

Project 3

Operation Analytics and Investigating Metric Spike

Project Description – Operation Analytics is the analysis done for the complete end to end operation of company. with the help of this and Investigating the metric spike company finds the area on which it must improve upon. this is the most important task of every company that helps them to predict the overall growth or decline of company fortunes.

Approach – Created the database and tables using stucture and links provided.

load the large csv files and converted them into the sql table of database.

Used the advance sql queries to analyzing the performance of users and determining the required result.

Tech-stack-used – Tech-stack used in this project is Mysql community server (version 8. 034). It is the best tool for executing the sql queries. it is very user-friendly with simple set-up and accessibility. Microsoft Excel (for import and export of data) and Microsoft Word.

Project Insight

Case Study 1: Job Data Analysis

1. Jobs Reviewed Over Time:

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

Result - SQL query to calculate the number of jobs reviewed per hour for each day in November 2020.

```
14
15
16 • use project_3;
17 • select ds, count(job_id) as jobs_per_day, sum(time_spent) ,
18       sum(time_spent)/3600 as hours_spent
19       from job_data
20       where ds <= '30-11-2020' and ds >= '01-11-2020'
21       group by ds ;
22
```

	ds	jobs_per_day	sum(time_spent)	hours_spent
▶	11/30/2020	2	40	0.0111
	11/29/2020	1	20	0.0056
	11/28/2020	2	33	0.0092
	11/27/2020	1	104	0.0289
	11/26/2020	1	56	0.0156
	11/25/2020	1	45	0.0125

Insight - the unit of time in table given in second so we will convert it into the second and then we determine the no of job reviewed per hour for each day. So on **27 November 2020 maximum hour** spent with execution of 1 job and minimum time spent on **29 November2020 with execution of 1 job**.

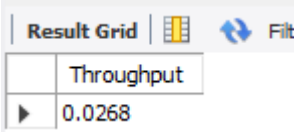
2. Throughput Analysis:

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

Result : SQL query to calculate the 7-day rolling average of throughput.

7 day average throughput

```
#Throughput Analysis
select count(event)/sum(time_spent) as 'Throughput'
from job_data;
```



The screenshot shows a 'Result Grid' with a single column labeled 'Throughput' and a single row containing the value '0.0268'.

Weekly throughput

```
28
29 #Daily Throughput Analysis
30 • select ds,count(event)/sum(time_spent) as 'Daily Throughput'
31 from job_data
32 group by ds
33 order by ds;
34
```

	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

Insight - in first query we calculated the 7-day average throughput which is equal to 0.0268 and in second query we calculated the daily throughput which is **highest for 28 November 2020** and **lowest for 26 November 2020**.

Both the approach has their own importance based on context of analysis. however, **7-day rolling average** is more suitable for identifying trend and providing the better throughput. it handle the daily anomaly and provides the better result with weekly insight.

3. Language Share Analysis:

Objective: Calculate the percentage share of each language in the last 30 days

Result: SQL query to calculate the percentage share of each language over the last 30 days.

```

70
71     #Language Share Analysis
72
73
74 •   select language, 100*count(language)/total as percent
75     from job_data
76     cross join (select count(*) as total from job_data) as sub
77     group by language ,sub.total;
78
79

```

	language	percent
►	English	12.5000
	Arabic	12.5000
	Persian	37.5000
	Hindi	12.5000
	French	12.5000
	Italian	12.5000

Insight – we calculated the percentage share of each language in the last 30 days in which **Persian have the highest percentage share** and rest of the language have equal shares.

4.Duplicate Rows Detection:

Objective: Identify duplicate rows in the data.

Result - SQL query to display duplicate rows from the job data table.

```

82      #Duplicate Rows Detection
83
84 •    select actor_id ,count(*)
85      from job_data
86      group by actor_id having count(*)>1;
87
88
89
90

```

	actor_id	count(*)
►	1003	2

Insight – In this query we calculated the duplicate row based on actor id from the table.so we find out that only **the actor id of 1003 have 2 duplicate row**.

Case Study 2: Investigating Metric Spike

1. Weekly User Engagement:

Objective: Measure the activeness of users on a weekly basis.

Result: SQL query to calculate the weekly user engagement.

```
33      #weekly user engagement
34
35 •    select extract(week from occurred_at)
36      as week ,count(distinct user_id) as active_user
37      from events
38      where event_type = 'engagement'
39      group by week
40      order by week;
41
42
```

Result Grid			Filter Rows:	Export:	Wrap Cell Content
	week	active_user			
▶	17	663			
	18	1068			
	19	1113			
	20	1154			
	21	1121			
	22	1186			
	23	1232			
	24	1275			
	25	1264			
	26	1302			
	27	1372			

Insight – The query shows the weekly user engagement with week in one column and corresponding active users of that week. Output have total 19 rows and **week 30 have highest engagement with total 1467 active users** and **week 35 have lowest engagement with 104 active users.**

2. User Growth Analysis:

Objective: Analyze the growth of users over time for a product.

Result: SQL query to calculate the user growth for the product

```

45     #user growth analysis
46
47 •   select year, week,num_users,sum(num_users)
48     over (order by year,week) as cum_users
49     from
50     (select extract(year from created_at) as year,
51      extract(week from created_at) as week,
52      count(distinct user_id) as num_users from users
53      where state = 'active'
54      group by year ,week
55      order by year ,week)sub;
56

```

	year	week	num_users	cum_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	21	425

Result 27 x

Insight - this query shows the growth of user over the time of product .it result the total number of users and active users on weekly basis of every year.

3.Weekly Retention Analysis:

Objective: Analyze the retention of users on a weekly basis after signing up for a product.

Result: SQL query to calculate the weekly retention of users based on their sign-up cohort.

```

28     #weekly retention analysis
29
30 • with cte1 as(
31     select distinct user_id,
32     Extract(week from occurred_at) as signup_week
33     from events
34     where event_type = 'signup_flow'
35     and event_name = 'complete_signup' and
36     extract(week from occurred_at) = 18 ),
37 • cte2 as (select distinct user_id,
38     Extract(week from occurred_at) as engagement_week
39     from events
40     where event_type = 'engagement')
41     select count(user_id) as total_engaged_users,
42     sum(case when retention_week > 8 then 1 else 0 end )
43     as retained_users
44 • from (select a.user_id,a.signup_week,
45
46     b.engagement_week,b.engagement_week-a.signup_week as retention_week
47     from cte1 a
48     left join cte2 b
49     on a.user_id = b.user_id
50     order by a.user_id)sub;

```

Result Grid	Filter Rows:	Export:	Wrap Ce
	total_engaged_users	retained_users	
▶	615	96	

Insight – it calculates the weekly retention of users based on their sign-up flow event and engagement event by determining the total engaged users which is **615** and then determine the retained users which is equal to **96**.

4.Weekly Engagement Per Device:

Objective: Measure the activeness of users on a weekly basis per device.

Result: SQL query to calculate the weekly engagement per device.

```

16
17
18 #Weekly Engagement Per Device
19
20 with cte as(select extract(year from occurred_at) as year,
21                extract(week from occurred_at) as week,
22                device,count(distinct user_id) as usercount from events
23                where event_type = 'engagement'
24                group by year,week,device
25                order by week)
26 select year ,week,device,usercount
27 from cte;
28

```

	year	week	device	usercount
	2014	17	acer aspire desktop	9
	2014	17	acer aspire notebook	20
	2014	17	amazon fire phone	4
	2014	17	asus chromebook	21
	2014	17	dell inspiron desktop	18
	2014	17	dell inspiron notebook	46
	2014	17	hp pavilion desktop	14
	2014	17	htc one	16
	2014	17	ipad air	27
	2014	17	ipad mini	19
	2014	17	iphone 4s	21
	2014	17	iphone 5	65
	2014	17	iphone 5s	42
	2014	17	kindle fire	6
	2014	17	lenovo thinkpad	86
	2014	17	mac mini	6
	2014	17	macbook air	54
	2014	17	macbook pro	143
	2014	17	nexus 10	16
	2014	17	nexus 5	40

Insight – This query calculates the number of user active on particular device per week.so it determines the count of users where event type is engagement. In **week 30 of 2014 maximum** number of users active on **macbookpro** which is **322**. Lowest number of user is **1** which are active on **week 35 on dell inspiron desktop, acer desktop, hp desktop and Samsung galaxy**.



Project3.xlsx

Output sheet-

5. Email Engagement Analysis:

Objective: Analyze how users are engaging with the email service.

Result: SQL query to calculate the email engagement metrics.

```
1  #Email Engagement Analysis
2
3  • select
4      100*sum(case when email_cat = 'email_open' then 1 else 0 end)/
5      sum(case when email_cat = 'email_sent' then 1 else 0 end) as email_open_rate,
6      100*sum(case when email_cat = 'email_clicked' then 1 else 0 end)/
7      sum(case when email_cat = 'email_sent' then 1 else 0 end) as email_click_rate
8  from(select*,
9  case
10     when action in('sent_weekly_digest','sent_reengagement_email')then 'email_sent'
11     when action in ('email_open') then 'email_open'
12     when action in ('email_clickthrough') then 'email_clicked'
13     end as email_cat
14  from EmailEvents)sub;
15
16
--
```

Result Grid

	email_open_rate	email_click_rate
▶	33.5834	14.7899

Insight – this query determines the users engagement with email services by calculating the email open rate and email click rate .it will tell us that how many users open the email which are send to them and how many users clicked the email. The **email open rate is 33.5834 and email click rate is 14.7899.**

Result

This project gave the depth understanding of the advance sql. It also teaches us how to work with large data and how to get the necessary and required insight from that data.it yield the insightful analyses through deep understanding of sql, join and extract. these finding helps for better decision making

across various domain. It also provides the data driven support for calculating the user engagement, email engagement on weekly basis. These insight plays a very important role in achieving the company objectives.