Project-3
Operation Analytics and Investigating Metric Spike

Project Description

- This project involves Operation Analysis -analysing company's end-to-end operations which helps in identifying areas for improvement within the company by deriving valuable insights from the collected data.
- It also includes Investigating Metric Spikes which involves understanding and explaining sudden changes in key metrics, such as dip in daily, weekly, monthly etc.; user engagement or growth of a company's product.
  This kind of analysis is further used to predict the overall growth or decline of a company's fortune.

Approach

- **Create the Database and Tables:** Started by creating a database for the project and then create the necessary tables using the provided table structures and links.

- **Perform Analysis:** Utilize SQL to perform the analysis and answer the questions mentioned in the case studies.

Tech-stack used

- *My SQL Workbench (Version- 8.0)* - used for working, analysing and reporting insights

- *MS-WORD* – used for creating this detailed analysis report

Insights

Case-1: JOB DATA Analysis

Creating job_data table:

```sql
create table job_data(
ds varchar(60),
job_id int,
actor_id int,
event varchar(20),
language varchar(20),
time_spent int,
org varchar(10)
);


show variables like 'secure_file_priv';

load data infile"C:/ProgramData/MySQL/MySQL Server 8.0/Uploads/job_data.csv"
into table job_data
fields terminated by ','
enclosed by '"'
lines terminated by'\n'
ignore 1 rows;

select * from job_data;
```

ds	job_id	actor_id	event	language	time_spent	org
11/30/2020	21	1001	skip	English	15	A
11/30/2020	22	1006	transfer	Arabic	25	B
11/29/2020	23	1003	decision	Persian	20	C
11/28/2020	23	1005	transfer	Persian	22	D
11/28/2020	25	1002	decision	Hindi	11	
11/27/2020	11	1007	decision	French	104	D
11/26/2020	23	1004	skip	Persian	56	A
11/25/2020	20	1003	transfer	Italian	45	C

22

A. Calculate the number of jobs reviewed per hour for each day in November 2020.

```
##  calculate the number of jobs reviewed per hour for each day in November 2020.
```

```
SELECT DS,ROUND(JID/TIME_SPENT_IN_HRS)  AS OUTPUT
FROM(
SELECT COUNT(JOB_ID) AS JID, SUM(TIME_SPENT)/3600 AS TIME_SPENT_IN_HRS, DS
FROM  JOB_DATA
GROUP BY DS
) AS aa ;
```

DS	OUTPUT
11/30/2020	180
11/29/2020	179
11/28/2020	217
11/27/2020	35
11/26/2020	64
11/25/2020	80

Max. No. of jobs reviewed is on 11/28/2020 i.e.; 217

B. Calculate the 7-day rolling average of throughput. Additionally, explain whether you prefer using the daily metric or the 7-day rolling average for throughput, and why?

```
##Calculate the 7-day rolling average of throughput (number of events per second).

SELECT COUNT(EVENT)/SUM(TIME_SPENT) AS 7day_THROUGHPUT
FROM JOB_DATA;

#
SELECT DS,COUNT(EVENT)/SUM(TIME_SPENT) AS DAILY_THROUGHPUT
FROM JOB_DATA
GROUP by DS
;
```

DS	DAILY_THROUGHPUT
11/25/2020	0.0222
11/26/2020	0.0179
11/27/2020	0.0096
11/28/2020	0.0606
11/29/2020	0.0500
11/30/2020	0.0500

Result 52

7day_THROUGHPUT
0.0268

Weekly Throughput is 0.0268

&

Daily throughput is highest 0.0606 on 28/11/2020

Using Weekly Throughput is more helpful in understanding the trends over time as it provides a wider perspective as compared to daily metric.

C. Calculate the percentage share of each language over the last 30 days.

```sql
### calculate the percentage share of each language over the last 30 days
select * from job_data;

SELECT  LANGUAGE,COUNTR,COUNTR*100/SUM(COUNTR) OVER() AS PERCENTAGE
FROM
(SELECT COUNT(LANGUAGE)AS COUNTR ,LANGUAGE
FROM JOB_DATA
GROUP BY  LANGUAGE
) AS SUB
ORDER BY PERCENTAGE DESC;
```

LANGUAGE	COUNTR	PERCENTAGE
Persian	3	37.5000
English	1	12.5000
Arabic	1	12.5000
Hindi	1	12.5000
French	1	12.5000
Italian	1	12.5000

Persian Language has the highest share.

D. Display duplicate rows from the job_data table.

```
62    ### to display duplicate rows from the job_data table
63
64    SELECT
65        JOB_ID,
66        CASE
67            WHEN CNTR = 1 THEN ''
68            ELSE 'DUPLICATE '
69        END AS DUPLICATE_ROWS
70    FROM
71        (SELECT
72            JOB_ID, COUNT(JOB_ID) AS CNTR
73        FROM
74            JOB_DATA
75        GROUP BY JOB_ID) SUB;
76
```

JOB_ID	DUPLICATE_ROWS
21	
22	
23	DUPLICATE
25	
11	
20	

Case-2: Investigating Metric Spikes

Creating users, events, email_events table:

```
102  create table users (
103  user_id int,
104  created_at varchar(60),
105  company_id int,
106   language varchar(50),
107  activated_at varchar(80),
108  state varchar(20));
109
110  show variables like 'secure_file_priv';
111
112  LOAD DATA INFILE "C:/ProgramData/MySQL/MySQL Server 8.0/Uploads/table-1_users.csv"
113  INTO table users
114  FIELDS TERMINATED BY ','
115  ENCLOSED BY '"'
116  LINES TERMINATED BY '\n'
117  IGNORE 1 ROWS;
118
119  select * from users;
120  ALTER TABLE USERS
121      ADD COLUMN TEMP_CREATED_AT DATETIME;
122  UPDATE USERS
123  SET
124      TEMP_CREATED_AT = STR_TO_DATE(created_at,'%d-%m-%Y %H:%i');
125  Alter table users
126  DROP COLUMN CREATED_AT;
127  ALTER table USERS
128  CHANGE column TEMP_CREATED_AT  CREATED_AT DATETIME;
```

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

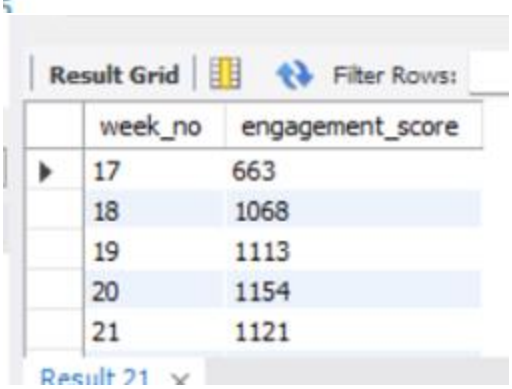
user_id	company_id	language	activated_at	state	CREATED_AT
0	5737	english	01-01-2013 21:01	active	2013-01-01 20:59:00
3	2800	german	01-01-2013 18:42	active	2013-01-01 18:40:00
4	5110	indian	01-01-2013 14:39	active	2013-01-01 14:37:00
6	11699	english	01-01-2013 18:38	active	2013-01-01 18:37:00
7	4765	french	01-01-2013 16:20	active	2013-01-01 16:19:00
8	2698	french	01-01-2013 04:40	active	2013-01-01 04:38:00
11	3745	english	01-01-2013 08:09	active	2013-01-01 08:07:00
13	4025	english	02-01-2013 12:29	active	2013-01-02 12:27:00
15	4259	english	02-01-2013 15:41	active	2013-01-02 15:39:00
17	5025	japanese	02-01-2013 10:57	active	2013-01-02 10:56:00
19	326	english	02-01-2013 09:55	active	2013-01-02 09:54:00
20	7	italian	02-01-2013 09:43	active	2013-01-02 09:41:00
21	2606	english	02-01-2013 09:30	active	2013-01-02 09:29:00
22	545	german	02-01-2013 17:38	active	2013-01-02 17:36:00
27	6	japanese	03-01-2013 16:15	active	2013-01-03 16:14:00
30	4148	english	03-01-2013 08:29	active	2013-01-03 08:28:00
31	39	arabic	03-01-2013 15:46	active	2013-01-03 15:45:00
33	10768	english	03-01-2013 12:18	active	2013-01-03 12:16:00
35	1891	english	03-01-2013 16:07	active	2013-01-03 16:06:00
36	2	english	03-01-2013 11:53	active	2013-01-03 11:51:00
47	1	indian	04-01-2013 10:41	active	2013-01-04 10:39:00
49	8727	spanish	05-01-2013 14:34	active	2013-01-05 14:33:00

Similarly, other two tables i.e.; events and email_events table are created.

A. Calculate the weekly user engagement.

```
######### calculate the weekly user engagement.
select * from events;

SELECT
    EXTRACT(WEEK FROM occurred_at) AS week_no,
    COUNT(distinct user_id) AS engagement_score
FROM
    events
WHERE
    event_type = 'engagement'
GROUP BY week_no;
```



Result Grid | Filter Rows:

week_no	engagement_score
17	663
18	1068
19	1113
20	1154
21	1121

Result 21

Week 35 has lowest engagement score 104

Week 30 has highest engagement score 1467

B. Calculate the user growth for the product.

```
##############calculate the user growth for the product.

use project3;
select * from users;

SELECT
    year, week, active_users,SUM(active_users)
    over(order by year,week) as growth
FROM
    (SELECT
        EXTRACT(YEAR FROM created_at) AS year,
            EXTRACT(WEEK FROM created_at) AS week,
            COUNT(user_id) AS active_users
    FROM
        users
    GROUP BY week , year) sub
;
```

year	week	active_users	growth
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

Week 33 saw the greatest no of users actively engaging

&Week 35 saw lowest

C. Calculate the weekly retention of users based on their sign-up cohort

```sql
select * from events;

SELECT
    first_sign AS week_nos,
    SUM(CASE
        WHEN week_num = 0 THEN 1
        ELSE 0
    END) AS week_0,
    SUM(CASE
        WHEN week_num = 1 THEN 1
        ELSE 0
    END) AS week_1,
    SUM(CASE
        WHEN week_num = 2 THEN 1
        ELSE 0
    END) AS week_2,
    SUM(CASE
        WHEN week_num = 3 THEN 1
        ELSE 0
    END) AS week_3,
    SUM(CASE
        WHEN week_num = 4 THEN 1
        ELSE 0
    END) AS week_4,
    SUM(CASE
        WHEN week_num = 5 THEN 1
        ELSE 0
    END) AS week_5,
    SUM(CASE
        WHEN week_num = 6 THEN 1
        ELSE 0
    END) AS week_6,
    SUM(CASE
        WHEN week_num = 7 THEN 1
        ELSE 0
    END) AS week_7,
    SUM(CASE
        WHEN week_num = 8 THEN 1
        ELSE 0
    END) AS week_8,
    SUM(CASE
        WHEN week_num = 9 THEN 1
        ELSE 0
    END) AS week_9,
```

```
    SUM(CASE
        WHEN week_num = 10 THEN 1
        ELSE 0
    END) AS week_10,
    SUM(CASE
        WHEN week_num = 11 THEN 1
        ELSE 0
    END) AS week_11,
    SUM(CASE
        WHEN week_num = 12 THEN 1
        ELSE 0
    END) AS week_12,
    SUM(CASE
        WHEN week_num = 13 THEN 1
        ELSE 0
    END) AS week_13,
    SUM(CASE
        WHEN week_num = 14 THEN 1
        ELSE 0
    END) AS week_14,
    SUM(CASE
276         SUM(CASE
277             WHEN week_num = 15 THEN 1
278             ELSE 0
279         END) AS week_15,
280         SUM(CASE
281             WHEN week_num = 16 THEN 1
282             ELSE 0
283         END) AS week_16,
284         SUM(CASE
285             WHEN week_num = 17 THEN 1
286             ELSE 0
287         END) AS week_17,
288         SUM(CASE
289             WHEN week_num = 18 THEN 1
290             ELSE 0
291         END) AS week_18
292       FROM
293         (SELECT
294             a.user_id,
295                 a.week,
296                 b.first_sign,
297                 a.week - b.first_sign AS week_num
298         FROM
299             (SELECT
300             user_id, EXTRACT(WEEK FROM occurred_at) AS week
301         FROM
302             events
303         GROUP BY user_id , week) a,
304         (SELECT
305             user_id, MIN(EXTRACT(WEEK FROM occurred_at)) AS first_sign
306         FROM
307             events
308         GROUP BY user_id) b
309         where a.user_id=b.user_id
310         )
311         sub
312       GROUP BY first_sign
313       ORDER BY first_sign;
```

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

week_nos	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	663	472	324	251	205	187	167	146	145	145	136	131	132	143	116	91	82	77	5
18	596	362	261	203	168	147	144	127	113	122	106	118	127	110	97	85	67	4	0
19	427	284	173	153	114	95	91	81	95	82	68	65	63	42	51	49	2	0	0
20	358	223	165	121	91	72	63	67	63	65	67	41	40	33	40	0	0	0	0
21	317	187	131	91	74	63	75	72	58	48	45	39	35	28	2	0	0	0	0
22	326	224	150	107	87	73	63	60	55	48	41	39	31	1	0	0	0	0	0
23	328	219	138	101	90	79	69	61	54	47	35	30	0	0	0	0	0	0	0
24	339	205	143	102	81	63	65	61	38	39	29	0	0	0	0	0	0	0	0
25	305	218	139	101	75	63	50	46	38	35	2	0	0	0	0	0	0	0	0
26	288	181	114	83	73	55	47	43	29	0	0	0	0	0	0	0	0	0	0
27	292	199	121	106	68	53	40	36	1	0	0	0	0	0	0	0	0	0	0
28	274	194	114	69	46	30	28	3	0	0	0	0	0	0	0	0	0	0	0
29	270	186	102	65	47	40	1	0	0	0	0	0	0	0	0	0	0	0	0
30	294	202	121	78	53	3	0	0	0	0	0	0	0	0	0	0	0	0	0
31	215	145	76	57	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0
32	267	188	94	8	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
33	286	202	9	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
34	279	44	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
35	18	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

D. Calculate the weekly engagement per device

```
359    ###############     Write an SQL query to calculate the weekly engagement per device.
360 •  select * from events;
361
362 •  select device, extract(week from occurred_at) as week ,
363    count(distinct user_id) as count from events
364    where event_type = 'engagement'
365
366    group by device, week
367    order by device;
368
369
```

device	week	count
acer aspire desktop	17	9
acer aspire desktop	18	26
acer aspire desktop	19	23
acer aspire desktop	20	23
acer aspire desktop	21	29
acer aspire desktop	22	25
acer aspire desktop	23	22
acer aspire desktop	24	24
acer aspire desktop	25	28
acer aspire desktop	26	29
acer aspire desktop	27	29
acer aspire desktop	28	30
acer aspire desktop	29	28
acer aspire desktop	30	33
acer aspire desktop	31	31
acer aspire desktop	32	35
acer aspire desktop	33	39
acer aspire desktop	34	30
acer aspire desktop	35	1
acer aspire notebook	17	20
acer aspire notebook	18	33
acer aspire notebook	19	41
acer aspire notebook	20	40
acer aspire notebook	21	47
acer aspire notebook	22	41
acer aspire notebook	23	43
acer aspire notebook	24	40
acer aspire notebook	25	47

device	week	count
acer aspire notebook	25	47
acer aspire notebook	26	35
acer aspire notebook	27	49
acer aspire notebook	28	49
acer aspire notebook	29	53
acer aspire notebook	30	60
acer aspire notebook	31	55
acer aspire notebook	32	55
acer aspire notebook	33	46
acer aspire notebook	34	63
acer aspire notebook	35	3
amazon fire phone	17	4
amazon fire phone	18	9
amazon fire phone	19	12
amazon fire phone	20	11
amazon fire phone	21	5
amazon fire phone	22	5
amazon fire phone	23	16
amazon fire phone	24	11
amazon fire phone	25	13
amazon fire phone	26	13
amazon fire phone	27	10
amazon fire phone	28	6
amazon fire phone	29	12
amazon fire phone	30	12
amazon fire phone	31	14
amazon fire phone	32	12
amazon fire phone	33	14

device	week	count
amazon fire phone	32	12
amazon fire phone	33	14
amazon fire phone	34	11
asus chromebook	17	21
asus chromebook	18	42
asus chromebook	19	27
asus chromebook	20	41
asus chromebook	21	38
asus chromebook	22	52
asus chromebook	23	49
asus chromebook	24	43
asus chromebook	25	38
asus chromebook	26	49
asus chromebook	27	52
asus chromebook	28	50
asus chromebook	29	49
asus chromebook	30	56
asus chromebook	31	56
asus chromebook	32	62
asus chromebook	33	49
asus chromebook	34	47
asus chromebook	35	6
dell inspiron desktop	17	18
dell inspiron desktop	18	58
dell inspiron desktop	19	36
dell inspiron desktop	20	52
dell inspiron desktop	21	41
dell inspiron desktop	22	52

.....and so on.

Highest engagement was with device was MacBook pro for week 30 with 322 users.

E. Calculate the email engagement metrics.

```
378      ########### Write an SQL query to calculate the email engagement metrics.
379
380 •   select * from email_events;
381
382 •   select  action, count(action)
383     from email_events
384      group by action
385     ;
386
387 •   select
388     sum(case when email_category= 'email_sent' then 1 else 0 end) as total_email_sent,
389     sum( case when email_category='email_open' then 1 else 0 end)*100 /sum(case when email_category='email_sent' then 1 else 0 end) as email_open_rate,
390     sum(case when email_category = 'email_click' then 1 else 0 end)*100 / sum(case when  email_category ='email_sent' then 1 else 0 end) as email_clcik_rate
391     from
392     (select
393     *,
394      case
395      when action in('email_open') then 'email_open'
396      when action in ('email_clickthrough') then 'email_click'
397      when action in ('sent_weekly_digest' and 'sent_reengagement_email' )then 'email_sent'
398      end as email_category
399      from  email_events)sub;
400
```

total_email_sent	email_open_rate	email_clcik_rate
60920	33.5834	14.7899

Only around 33 percent of the total mails sent were opened and around 15 percent were clicked.

Results

Working on this project helped me to hone my SQL skills which is very important for the analysts and other data related positions.
Further, I gained insights regarding operation analytics and metric spikes by writing queries for the given questions like investigating engagement levels, retention rate, duplicity, user growth etc.

In order to run a company profitably, one must do analysis to make informed decisions. The results generated above after brainstorming and analysing properly the input data sets is used making such decisions.