktop	6
2014	18	htc one	2
2014	18	ipad air	8
2014	18	ipad mini	7
2014	18	iphone 4s	4
2014	18	iphone 5	7
2014	18	iphone 5s	8
2014	18	kindle fire	5
2014	18	lenovo thinkpad	28
2014	18	mac mini	1
2014	18	macbook air	18

## RESULTS

From the results provided in the previous slide, it is clear to see that users opted for a variety of different devices, with the Macbook Air having the most number of users (41) in 2014 Week 18, and Week 24.

## INSIGHTS

Macbook Air users could be offered some sort of discount, or special promotion as it is the most used device.

# Email Engagement Analysis

Write an SQL query to calculate the email engagement metrics.

## SQL Code

## Task E: Email Engagement Analytics

```
SELECT user_id, emails_sent, emails_opened, emails_clicked,  
       ROUND(SUM(emails_opened)/SUM(emails_sent),2)*100 AS open_rate,  
       ROUND(SUM(emails_clicked)/SUM(emails_opened),2)*100 AS click_through_rate  
FROM (  
  SELECT user_id,  
         SUM(CASE WHEN `action` = "sent_weekly_digest" THEN 1 ELSE 0 END) AS emails_sent,  
         SUM(CASE WHEN `action` = "email_open" THEN 1 ELSE 0 END) AS emails_opened,  
         SUM(CASE WHEN `action` = "email_clickthrough" THEN 1 ELSE 0 END ) AS emails_clicked  
FROM email_events  
GROUP BY user_id  
  ) AS email_engagement
```

## Results

user_id	emails_sent	emails_opened	emails_clicked	open_rate	click_through_rate
0	17	5	0	29.00	0.00
4	17	5	4	29.00	80.00
8	17	3	1	18.00	33.00
11	17	5	2	29.00	40.00
17	17	4	1	24.00	25.00
19	17	5	1	29.00	20.00
20	17	8	3	47.00	38.00
22	17	7	3	41.00	43.00
30	18	6	1	33.00	17.00
49	17	5	1	29.00	20.00
59	17	5	3	29.00	60.00
64	17	5	2	29.00	40.00
66	17	5	0	29.00	0.00
67	17	5	0	29.00	0.00
78	17	7	4	41.00	57.00
80	17	7	2	41.00	29.00
83	17	4	1	24.00	25.00
86	17	4	3	24.00	75.00
98	18	7	2	39.00	29.00
101	18	9	6	50.00	67.00
108	18	8	0	44.00	0.00
117	17	4	0	24.00	0.00
120	17	4	0	24.00	0.00
124	17	6	2	35.00	33.00
128	17	6	4	35.00	67.00
134	17	9	4	53.00	44.00
136	17	6	2	35.00	33.00
138	17	5	0	29.00	0.00
140	17	3	1	18.00	33.00
145	17	3	0	18.00	0.00
150	17	6	4	35.00	67.00
155	17	4	0	24.00	0.00
163	17	10	3	59.00	30.00
170	18	6	1	33.00	17.00
171	18	6	2	33.00	33.00
172	18	6	2	33.00	33.00
173	18	7	0	39.00	0.00
175	18	6	4	33.00	67.00
179	18	5	0	28.00	0.00
181	18	4	2	22.00	50.00

# RESULTS

From the results provided in the previous slide, it is clear to see that most users have a relatively high click-through rate and open-rate.

# INSIGHTS

This suggests that the emails being sent out are interactive and interesting to the user base. It could be investigated further in order to understand whether the click-through rate made any direct impact towards sales.



# DRIVE LINK

[https://docs.google.com/presentation/d/1\\_K-e0AE1\\_zjRJmPH5OLUB5ho\\_3RBkkyxYPX7QVSaj3k/edit?usp=sharing](https://docs.google.com/presentation/d/1_K-e0AE1_zjRJmPH5OLUB5ho_3RBkkyxYPX7QVSaj3k/edit?usp=sharing)



THANK YOU FOR YOUR TIME.

