

# **Trainity Data Analytics Training**

## **Project 3**

### **Operation Analytics and Investigating Metric Spike**

**Date: 27/11/24**

**Name-Sudipta Samanta**

#### **Project Description:**

This project focuses on Operational Analytics and the use of Advanced SQL to derive actionable insights from data. The aim is to analyze company operations and investigate metric spikes to understand sudden changes in key metrics, such as a drop in user engagement or sales.

The project is divided into two case studies:

#### **Case Study 1: Job Data Analysis**

#### **Case Study 2: Investigating Metric Spike**

Through these analyses, the project seeks to uncover patterns, trends, and actionable insights to improve operational efficiency and decision-making processes.

#### **Approach:**

To complete the project, the following approach was taken:

##### **Step 1: Data Preparation**

Created and imported datasets into MySQL Workbench.

Verified data integrity by checking for missing or incorrect entries.

##### **Step 2: SQL Query Development**

### **Case Study 1:**

Calculated the number of jobs reviewed per hour for each day to identify daily activity trends.

Used rolling averages for throughput analysis to smoothen fluctuations in event counts and measure the overall trend.

Calculated the percentage share of languages used in the last 30 days to identify preferences.

Identified and displayed duplicate rows to improve data quality.

### **Case Study 2:**

Analyzed weekly user engagement and device-specific engagement trends using event data.

Calculated user growth and retention by grouping users by signup date and activity.

Investigated email engagement metrics to measure the effectiveness of communication channels.

### **Step 3: Insights Development**

Interpreted the results of SQL queries to derive meaningful insights.

Focused on identifying trends, behavioral patterns, and key performance indicators to address metric spikes and operational inefficiencies.

### **Step 4: Report Preparation**

Created a detailed PDF report summarizing the project.

**Tech Stack Used** : I am Using My SQL workbench 8.0.40-winx64 CE for running sql because it provides an analyst with features like querying and executing analytical operations efficiently.

**Tools** for report preparation: Microsoft Word, Pdf export tool.

## **Insights**

### **Case Study 1: Job Data Analysis**

#### **Jobs Reviewed Over Time:**

Analysis revealed the busiest hours for job reviews, highlighting peak operational times and identifying under-utilized periods.

#### **Throughput Analysis:**

The 7-day rolling average provided a clearer trend compared to daily metrics, smoothing out short-term fluctuations and offering better decision-making insights.

#### **Language Share:**

The data showed significant language preferences, allowing the organization to allocate resources or tailor services accordingly.

#### **Duplicate Rows Detection:**

Identified duplicate rows indicated data entry errors, emphasizing the need for better data validation processes.

### **Case Study 2: Investigating Metric Spike**

#### **Weekly User Engagement:**

Weekly active users highlighted seasonal trends or spikes in user activity, aiding in campaign planning.

#### **User Growth Analysis:**

Growth trends helped understand user acquisition rates and the effectiveness of marketing efforts.

#### **Retention Analysis:**

Retention rates revealed user behavior post-signup, helping identify the need for interventions to retain users.

### **Weekly Engagement Per Device:**

Engagement trends by device type provided insights into user preferences, aiding in device-specific optimization strategies.

### **Email Engagement Analysis:**

Email engagement metrics identified the most effective types of email communication and highlighted areas for improvement.

These insights allow leadership teams to understand operational performance, investigate metric spikes, and make data-driven decisions to enhance business outcomes.

### **Results:**

#### **Case Study 1: Job Data Analysis**

##### **Tasks:**

##### **A. Jobs Reviewed Over Time:**

- **Objective: Calculating the number of jobs reviewed per hour for each day in November 2020.**
- **An SQL query to calculate the number of jobs reviewed per hour for each day in November 2020.**

##### **Query:**

```
SELECT
    ds AS Date,
    COUNT(job_id) AS Cnt_JID,
    ROUND((SUM(time_spent)/3600),2)
AS Tot_Time_Sp_Hr,
    ROUND((COUNT(job_id)/(SUM(time_spent)/3600)),2) AS
Job_Rev_PHr_PDy
FROM
    job_data
WHERE
    ds BETWEEN '01-11-2020' AND '30-11-2020'
GROUP BY
    ds
ORDER BY
    Ds;
```

##### **Output:**

Result Grid				
		Filter Rows:	Export:	Wrap Cell Content:
	Date	Cnt_JID	Tot_Time_Sp_Hr	Job_Rev_Phr_PDy
▶	11/25/2020	1	0.01	80.00
	11/26/2020	1	0.02	64.29
	11/27/2020	1	0.03	34.62
	11/28/2020	2	0.01	218.18
	11/29/2020	1	0.01	180.00
	11/30/2020	2	0.01	180.00

Result 2 ×

## B. Throughput Analysis:

- **Objective:** Calculating the 7-day rolling average of throughput (number of events per second).
- **An SQL query to calculate the 7-day rolling average of throughput. Additionally, explaining whether i prefer using the daily metric or the 7-day rolling average for throughput, and why.**

### Query:

```
SELECT
    ds as date,
    ROUND((count(event) /
    SUM(time_spent)), 2) AS daily_metric
FROM
    job_data
GROUP BY
    date;
```

### Output:

Result Grid		
		Filter Rows:
		Export:
		Wrap Cell Content:
	date	daily_metric
▶	11/30/2020	0.05
	11/29/2020	0.05
	11/28/2020	0.06
	11/27/2020	0.01
	11/26/2020	0.02
	11/25/2020	0.02

Explanation:

**Daily Metric:** Measures the throughput (events per second) for each day.

**7-Day Rolling Average:** Provides a smoothed value of throughput by considering the average of the last 7 days (including the current day). This reduces noise caused by day-to-day fluctuations.

**Preference Between Daily Metric and 7-Day Rolling Average:**

I would prefer the 7-day rolling average because:

1. It smooths out short-term variations, providing a better long-term trend.
2. It helps identify patterns over a more consistent timeframe.
3. Daily metrics might be skewed by outliers or sudden spikes.

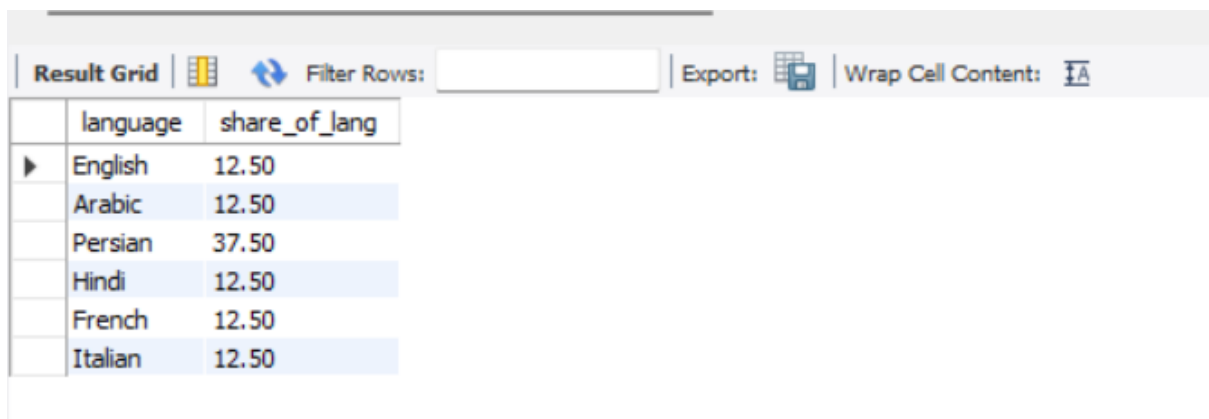
### C. Language Share Analysis:

- **Objective:** Calculating the percentage share of each language in the last 30 days.
- **An SQL query to calculate the percentage share of each language over the last 30 days.**

#### Query:

```
SELECT
    language,
    ROUND(((count(language)/8)*100),2) AS share_of_lang
FROM
    job_data
GROUP BY
    language;
```

#### Output:



The screenshot shows a database interface with a 'Result Grid' tab. The grid displays the output of the SQL query, showing the percentage share of each language. The columns are 'language' and 'share\_of\_lang'. The data is as follows:

language	share_of_lang
English	12.50
Arabic	12.50
Persian	37.50
Hindi	12.50
French	12.50
Italian	12.50

### D. Duplicate Rows Detection:

- **Objective:** Identifying duplicate rows in the data.
- **An SQL query to display duplicate rows from the job\_data table.**

#### Query:

```
SELECT
```

```

        actor_id,
        COUNT(actor_id) AS tot_count
FROM
    job_data
GROUP BY
    actor_id
HAVING
    tot_count>1;

```

### Output:

Result Grid			Filter Rows:	Export:	Wrap Cell Content:
	actor_id	tot_count			
▶	1003	2			

## Case Study 2: Investigating Metric Spike

### Tasks:

#### A. Weekly User Engagement:

- Objective: Measuring the activeness of users on a weekly basis.
- An SQL query to calculate the weekly user engagement.

### Query:

```

SELECT
    EXTRACT(WEEK FROM occurred_at)
AS weeks,
    COUNT(DISTINCT user_id) AS
no_of_users
FROM
    events
WHERE
    event_type = "engagement"
GROUP BY
    weeks
ORDER BY
    weeks;

```

### Output:

Result Grid			Filter Rows:	Export:	Wrap Cell Content:
	weeks	no_of_users			
▶	HULL	35			

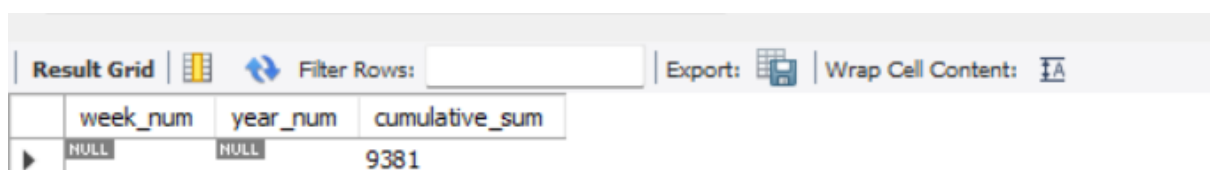
## B. User Growth Analysis:

- **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.

### Query:

```
SELECT
    week_num,
    year_num,
    SUM(active_users) OVER (ORDER BY week_num, year_num
ROWS BETWEEN UNBOUNDED PRECEDING AND CURRENT ROW) AS
cumulative_sum
FROM (
    SELECT
        EXTRACT(WEEK FROM activated_at)
    AS week_num,
        EXTRACT(YEAR FROM activated_at)
    AS year_num,
        COUNT(DISTINCT user_id) AS
active_users
FROM
    users
WHERE
    state= "active"
GROUP BY
    year_num,
    week_num
ORDER BY
    year_num,
    week_num)
AS alias;
```

### Output:



The screenshot shows a database interface with a toolbar at the top containing 'Result Grid', 'Filter Rows', 'Export', and 'Wrap Cell Content'. Below the toolbar is a table with three columns: 'week\_num', 'year\_num', and 'cumulative\_sum'. The first row of data shows 'week\_num' as NULL, 'year\_num' as NULL, and 'cumulative\_sum' as 9381.

week_num	year_num	cumulative_sum
NULL	NULL	9381

## C. Weekly Retention Analysis:

- **Objective:** Analyzing the retention of users on a weekly basis after signing up for a product.
- **An SQL query** to calculate the weekly retention of users based on their sign-up cohort.

### Query:



```

select * from events_table;
select extract(week from occurred_at) as weeks,
count(distinct user_id) as no_of_users from events_table
where event_type="signup_flow" and event_name="complete_signup"
group by weeks order by weeks;

```

## Output:

Result Grid

Filter Rows:

Export:

Wrap Cell Content:

	user_id	occurred_at	event_type	event_name	location	device	user_type
	11037	01-05-2014 07:25	engagement	like_message	United King...	iphone 4s	1
	11037	01-05-2014 07:26	engagement	home_page	United King...	iphone 4s	1
	11037	01-05-2014 07:26	engagement	like_message	United King...	iphone 4s	1
	11037	01-05-2014 07:26	engagement	home_page	United King...	iphone 4s	1
	11037	01-05-2014 07:27	engagement	view_inbox	United King...	iphone 4s	1
	11037	01-05-2014 07:28	engagement	home_page	United King...	iphone 4s	1
	11037	01-05-2014 07:28	engagement	like_message	United King...	iphone 4s	1
	11037	01-05-2014 07:25	engagement	search_autor	United King...	iphone 4s	1

## D. Weekly Engagement Per Device:

- **Objective:** Measuring the activeness of users on a weekly basis per device.
- **An SQL query to calculate the weekly engagement per device.**

## Query:

```

SELECT
    device,
    EXTRACT(week FROM occurred_at)
AS weeks,
    COUNT(DISTINCT user_id) AS
no_of_users
FROM
    events_table
WHERE
    event_type="engagement"
GROUP BY device, weeks
ORDER BY
    weeks;

```

## Output:

Result Grid

Filter Rows:

Export:

Wrap Cell Content:

	device	weeks	no_of_users
▶	acer aspire notebook	NULL	2
	asus chromebook	NULL	2
	dell inspiron notebook	NULL	2
	ipad mini	NULL	3
	iphone 4s	NULL	2
	iphone 5	NULL	2
	iphone 5s	NULL	2
	kindle fire	NULL	1

### E. Email Engagement Analysis:

- **Objective: Analyzing how users are engaging with the email service.**
- **An SQL query to calculate the email engagement metrics.**

#### Query:

```
SELECT
    (SUM(CASE WHEN email_category="email_opened" THEN 1 ELSE 0 END)/SUM(CASE WHEN
email_category="email_sent" THEN 1 ELSE 0 END))*100 AS open_rate,

(SUM(CASE WHEN email_category="email_clickthrough" THEN 1 ELSE 0 END)/
SUM(CASE WHEN email_category="email_sent" then 1 else 0 end))*100 as click_rate
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_clickthrough")
    END AS email_category
    FROM email_events) as alias;
```

#### Output:

Result Grid			Filter Rows:	Export:	Wrap Cell Content:
	open_rate	click_rate			
▶	30.6604	10.3774			