# **Project Summary: Operational Analytics**

**Objective:** Operational Analytics plays a vital role in analysing a company's operations from start to finish. This process helps pinpoint areas for improvement within the organisation. As a Data Analyst, I work closely with various teams, such as operations, support, and marketing, to extract valuable insights from the data they collect. One of my key responsibilities is investigating metric spikes—understanding and explaining sudden changes in key metrics like daily user engagement or sales dips.

In this project, I take on the role of a Lead Data Analyst at a company like Microsoft. I am provided with various datasets and tables, and my task is to derive insights from this data to answer questions posed by different departments within the company. Using advanced SQL skills, I analyse the data and provide valuable insights to help improve the company's operations and understand sudden changes in key metrics.

# Case Study 1: Job Data Analysis

Table: job\_data

#### Columns:

- job\_id: Unique identifier of jobs
- actor\_id: Unique identifier of actor
- event: Type of event (decision/skip/transfer)
- language: Language of the content
- time\_spent: Time spent reviewing the job (in seconds)
- org: Organization of the actor
- ds: Date (yyyy/mm/dd format, stored as text)

#### Tasks:

#### 1. Jobs Reviewed Over Time

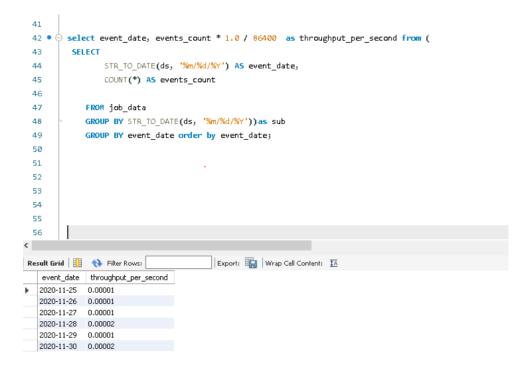
- Objective: Calculate the number of jobs reviewed per hour for each day in November 2020.
- Task: Write an SQL query to calculate the number of jobs reviewed per hour for each day in November 2020.



## **Throughput Analysis**

- **Objective:** Calculate the 7-day rolling average of throughput (number of events per second).
- Task: Write an SQL query to calculate the 7-day rolling average of throughput.
   Additionally, explain whether I prefer using the daily metric or the 7-day rolling average for throughput, and why.

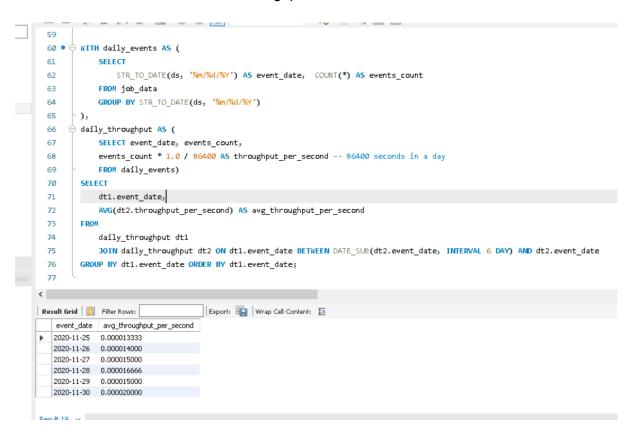
### Daily throughput:



# Interpretation

# 1. Daily Metric Analysis:

- **Consistent Throughput**: For most days, the throughput remains consistent at 0.00001 events per second.
- **Spike on 2020-11-28 and 2020-11-30**: There are noticeable spikes on 2020-11-28 and 2020-11-30, where the throughput doubles to 0.00002.



# Interpretation

### 7-Day Rolling Average Analysis:

- **Smoothed Trends**: The rolling average shows a smoother trend without abrupt changes. It gradually increases from 0.000013333 on 2020-11-25 to 0.000020000 on 2020-11-30.
- Less Impact of Spikes: The rolling average mitigates the impact of the daily spikes observed on 2020-11-28 and 2020-11-30, providing a more gradual increase in throughput.

Given these observations, I prefer using the 7-day rolling average for throughput because:

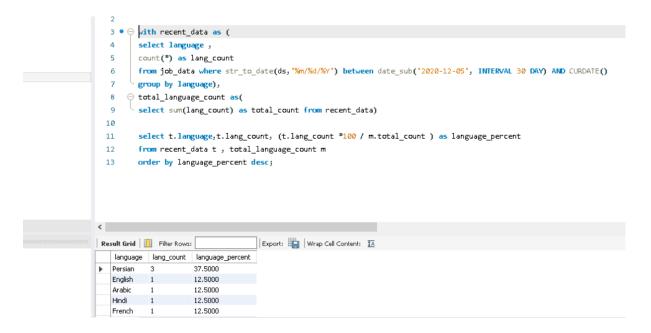
- Smoother Trends: It provides a clearer view of overall trends and performance.
- Noise Reduction: It minimises the impact of daily noise and anomalies, making it easier to detect genuine trends.

 Better Decision Making: Long-term trends are more reliable for making strategic decisions, as they are less likely to be influenced by short-term irregularities.

However, the daily metric is still valuable for identifying and addressing specific days with significant changes, so both metrics have their uses depending on the context and needs of the analysis.

### **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.



Let's assume the current date is '2020-12-05' as the data available in this table is of 2020 and to calculate the percentage share of each language in the last 30 days won't make any sense if take the current date which is year 2024 month 7 july

This query filters the data for last 30 days

Count the event per language

Calculate the total event within the 30 days

Calculate the percentage share of each language

## **Duplicate Rows Detection**

- Objective: Identify duplicate rows in the data.
- Task: Write an SQL query to display duplicate rows from the job\_data table.

There is no duplicate data, this query group by each column value and count the occurrence of any set of rows similar occurring more than one that is duplicate which will be displayed as of now there is no duplicate in the given data.

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

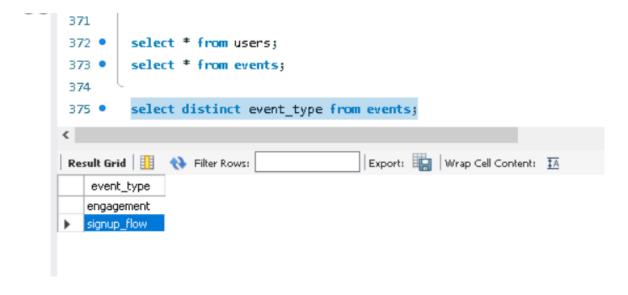
#### Tables:

- users: Contains one row per user, with descriptive information about that user's account.
- events: Contains one row per event, where an event is an action that a user has taken (e.g., login, messaging, search).
- email\_events: Contains events specific to the sending of emails.

#### Tasks:

#### 1. Weekly User Engagement

- Objective: Measure the activeness of users on a weekly basis.
- Task: Write an SQL query to calculate weekly user engagement.



I looked for different event type in the events table

```
416
417 •
             SELECT
418
             occurred_at,
419
             YEAR(occurred at) AS year,
             MONTH(occurred_at) AS month,
420
             WEEK(occurred_at, 1) AS year_week, -- Week number with Monday as the first day of the we
421
422
             DAY(occurred_at) AS day,
             DAYOFWEEK(occurred_at) AS day_of_week, -- 1 = Sunday, 2 = Monday, ..., 7 = Saturday
423
424
             HOUR(occurred_at) AS hour,
             MINUTE(occurred_at) AS minute,
425
426
             SECOND(occurred_at) AS second,
             DATE_FORMAT(occurred_at, '%Y-%m-%d %H:%i:%s') AS formatted_date
427
428
         FROM
429
             events;
430
<
                                                                                     4
Export: Wrap Cell Content: A Fetch rows:
                                                   day_of_week hour
    occurred_at
                                             day
                                                                    minute second formatted_date
   2014-05-02 11:02:00
                           5
                                                                                   2014-05-02 11:02:00
                                  18
                                                  6
                                                              11
                                                                           0
   2014-05-02 11:02:00 2014 5
                                  18
                                            2
                                                  6
                                                                           0
                                                                                   2014-05-02 11:02:00
   2014-05-02 11:03:00 2014 5
                                  18
                                                  6
                                                              11
                                                                    3
                                                                           0
                                                                                   2014-05-02 11:03:00
                                            2
                                                              11
   2014-05-02 11:04:00 2014 5
                                  18
                                                  6
                                                                   4
                                                                           0
                                                                                   2014-05-02 11:04:00
   2014-05-02 11:03:00 2014 5
                                  18
                                                                           0
                                                                                   2014-05-02 11:03:00
                                                  6
                                                              11
                                                                    3
   2014-05-02 11:03:00 2014 5
                                  18
                                            2
                                                              11
                                                                    3
                                                                           0
                                                                                   2014-05-02 11:03:00
   2014-05-01 09:59:00
                    2014
                                  18
                                                                    59
                                                                           0
                                                                                   2014-05-01 09:59:00
   2014-05-01 10:00:00 2014 5
                                                  5
                                                              10
                                                                                   2014-05-01 10:00:00
                                  18
                                                                    0
                                                                           0
   2014-05-01 10:00:00 2014
                                                                                   2014-05-01 10:00:00
                                  18
                                                              10
                                                                    Π
                                                                           Π
Result 23 🗶
```

To find weekly user engagement i need find out the weekyear and this query helped me to understand the date format very well

```
435
         select f.user_id , yearweek(occurred_at,1) as year_week,
436 •
         count(e.event_name) as event_count from users f , events e
437
         where f.user_id = e.user_id group by yearweek(e.occurred_at,1),f.user_id
438
         order by year_week ;
439
440
441
442
         select * from events where user_id = 8
443 •
444
445
446
                                         Export: Wrap Cell Content: 🔣 | Fetch rows:
event_count
          year_week
          201418
                    2
   22
          201418
                    8
   66
          201418
                    14
                    22
   163
          201418
   172
          201418
                    15
   209
          201418
                    9
   232
                    47
          201418
   239
          201418
                    9
  253
          201418
Result 29 🗶
```

This gives the weekly engagement of user here 201418 (2014 is year and 18 is the 18th week of this year)

### **User Growth Analysis**

- Objective: Analyze user growth over time for a product.
- Task: Write an SQL query to calculate user growth.



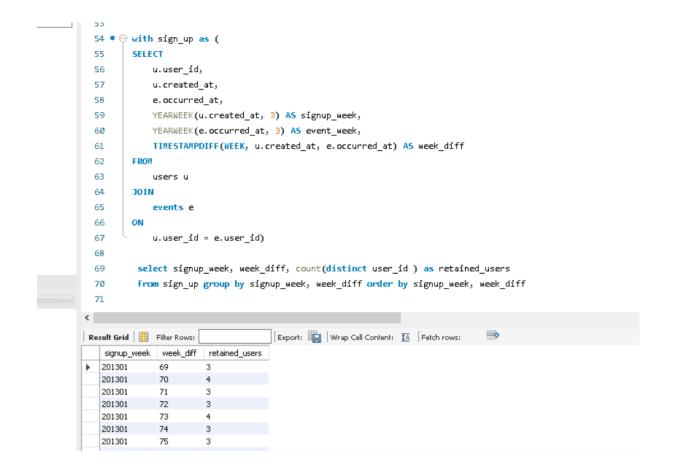
Using LAG(COUNT(user\_id)) OVER (ORDER BY DATE\_FORMAT(created\_at, '%Y-%m')), we get the count of users from the previous month. IFNULL is used to handle cases where there is no previous month

User growth is calculated previous month user - the current month user

This gives the clear details about user growth of every month

### **Weekly Retention Analysis**

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



# Interpretation

- signup\_week: The week number when the users signed up.
- week\_diff: The difference in weeks between the sign-up date and the event date.
- retained\_users: The count of distinct users who performed an event after a certain number of weeks since signing up.

## **Weekly Engagement Per Device**

- Objective: Measure user activeness on a weekly basis per device.
- Task: Write an SQL query to calculate weekly engagement per device.

```
75 👪
        select * from events;
76
77 •
        SELECT count(distinct user_id) as active_users,
             YEARWEEK(occurred_at, 1) AS event_year_week,
78
79
             device
        FROM
30
31
             events
32
        GROUP BY
33
             YEARWEEK(occurred_at, 1), device
34
        ORDER BY
35
             event year week;
                                              Export: Wrap Cell Content: IA
sult Grid 🔡
              Filter Rows:
              event_year_week
                                device
 active_users
                                acer aspire desktop
 10
              201418
 21
              201418
                                acer aspire notebook
 4
                                amazon fire phone
              201418
                                asus chromebook
 23
              201418
                                dell inspiron desktop
 21
              201418
 49
              201418
                                dell inspiron notebook
sult 14 🛛 🗙
```

- user\_count: The number of distinct users who were active in each week.
- event\_date: The year and week number during which the users were active.
- device: The type of device used by the users.

## **Email Engagement Analysis**

- Objective: Analyze how users are engaging with the email service.
- Task: Write an SQL query to calculate the email engagement metrics.

```
95
         SELECT
             COUNT(*) AS total_actions,
 96
 97
             COUNT(DISTINCT user_id) AS unique_users,
 98
             user_type,
 99
             action,
100
             COUNT(*) AS action_count,
             DATE(occurred_at) AS date
101
         FROM
102
103
             email_events
         GROUP BY
104
105
             user_type,
106
             action,
107
             DATE(occurred_at)
108
         ORDER BY
109
             DATE(occurred_at), user_type, action;
110
                                                                                         Export: 📳 | Wrap Cell Content: 🏗 | Fetch rows:
   total_actions
                                                            action_count date
               unique_users user_type action
                                                                        2014-05-01
               18
                                      email_clickthrough
                                                            18
   18
                           1
   50
                                                            50
                                                                        2014-05-01
                                      email_open
                                      sent_weekly_digest
                                                                        2014-05-01
   158
               158
                                                            158
   14
               14
                                     email_clickthrough
                                                            14
                                                                       2014-05-01
   33
               33
                                      email_open
                                                            33
                                                                        2014-05-01
               2
                                      sent_reengagement_email
                                                            2
                                                                        2014-05-01
  119
               119
                                      sent_weekly_digest
                                                            119
                                                                        2014-05-01
                                      email_clickthrough
  29
               29
                           3
                                                            29
                                                                        2014-05-01
   62
               62
                           3
                                      email_open
                                                            62
                                                                        2014-05-01
  5
                           3
                                      sent reengagement email
                                                                        2014-05-01
```

By using these clauses together, the query provides a comprehensive set of email engagement metrics, including:

- The total number of actions.
- The number of unique users interacting with the service.
- The breakdown of actions by user type and action type.
- The number of actions on a daily basis.