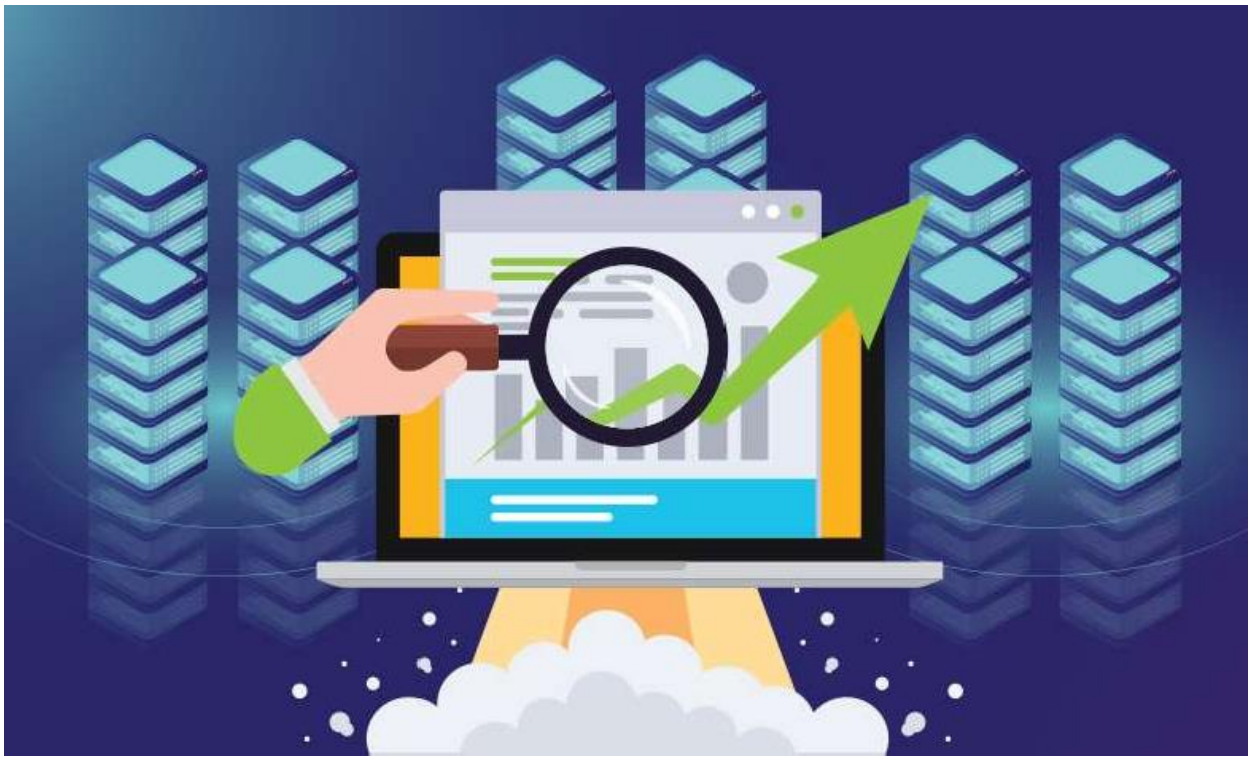
Operation Analytics and Investigating Metric Spike

Advanced SQL



Introduction

This project focuses on analyzing the data which is provided by company. My task is to derive insights and answer the questions asked by different departments. So that these insights are then used by **ops team**, **support team**, **marketing team**, etc to predict the overall growth or decline of a company's fortune. It means better automation, better understanding between cross-functional teams, and more effective workflows.

In case study 1 there is **job_data** table while in case study 2 there are **users**, **events** and **email_events** tables.

In case study 1 the insights are found based on following questions:

1. **Number of jobs reviewed:** Amount of jobs reviewed over time.
2. **Throughput:** Calculate 7 day rolling average of throughput? For throughput, do you prefer daily metric or 7-day rolling and why?
3. **Percentage share of each language:** Share of each language for different contents.
4. **Duplicate rows:** Rows that have the same value present in them.

In case study 2 the insights are found based on following questions:

1. **User Engagement:** To measure the activeness of a user. Measuring if the user finds quality in a product/service.
2. **User Growth:** Amount of users growing over time for a product.
3. **Weekly Retention:** Users getting retained weekly after signing-up for a product.
4. **Weekly Engagement:** To measure the activeness of a user. Measuring if the user finds quality in a product/service weekly.
5. **Email Engagement:** Users engaging with the email service.

What is Operation Analytics?

Operational analytics focuses on measuring the existing and real-time operations of the company so that the company can monitor their day-to-day operations basis which they can take the necessary actions to improve customer satisfaction and bottom line.

Operation Analytics is the analysis done for the complete end to end operations of a company. With the help of this, the company then finds the areas on which it must improve upon. Being one of the most important parts of a company, this kind of analysis is further used to predict the overall growth or decline of a company's fortune.

Here are few examples:

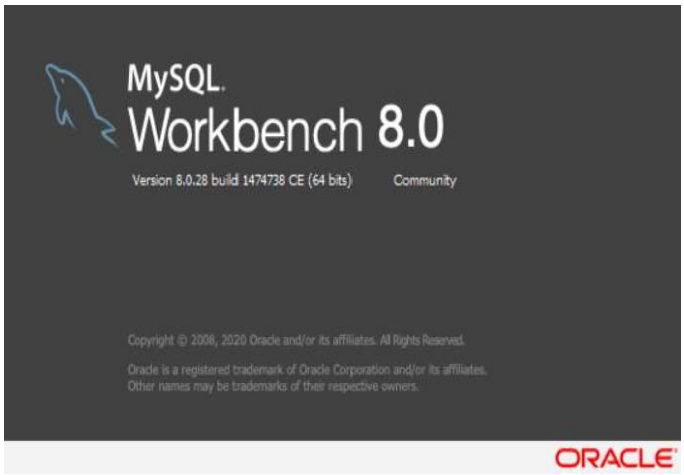
- **Ops:** Developers can use real-time data to look at how customers are using their products and make changes on the fly.
- **Marketing:** Businesses can optimize user engagement in real-time by using operational analytics to make personalized recommendations.

Project Approach

This project is developed using SQL Workbench. First I need to create database by using dataset file which was provided by the company. Next step load the data into SQL Workben h then performed analysis and find the information that will the help the **ops team**, **support team**, **marketing team**, etc to understand questions like - Why is there a dip in daily engagement? Why have sales taken a dip? Etc. Questions like these must be answered daily and for that it's very important to investigate metric spike.

Tech-Stack Used

MySQL Workbench is a visual editor that unifies data modeling, SQL development, and database administration in one interface. It allows you to visually design, generate, and manage databases.



MySQL Workbench is widely used to handle structured data. It is an open-source Relational Database Management System (RDBMS) developed by Oracle Corporation, Sun Microsystems that uses Structured Query language (SQL) to interact with databases.

MySQL Workbench offers database migration options, making it easier to move data to and from the Microsoft SQL Server, Microsoft Access and other RDBMS tables.

Key Functionalities of MySQL Workbench

- **Visual SQL Editor:** MySQL Workbench is equipped with a visual SQL editor where developers can build, edit, and run queries. What's great about this is that it allows you to preview your changes before applying them.
- **Database Administration:** Aside from providing you with SQL editing tools, MySQL Workbench also comes with a database administration suite. This makes it easy for you to audit your databases, configure servers, and view logs.
- **Performance Monitoring:** MySQL Workbench gives users a dashboard where they can view the status of their queries, client timing, network latency, and index usage. This allows for simpler identification of possible ways to optimize SQL performance.

Insights

Case Study 1 (Job Data)

1. Calculate the number of jobs reviewed per hour per day for November 2020?

```
SELECT ds AS Dates, ROUND((COUNT(job_id)/SUM(time_spent))*3600) AS "Jobs Reviewed per Hour per Day"
FROM job_data
WHERE ds BETWEEN '2020-11-01' AND '2020-11-30'
GROUP BY ds;
```

On date 2020-11-28 there is maximum number of jobs reviewed that is **218**.

Dates	Jobs Reviewed per Hour per Day
2020-11-30	180
2020-11-29	180
2020-11-28	218
2020-11-27	35
2020-11-26	64
2020-11-25	80

2. Calculate 7 day rolling average of throughput? For throughput, do you prefer daily metric or 7-day rolling and why?

```
SELECT ROUND(COUNT(event)/SUM(time_spent), 2) AS "Weekly Throughput"
FROM job_data;
```

The weekly throughput is **0.03**.

Weekly Throughput
0.03

```
SELECT ds AS Dates, ROUND(COUNT(event)/SUM(time_spent), 2) AS "Daily Throughput"
FROM job_data
GROUP BY ds
ORDER BY ds;
```

On date 2020-11-28 the throughput is highest **0.06**.

Dates	Daily 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

Metrics will always go up and down on a weekly and daily basis. You'll get numbers faster every day or minute if you want. As a result, rolling metrics are superb at showing if your metrics are trending up or down on a daily level.

3. Calculate the percentage share of each language in the last 30 days?

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

Persian language is highest with **37.5%** total.

Languages	Percentage
English	12.50
Arabic	12.50
Persian	37.50
Hindi	12.50
French	12.50
Italian	12.50

4. Let's say you see some duplicate rows in the data. How will you display duplicates from the table?

```
SELECT actor_id, COUNT(*) AS Duplicates
FROM job_data
GROUP BY actor_id
HAVING COUNT(*) > 1;
```

Actor ID **1003** has duplicate rows.

actor_id	Duplicates
1003	2

Case Study 2 (Investigating Metric Spike)

1. Calculate the weekly user engagement?

```
SELECT EXTRACT(WEEK FROM occurred_at) AS "Week Numbers", COUNT(DISTINCT
user_id) AS "Weekly Active Users"
FROM events
WHERE event_type = 'engagement'
GROUP BY 1;
```

Week Numbers	Weekly Active 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
29	1376
30	1467
31	1299
32	1225
33	1225
34	1204
35	104

2. Calculate the user growth for product?

```
SELECT Months, Users, ROUND(((Users/LAG(Users, 1) OVER (ORDER BY Months) -
1)*100), 2) AS "Growth in %"
FROM
(
SELECT EXTRACT(MONTH FROM created_at) AS Months, COUNT(activated_at) AS Users
FROM users
WHERE activated_at NOT IN("")
GROUP BY 1
ORDER BY 1
) sub;
```

Months	Users	Growth in %
1	712	NULL
2	685	-3.79
3	765	11.68
4	907	18.56
5	993	9.48
6	1086	9.37
7	1281	17.96
8	1347	5.15
9	330	-75.50
10	390	18.18
11	399	2.31
12	486	21.80

3. Calculate the weekly retention of users-sign up cohort?

```
SELECT first AS "Week Numbers",
SUM(CASE WHEN week_number = 0 THEN 1 ELSE 0 END) AS "Week 0",
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 week_number = 16 THEN 1 ELSE 0 END) AS "Week 16",
SUM(CASE WHEN week_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 m.user_id, m.login_week, n.first, m.login_week - first AS week_number
FROM
(SELECT user_id, EXTRACT(WEEK FROM occurred_at) AS login_week FROM events
GROUP BY 1, 2) m,
(SELECT user_id, MIN(EXTRACT(WEEK FROM occurred_at)) AS first FROM events
GROUP BY 1) n
WHERE m.user_id = n.user_id
) sub
GROUP BY first
ORDER BY first;
```

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	Week 13	Week 14	Week 15	Week 16	Week 17	Week 18
17	740	472	324	251	205	187	167	146	145	145	136	131	132	143	116	91	82	77	5
18	788	362	261	203	168	147	144	127	113	122	106	118	127	110	97	85	67	4	0
19	601	284	173	153	114	95	91	81	95	82	68	65	63	42	51	49	2	0	0
20	555	223	165	121	91	72	63	67	63	65	67	41	40	33	40	0	0	0	0
21	495	187	131	91	74	63	75	72	58	48	45	39	35	28	2	0	0	0	0
22	521	224	150	107	87	73	63	60	55	48	41	39	31	1	0	0	0	0	0
23	542	219	138	101	90	79	69	61	54	47	35	30	0	0	0	0	0	0	0
24	535	205	143	102	81	63	65	61	38	39	29	0	0	0	0	0	0	0	0
25	500	218	139	101	75	63	50	46	38	35	2	0	0	0	0	0	0	0	0
26	495	181	114	83	73	55	47	43	29	0	0	0	0	0	0	0	0	0	0
27	493	199	121	106	68	53	40	36	1	0	0	0	0	0	0	0	0	0	0
28	486	194	114	69	46	30	28	3	0	0	0	0	0	0	0	0	0	0	0
29	501	186	102	65	47	40	1	0	0	0	0	0	0	0	0	0	0	0	0
30	533	202	121	78	53	3	0	0	0	0	0	0	0	0	0	0	0	0	0
31	430	145	76	57	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0
32	496	188	94	8	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
33	499	202	9	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
34	518	44	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
35	32	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

4. Calculate the weekly engagement per device?

```
SELECT EXTRACT(WEEK FROM occurred_at) AS "Week Numbers",
COUNT(DISTINCT CASE WHEN device IN('dell inspiron notebook') THEN user_id ELSE NULL END) AS "Dell Inspiron Notebook",
COUNT(DISTINCT CASE WHEN device IN('iphone 5') THEN user_id ELSE NULL END) AS "iPhone 5",
COUNT(DISTINCT CASE WHEN device IN('iphone 4s') THEN user_id ELSE NULL END) AS "iPhone 4S",
COUNT(DISTINCT CASE WHEN device IN('windows surface') THEN user_id ELSE NULL END) AS "Windows Surface",
COUNT(DISTINCT CASE WHEN device IN('macbook air') THEN user_id ELSE NULL END) AS "Macbook Air",
COUNT(DISTINCT CASE WHEN device IN('iphone 5s') THEN user_id ELSE NULL END) AS "iPhone 5S",
COUNT(DISTINCT CASE WHEN device IN('macbook pro') THEN user_id ELSE NULL END) AS "Macbook Pro",
COUNT(DISTINCT CASE WHEN device IN('kindle fire') THEN user_id ELSE NULL END) AS "Kindle Fire",
COUNT(DISTINCT CASE WHEN device IN('ipad mini') THEN user_id ELSE NULL END) AS "iPad Mini",
COUNT(DISTINCT CASE WHEN device IN('nexus 7') THEN user_id ELSE NULL END) AS "Nexus 7",
COUNT(DISTINCT CASE WHEN device IN('nexus 5') THEN user_id ELSE NULL END) AS "Nexus 5",
COUNT(DISTINCT CASE WHEN device IN('samsung galaxy s4') THEN user_id ELSE NULL END) AS "Samsung Galaxy S4",
COUNT(DISTINCT CASE WHEN device IN('lenovo thinkpad') THEN user_id ELSE NULL END) AS "Lenovo Thinkpad",
COUNT(DISTINCT CASE WHEN device IN('samsumg galaxy tablet') THEN user_id ELSE NULL END) AS "Samsumg Galaxy Tablet",
COUNT(DISTINCT CASE WHEN device IN('acer aspire notebook') THEN user_id ELSE NULL END) AS "Acer Aspire Notebook",
COUNT(DISTINCT CASE WHEN device IN('asus chromebook') THEN user_id ELSE NULL END) AS "Asus Chromebook",
COUNT(DISTINCT CASE WHEN device IN('htc one') THEN user_id ELSE NULL END) AS "HTC One",
COUNT(DISTINCT CASE WHEN device IN('nokia lumia 635') THEN user_id ELSE NULL END) AS "Nokia Lumia 635",
COUNT(DISTINCT CASE WHEN device IN('samsung galaxy note') THEN user_id ELSE NULL END) AS "Samsung Galaxy Note",
COUNT(DISTINCT CASE WHEN device IN('acer aspire desktop') THEN user_id ELSE NULL END) AS "Acer Aspire Desktop",
COUNT(DISTINCT CASE WHEN device IN('mac mini') THEN user_id ELSE NULL END) AS "Mac Mini",
COUNT(DISTINCT CASE WHEN device IN('hp pavilion desktop') THEN user_id ELSE NULL END) AS "HP Pavilion Desktop",
COUNT(DISTINCT CASE WHEN device IN('dell inspiron desktop') THEN user_id ELSE NULL END) AS "Dell Inspiron Desktop",
COUNT(DISTINCT CASE WHEN device IN('ipad air') THEN user_id ELSE NULL END) AS "iPad Air",
```

```
COUNT(DISTINCT CASE WHEN device IN('amazon fire phone') THEN user_id ELSE NULL END) AS "Amazon Fire Phone",
COUNT(DISTINCT CASE WHEN device IN('nexus 10') THEN user_id ELSE NULL END) AS "Nexus 10"
FROM events
WHERE event_type = 'engagement'
GROUP BY 1
ORDER BY 1;
```

Week Numbers	Dell Inspiron Notebook	iPhone 5	iPhone 4S	Windows Surface	Macbook Air	iPhone 5S	Macbook Pro	Kindle Fire	iPad Mini	Nexus 7	Nexus 5	Samsung Galaxy S4	Lenovo Thinkpad	Samsumg Galaxy Tablet
17	46	65	21	10	54	42	143	6	19	18	40	52	86	8
18	77	113	46	10	121	73	252	27	30	30	73	82	153	11
19	83	115	44	16	112	79	266	21	36	41	87	91	178	6
20	84	125	55	21	119	79	256	23	32	32	103	93	173	9
21	80	137	45	17	110	74	247	30	23	29	91	84	167	6
22	92	125	45	15	145	71	251	21	34	45	96	105	176	10
23	103	152	53	14	124	79	266	25	33	36	88	99	176	14
24	99	142	53	22	152	79	255	25	39	49	87	101	165	11
25	105	137	40	22	121	78	275	24	30	51	89	99	197	12
26	89	152	50	21	134	94	269	26	43	46	87	112	192	12
27	89	163	67	33	142	83	302	25	35	40	84	116	202	15
28	103	151	61	33	148	93	295	31	35	39	85	122	220	9
29	113	144	60	28	148	90	295	37	34	45	77	123	209	13
30	127	152	65	19	159	103	322	25	35	62	84	103	206	9
31	113	135	56	19	147	71	321	14	27	38	69	100	207	8
32	104	119	34	10	125	67	307	12	30	25	67	82	179	6
33	110	110	35	15	133	65	312	14	28	30	70	80	191	12
34	105	101	50	18	136	70	292	13	25	33	70	90	193	14
35	9	2	6	3	10	3	17	3	2	2	4	6	16	0

Week Numbers	Acer Aspire Notebook	Asus Chromebook	HTC One	Nokia Lumia 635	Samsung Galaxy Note	Acer Aspire Desktop	Mac Mini	HP Pavilion Desktop	Dell Inspiron Desktop	iPad Air	Amazon Fire Phone	Nexus 10
17	20	21	16	17	7	9	6	14	18	27	4	16
18	33	42	19	33	15	26	13	37	58	52	9	30
19	41	27	30	23	11	23	18	40	36	55	12	25
20	40	41	29	22	18	23	26	30	52	59	11	22
21	47	38	21	25	20	29	18	44	41	51	5	25
22	41	52	24	25	19	25	25	38	52	58	5	27
23	43	49	20	31	14	22	18	54	53	41	16	45
24	40	43	20	35	20	24	29	56	59	57	11	38
25	47	38	21	37	14	28	21	52	52	57	13	29
26	35	49	23	42	9	29	11	46	60	56	13	29
27	49	52	27	31	15	29	15	56	53	55	10	37
28	49	50	26	35	10	30	28	56	56	54	6	26
29	53	49	31	43	16	28	31	58	54	52	12	25
30	60	56	31	34	15	33	23	42	54	70	12	36
31	55	56	13	28	14	31	24	51	44	55	14	24
32	55	62	18	28	12	35	20	51	57	48	12	30
33	46	49	19	27	13	39	32	38	37	40	14	23
34	63	47	25	17	13	30	30	36	49	39	11	25
35	3	6	2	2	1	1	2	1	1	0	0	2

5. Calculate the email engagement metrics?

```
SELECT Week,
ROUND((weekly_digest/total*100),2) AS "Weekly Digest Rate",
ROUND((email_opens/total*100),2) AS "Email Open Rate",
ROUND((email_clickthroughs/total*100),2) AS "Email Clickthrough Rate",
ROUND((reengagement_emails/total*100),2) AS "Reengagement Email Rate"
FROM
(
SELECT EXTRACT(WEEK FROM occurred_at) AS Week,
COUNT(CASE WHEN action = 'sent_weekly_digest' THEN user_id ELSE NULL END) AS
weekly_digest,
COUNT(CASE WHEN action = 'email_open' THEN user_id ELSE NULL END) AS
email_opens,
COUNT(CASE WHEN action = 'email_clickthrough' THEN user_id ELSE NULL END) AS
email_clickthroughs,
COUNT(CASE WHEN action = 'sent_reengagement_email' THEN user_id ELSE NULL END)
AS reengagement_emails,
COUNT(user_id) AS total
FROM email_events
GROUP BY 1
) sub
GROUP BY 1
ORDER BY 1;
```

Week	Weekly Digest Rate	Email Open Rate	Email Clickthrough Rate	Reengagement Email Rate
17	62.32	21.28	11.39	5.01
18	63.45	22.24	10.49	3.83
19	62.16	22.67	11.13	4.04
20	61.62	22.64	11.43	4.31
21	63.52	22.82	9.97	3.69
22	63.59	21.56	10.66	4.19
23	62.39	22.34	11.18	4.09
24	61.61	22.92	10.99	4.48
25	63.77	21.79	10.54	3.90
26	62.99	22.22	10.61	4.18
27	62.24	22.49	11.37	3.90
28	62.92	22.48	10.77	3.83
29	63.98	21.71	10.51	3.79
30	62.29	23.24	10.59	3.88
31	65.27	23.25	7.66	3.82
32	66.59	22.85	7.14	3.42
33	64.73	23.10	7.91	4.26
34	64.33	23.91	7.67	4.08
35	0.00	32.28	29.92	37.80

Result

How this project helped me: This project helps me to understand the importance of operation analytics. Through this project I am able to understand how the companies use metric spike as a secret weapon. With an informed and proactive approach, they can leverage insights to make data-backed decisions that optimize their strategy and boost ROI.

Challenges that I faced in this project: The challenge here is that the data in **case study 2** is very huge, as the huge amount of data SQL Workbech is very slow to import it. To tackle this situation I have to use **LOAD DATA** statements. Now, there is another problem arises in the column **user_type** in **events** table that has datatype **int** which is stopping the process of importing. First I need to change its datatype to **text** then restart the process of loading the data into **events** table.

Conclusion: Operational Analytics tackles the problem by synchronizing real-time data. Operational Analytics has the capability to aggregate data from multiple data sources into a cumulative, organized, actionable solution capable of delivering analytical models in real-time to create individual customer profiles and a holistic view of operations for a company. This guarantees that your operational routines and systems are used efficiently. Whenever utilized correctly, operational analyt cs can achieve a significant positive effect on our general public and world everywhere and increment the general efficiency of specific areas.