

Operation Analytics and Investigating Metric Spike

Project Description:

Operation Analytics is a crucial process that involves analyzing a company's end to end operations. This analysis helps identify areas for improvement within the company. As a Data Analyst, I will work closely with various teams, such as operations, support and marketing, helping them derive valuable insights from the data collected.

One of the key aspects of Operation Analytics is investigating metric spikes. This involves understanding and explaining sudden changes in **key metrics**, such as dip in daily user engagement or a drop in sales. As a Data Analyst these questions needs to be answered daily, making it crucial to understand how to investigate these **metric spikes**.

In this project, I will work as a Lead Data Analyst at a company like Microsoft, where I'll be provided with various datasets and tables, and my task will be to derive insights from the data to answer questions posed by different departments within the company. My goal is to use my advanced SQL skills to analyze the data and provide valuable insights that can help improve the company's operations and understand sudden changes in key metrics.

Approach:

For the project, the data is collected from various departments and combined into various tables, ensuring data integrity and quality. For the **Case Study-1**, we have a small dataset so, it can directly be imported using the **Table Data Import Wizard**, however, for **Case Study-2**, since we have a large dataset. So, we need to make use of **load data infile** to import the dataset. After importing the dataset, we see that the column containing the dates are stored as text and not datetime, so we need to alter them and change their datatype.

After this, through SQL queries, we will explore and analyze the patterns, trends in the data. We will find out the key performance metrics and their change with time. Based on the findings, the insights and recommendations will be made for various departments to make data-driven informed decisions for process improvement, increase in sales, user engagement and growth of the company.

Tech-Stack Used:

For the project, **MYSQL Workbench 8.0** which is owned by Oracle was used as this is the most widely used database software in the industry. It has a user-friendly GUI tool for MYSQL which allows to analyze the data and work on it easily. Also, it allows to virtually create physical database design models that can easily be transitioned into MYSQL databases.

CASE STUDY-1: Job Data Analysis

Task 1: Jobs Reviewed Over Time

```
22
23 • select * from job_data;
24 • SELECT
25     DATE_FORMAT(ds, '%d-%M-%Y') AS Day,
26     round(COUNT(job_id) / sum(time_spent)*3600) AS No_of_Jobs_Reveiwed
27 FROM
28     job_data
29 WHERE
30     ds BETWEEN '2020-11-01' AND '2020-11-30'
31 GROUP BY 1
32 ORDER BY 1;
33
```

Day	No_of_Jobs_Reveiwed
25-November-2020	80
26-November-2020	64
27-November-2020	35
28-November-2020	218
29-November-2020	180
30-November-2020	180

Here, we see the total number of jobs reviewed per hour for each day in November 2020. The maximum jobs reviewed in a day was on **28-November-2020** which is **218**. The number of jobs being reviewed has the lowest count of **35** and highest being **218**. We need to find out the reason for these differences and work on it to get a consistent and better result, thus improving overall performance.

Task 2: Throughput Analysis

```
41
42 • select round((count(event)/sum(time_spent)),2) as weekly_throughput
43 from job_data;
44
45 • select ds,
46         throughput,
47         avg(throughput)over(order by ds rows between 6 preceding and current row) as Daily_throughput
48 from(
49 select ds, count(event)/sum(time_spent) as throughput from job_data
50 where ds between '2020-11-24' and '2020-11-30'group by ds order by ds ) as a;
51
```

Result Grid

Weekly_throughput
0.03

ds	throughput	Daily_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

Here, we find out the 7-day rolling average of throughput (number of events per second). This provides us to analyze the trends in the metrics on a weekly basis. Also, found out the throughput on a daily basis.

It is preferable to calculate the throughput on a weekly basis as this gives us the clearer picture of the performance trends without getting into the complexity of analyzing trend on a daily basis.

We see that the weekly throughput for the data is **0.03**.

Task 3: Language Share Analysis

```
58 • SELECT
59     language,
60     COUNT(job_id) AS jobs,
61     ROUND(100 * COUNT(job_id) / (SELECT
62         COUNT(job_id)
63     FROM
64         job_data),
65     2) AS percentage_share
66 FROM
67     job_data
68 GROUP BY language;
```

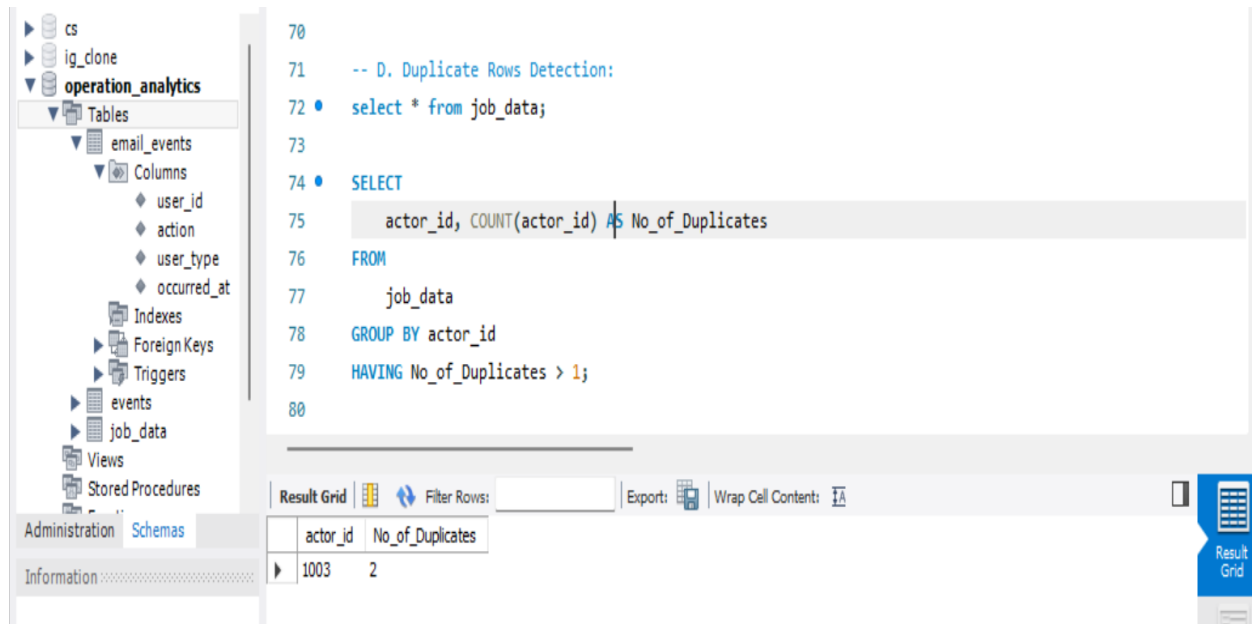
Result Grid | Filter Rows: | Export: | Wrap Cell Content: |

	language	jobs	percentage_share
▶	English	1	12.50
	Arabic	1	12.50
	Persian	3	37.50
	Hindi	1	12.50
	French	1	12.50
	Italian	1	12.50

In this, we find out the percentage share of each language in the last 30 days.

Based upon the findings, we conclude that **Persian** is the most used language with a share of **37.50%**. Also, we see that all the other languages have the same amount of share which is **12.50%**. We can include more language specific content / features to increase the overall user engagement.

Task 4: Duplicate Rows Detection



The screenshot displays a database management interface. On the left, a tree view shows the database structure, including a schema named 'operation_analytics' which contains a table 'job_data'. The main area shows a SQL query being executed:

```
70
71 -- D. Duplicate Rows Detection:
72 • select * from job_data;
73
74 • SELECT
75     actor_id, COUNT(actor_id) AS No_of_Duplicates
76 FROM
77     job_data
78 GROUP BY actor_id
79 HAVING No_of_Duplicates > 1;
80
```

Below the query editor, a 'Result Grid' is visible, showing the results of the query:

actor_id	No_of_Duplicates
1003	2

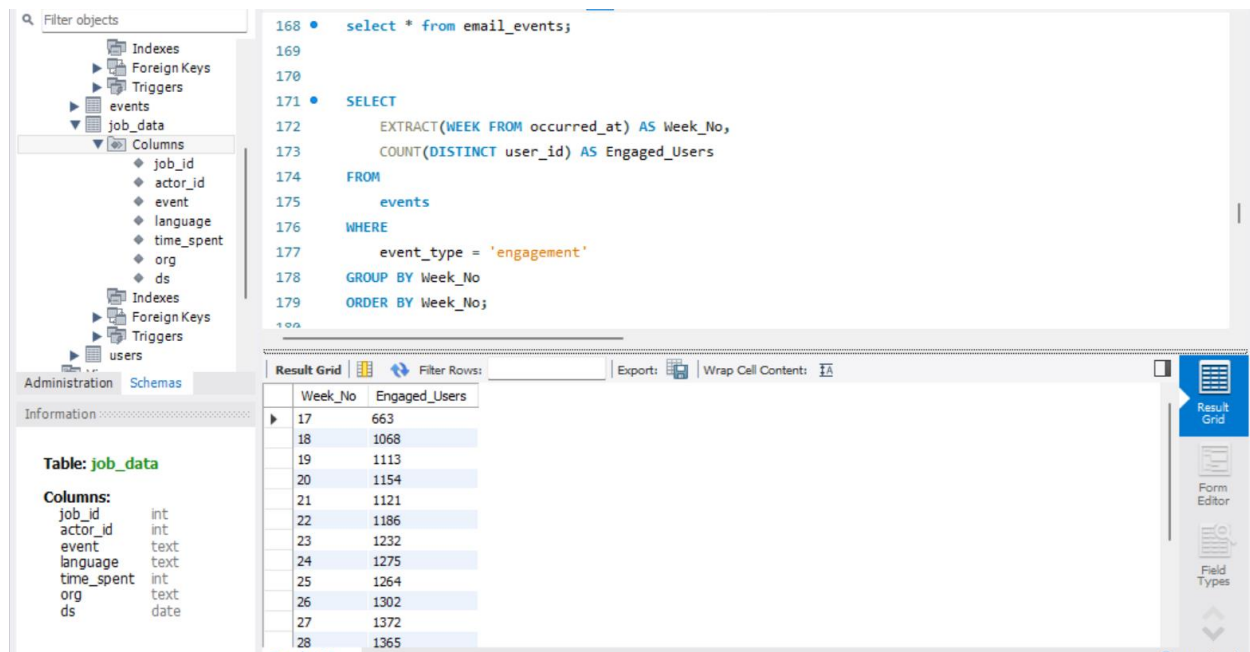
The interface also includes a toolbar with options like 'Filter Rows', 'Export', and 'Wrap Cell Content'.

In this, we checked for any duplicate rows in the data. For this, we used **actor_id** column.

We see that there is one duplicate row present in the row. Data validation methods can be used here to ensure there are no duplicate data in future to get an accurate analysis and result.

CASE STUDY-2: Investigating Metric Spike

Task 1: Weekly User Engagement



Here, we see the total number of users engaged per week which helps us understand the user's activeness on a weekly basis.

We see that for the **week no 30**, the user engagement was at its peak with a total count of **1467**.

We can check for the reason of this increase whether it is due to a marketing campaign, updates or any other events and then plan our future strategies to engage more users.

Task 2: User Growth Analysis

The screenshot displays a database management interface. On the left, a tree view shows the database schema with tables like job_data, users, and others. The main area shows a SQL query being executed. The query is a window function that calculates the total active users over time, ordered by year and week number. The results are shown in a grid with columns: year, week_no, no_of_active_users, and total_active_users. The data shows a steady increase in users from 2013 week 0 to week 11.

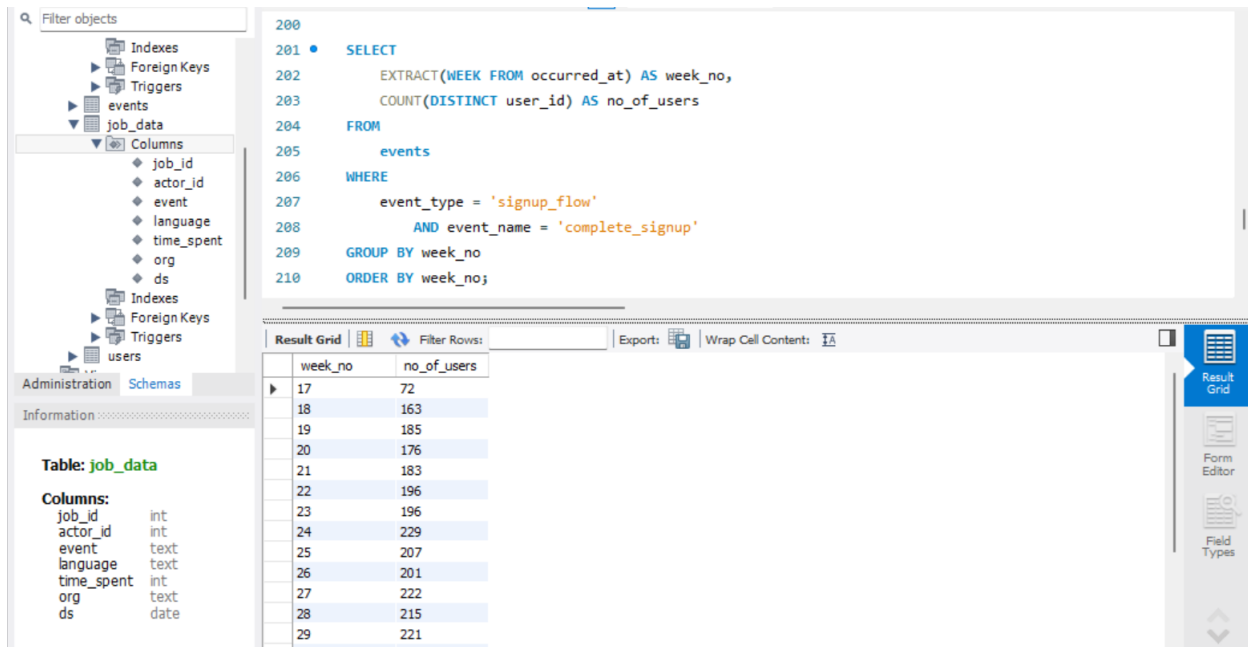
```
188
189 • select year,week_no,no_of_active_users,sum(no_of_active_users)
190 over(order by year, week_no rows between unbounded preceding and current row)
191 as total_active_users from(
192 select extract(year from activated_at) as year, extract(week from activated_at) as week_no,
193 count(distinct user_id) as no_of_active_users
194 from users group by year,week_no order by year,week_no)a;
```

year	week_no	no_of_active_users	total_active_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
2013	7	42	295
2013	8	34	329
2013	9	43	372
2013	10	32	404
2013	11	31	435

Here, we are analyzing the growth of users for a product over time.

We see that the user growth has been positive and the number of users has been increasing over time. We can check for various factors driving this growth whether it is a marketing campaign, any updates etc and replicate it for the overall growth of the company.

Task 3: Weekly Retention Analysis



The screenshot displays a database management interface. On the left, a 'Filter objects' pane shows a tree view of database objects, including 'job_data' and 'users'. The 'job_data' table is selected, and its columns are listed: job_id, actor_id, event, language, time_spent, org, and ds. The 'users' table is also visible. The main area shows a SQL query in a text editor, which is executed. The query is as follows:

```
200
201 • SELECT
202     EXTRACT(WEEK FROM occurred_at) AS week_no,
203     COUNT(DISTINCT user_id) AS no_of_users
204 FROM
205     events
206 WHERE
207     event_type = 'signup_flow'
208     AND event_name = 'complete_signup'
209 GROUP BY week_no
210 ORDER BY week_no;
```

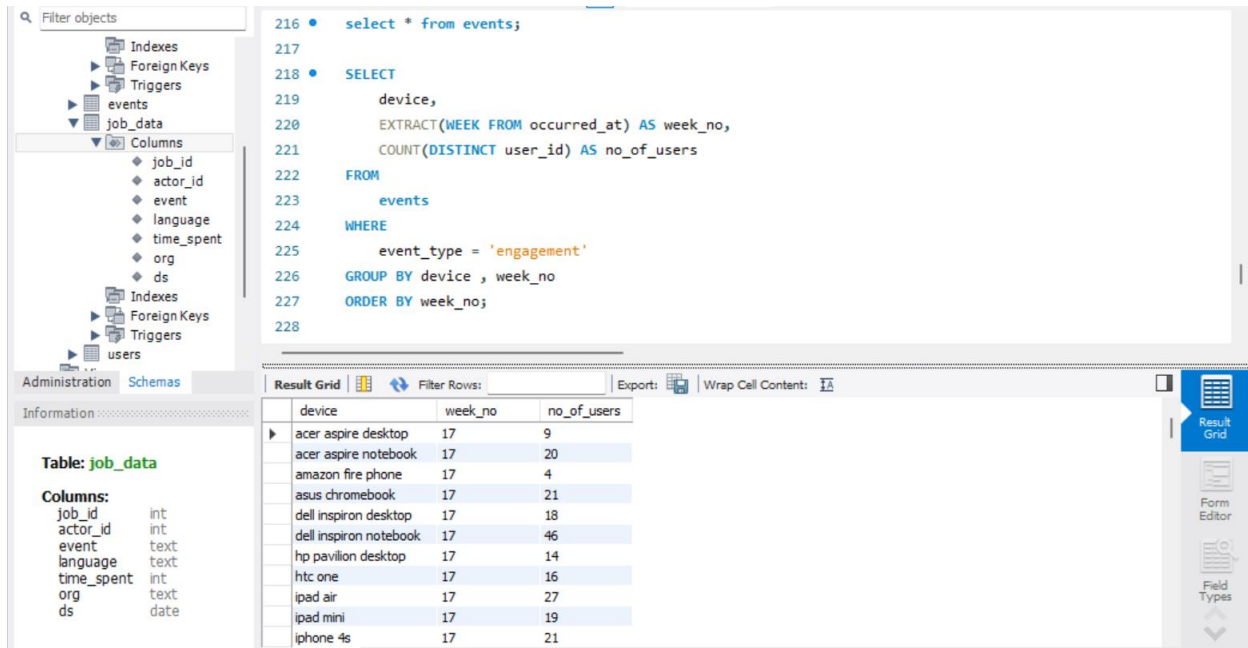
Below the query editor, a 'Result Grid' shows the results of the query. The grid has two columns: 'week_no' and 'no_of_users'. The results are as follows:

week_no	no_of_users
17	72
18	163
19	185
20	176
21	183
22	196
23	196
24	229
25	207
26	201
27	222
28	215
29	221

Here, we are analyzing the weekly retention of users after signing up for a product.

We see from week **17** to **35**, a **rise** and **drop** in the retention of users. We can check for factors resulting in the drop and use retention strategies, check for points where the users are dropping off and improve user engagement and experience to have more users.

Task 4: Weekly Engagement Per Device



The screenshot displays a database management interface. On the left, a 'Filter objects' pane shows a tree view of database objects, including 'job_data' and 'users'. The main area shows a SQL query being executed. The query is as follows:

```
216 • select * from events;
217
218 • SELECT
219     device,
220     EXTRACT(WEEK FROM occurred_at) AS week_no,
221     COUNT(DISTINCT user_id) AS no_of_users
222 FROM
223     events
224 WHERE
225     event_type = 'engagement'
226 GROUP BY device , week_no
227 ORDER BY week_no;
228
```

Below the query, the 'Result Grid' shows the output of the query. The grid has three columns: 'device', 'week_no', and 'no_of_users'. The data is as follows:

device	week_no	no_of_users
acer aspire desktop	17	9
acer aspire notebook	17	20
amazon fire phone	17	4
asus chromebook	17	21
dell inspiron desktop	17	18
dell inspiron notebook	17	46
hp pavilion desktop	17	14
htc one	17	16
ipad air	17	27
ipad mini	17	19
iphone 4s	17	21

Here, we are checking for the activeness of users on a weekly basis per device.

We see that the user engagement varies across devices per week. Some devices have a higher engagement as compared to others. We can check for factors affecting the engagement and work more on device compatibility, optimizing user experience. Also, we need to monitor trends for such devices with lower user engagement and devise strategies to overcome the challenges that users are facing with the devices for more user engagement.

Task 5: Email Engagement Analysis

```
235 • select count(action) as action_count, action from email_events group by action;
236 • SELECT
237   (SUM(CASE
238     WHEN email_category = 'email_opened' THEN 1 ELSE 0 END) / SUM(CASE
239     WHEN email_category = 'email_sent' THEN 1 ELSE 0 END)) * 100 AS email_open_rate,
240   (SUM(CASE
241     WHEN email_category = 'email_clicked' THEN 1 ELSE 0 END) / SUM(CASE
242     WHEN email_category = 'email_sent' THEN 1 ELSE 0
243     END)) * 100 AS email_clicked_rate
244 FROM
245   (SELECT *,
246     CASE
247       WHEN action IN ('sent_weekly_digest', 'sent_reengagement_email') THEN ('email_sent')
248       WHEN action IN ('email_open') THEN ('email_opened')
249       WHEN action IN ('email_clickthrough') THEN ('email_clicked')
250     END AS email_category
251   FROM
252     email_events) AS a;
```

Table: job_data

Columns:

job_id	int
actor_id	int
event	text
language	text
time_spent	int
org	text

Result Grid

email_open_rate	email_clicked_rate
33.5834	14.7899

Here, we are analyzing how users are engaging with the email service and calculating the email engagement metrics.

In the table of **email_events**, we have **4** actions for emails which are: **sent_weekly_digest**, **email_open**, **email_clickthrough**, **sent_reengagement_email**.

We can use them to calculate the engagement metrics which are **email_open_rate** and **email_clicked_rate**.

After calculating, we see **email_open_rate = 33.5834** and **email_clicked_rate = 14.7899**.

Now, we can check these metrics with the industry standard metrics and if found less, we can use more refined words in the email, change our subject line for the emails, targeting audience etc.

Atlast, regularly check and optimize our email content and campaigns to reach the industry standards.

Result:

While completing the project, I have gained knowledge on operational analytics, understanding its principle and methodologies and how important it is for a business or company to find a trend on the working and helping the business grow. Also, this project helped me on learning how to deal with large datasets and use aggregations, joins, subqueries to refine results to ensure accuracy, which gives us a correct and reliable analysis. In the project, I have worked with people from various departments and collaborated with them which enhanced my communication skills enabling me to share my findings and recommendations to the respective departments for data-driven decision making to improve and optimize overall user experience, user engagement, process engagement thus, contributing in the overall success of the project.