Data Analytics Project: Operation Analytics and Investigating Metric Spike

Project Description:

Operation Analytics is the analysis done for the complete end to end operations of a company. It is used to find the areas of improvement, predict the overall growth or decline of a company's fortune.

Investigating metric spike is an important part of operation analytics to understand the growth, decline and other insights of the data.

This project insights are to be used by the ops team, support team, marketing team.

Approach:

Based on the provided dataset, and the questions asked by different departments, firstly, the tables were studied, to understand the given data and the relations between each table. Multiple queries were run to get the necessary insights from the data.

Tech-Stack Used:

MySQL Workbench 8.0 CE was used to execute this project. As it provided enough useful set of functions. Also, the connectivity, speed and security of MySQL was suitable for accessing given dataset and run the queries. Also, MS Excel was used for the validation of the outputs derived and for graphical representation of output derived.

Process:

Database named dskdb was created for this project. Tables were created by directly importing the csv files in MySQL Workbench into dskdb database, the type of each column was specified before importing the data.

Insights:

Analysis Performed:

A) **Case Study 1 (Job Data)**:

1. **Number of jobs reviewed:**
    - For this task, we have to the number of jobs reviewed per hour per day for November 2020
    - Below query was used to get the total number of jobs reviewed every day, and the time spent on hourly basis per day on the review.

    Query:

    use dskdb;

    select distinct ds as "Date",

    count(job_id) as "Total Number of Jobs Reviewed", sum(time_spent)/3600 as "Time_Spent_in_Hours",

round(count(distinct job_id)/(sum(time_spent)/3600)) as "Jobs Reviewed Per Hour Per Day" from job_data

group by ds;

Output:

Date	Total Number of Jobs Reviewed	Time_Spent_in_Hours	Jobs Reviewed Per Hour Per Day
2020-11-25 00:00:00	1	0.0125	80
2020-11-26 00:00:00	1	0.0156	64
2020-11-27 00:00:00	1	0.0289	35
2020-11-28 00:00:00	2	0.0092	218
2020-11-29 00:00:00	1	0.0056	180
2020-11-30 00:00:00	2	0.0111	180

From the output table, we conclude that on 28 Nov 2020 maximum number of jobs were reviewed that is 218, and on 27 Nov 2020 least number of jobs were reviewed that is 35.

2. **Throughput:**
It is the no. of events happening per second.
The question is
a) To calculate 7 day rolling average of throughput?
b) For throughput, do you prefer daily metric or 7-day rolling and why?

For this, calculation we use below query:

Query:

SELECT ds as "Date", (count(event)/sum(time_spent)) as "Daily Average of Throughput",

SUM((count(event)/sum(time_spent))) OVER (ORDER BY ds) AS "Rolling Total of Throughput",

avg((count(event)/sum(time_spent))) over (order by ds rows between 6 preceding and current row) as "7 day Rolling Average" FROM job_data group by ds;

Output:

Date	Daily Average of Throughput	Rolling Total of Throughput	7 day Rolling Average
2020-11-25 00:00:00	0.0222	0.0222	0.02222222
2020-11-26 00:00:00	0.0179	0.0401	0.02003968
2020-11-27 00:00:00	0.0096	0.0497	0.01656492
2020-11-28 00:00:00	0.0606	0.1103	0.02757520
2020-11-29 00:00:00	0.0500	0.1603	0.03206016
2020-11-30 00:00:00	0.0500	0.2103	0.03505013

As seen in the output table above, daily average results have huge difference between days, whereas 7 day rolling average shows a stable graph for the throughput. That's why, 7 day rolling average is to be preferred.

3. **Percentage share of each language:**

The question is to find the percentage share of each language in last 30 days
For this task, following query was used:

Query:
use dskdb;
SELECT distinct language, count (*) as "Content Count",
sum (count (*)) over () as "Total Content",
(count (*)/sum (count (*)) over () *100) as "Percentage"
FROM dskdb.job_data
group by language;

Output:

Result Grid | Filter Rows: | Export:

	language	Content Count	Total Content	Percentage
▶	English	1	8	12.5000
	Arabic	1	8	12.5000
	Persian	3	8	37.5000
	Hindi	1	8	12.5000
	French	1	8	12.5000
	Italian	1	8	12.5000

As shown in the table above, Persian language has highest share with 37.5 %, all other language has 12.5% share.

4. **Duplicate rows:**

For this task, we check for the duplicate rows in the given table by using the query below:

Query:
use dskdb;
select * from
(
select *, row_number()over(partition by ds,job_id,actor_id,event,language,time_spent, org) as rownum from job_data) a where rownum>1;
;

Output:

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

	ds	job_id	actor_id	event	language	time_spent	org	rownum

The above output table with no records states that there are no duplicate rows in the table.

However, individual columns having unique ids like column job_id and column actor_id, may have duplicates, which can be checked using the query below:

<u>Query (a):</u>

use dskdb;

```
select * from
(
select *, row_number()over(partition by job_id) as rownum
from job_data
) a
where rownum>1;
;
```

Output:

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

ds	job_id	actor_id	event	language	time_spent	org	rownum
2020-11-28 00:00:00	23	1005	transfer	Persian	22	D	2
2020-11-26 00:00:00	23	1004	skip	Persian	56	A	3

As shown in the output above, job_id 23 has 2 duplicates in the column.

<u>Query (b):</u>

```
select * from
(
select *, row_number()over(partition by actor_id) as rownum
from job_data
) a
where rownum>1;
;
```

Output:

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

ds	job_id	actor_id	event	language	time_spent	org	rownum
2020-11-25 00:00:00	20	1003	transfer	Italian	45	C	2

As shown in the table, actor_id 1003 is duplicate in the column

B) Case Study 2 (Investigating metric spike):

1. **User Engagement**:
   For this task, we calculate the weekly user engagement, by using the query given below:

   Query:
   SELECT week(occurred_at) as Week,event_name,
   count(distinct user_id) as "User Engagement"
   FROM dskdb.events
   group by week(occurred_at), event_name;

   The output:

Week	event_name	User Engagement
30	login	1467
29	login	1376
27	login	1372
28	login	1365
30	home_page	1362
26	login	1302
31	login	1299
27	home_page	1282
24	login	1275
25	login	1264
28	home_page	1259
29	home_page	1257
23	login	1232
32	login	1225
33	login	1225
26	home_page	1204

The above output table states that in week 30, highest number of users, i.e 1467 users logged in, the table also shows that the highly used service is login and home_page.

2. **User Growth**:
   For this task, we find the amount of users growing over time for a product, using the beow query:

Query:

SELECT Year, Month, Users,Active_Users,

sum(Users) over (order by Year,Month rows between unbounded preceding and current row) as Total_Users_Growth,

sum(Active_Users) over (order by Year,Month rows between unbounded preceding and current row) as Active_Users_Growth

from (select year(created_at) as Year, month(created_at) as Month,

count(distinct user_id) as Users,

count(distinct(case when state="active" then user_id end)) as Active_Users

from dskdb.users

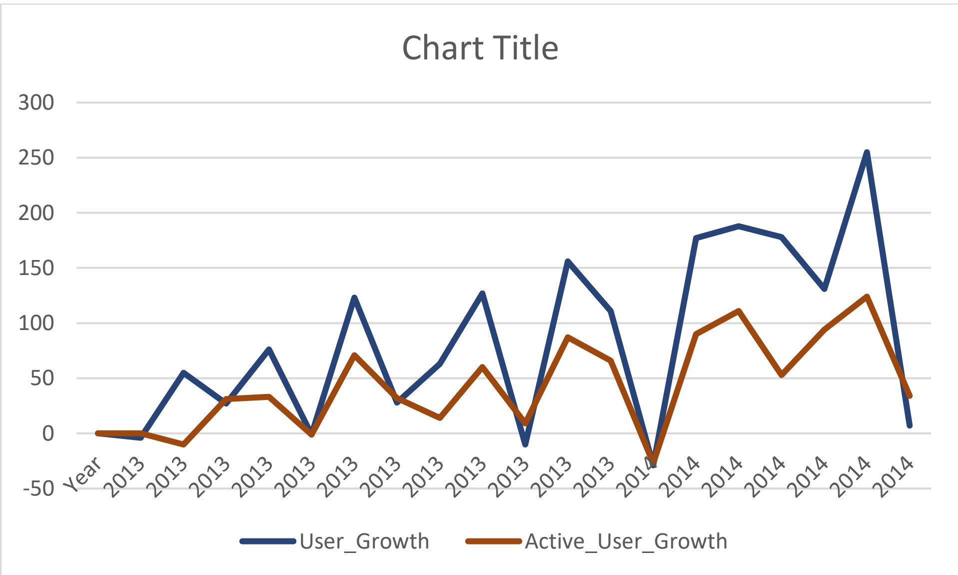
group by Year, Month) a;

The output:

Year	Month	Users	Active_Users	Cummulative_Users_Total	Cummulative_Active_Total	User_Growth	Active_User_Growth
2013	1	332	160	332	160	NULL	NULL
2013	2	328	160	660	320	-4	0
2013	3	383	150	1043	470	55	-10
2013	4	410	181	1453	651	27	31
2013	5	486	214	1939	865	76	33
2013	6	485	213	2424	1078	-1	-1
2013	7	608	284	3032	1362	123	71
2013	8	636	316	3668	1678	28	32
2013	9	699	330	4367	2008	63	14
2013	10	826	390	5193	2398	127	60
2013	11	816	399	6009	2797	-10	9
2013	12	972	486	6981	3283	156	87
2014	1	1083	552	8064	3835	111	66
2014	2	1054	525	9118	4360	-29	-27
2014	3	1231	615	10349	4975	177	90
2014	4	1419	726	11768	5701	188	111
2014	5	1597	779	13365	6480	178	53

The above output shows the growth of all users and active users.

When put in excel graph, it shows a clear growth rate as shown in below graph:



The above graph was derived using Excel charts, to show the growth over the years

3. **Weekly Retention**:

The task is to calculate the weekly retention of users-sign up cohort. For this first we get the list of users who have completed the sign-up using the query below:

Query:

```
SELECT * FROM dskdb.events
where event_type="signup_flow" and event_name="complete_signup";
```

The output:

user_id	occurred_at	event_type	event_name	location	device	user_type
11768	2014-05-01 08:03:00	signup_flow	complete_signup	France	macbook pro	3
11770	2014-05-01 06:08:00	signup_flow	complete_signup	Japan	iphone 5s	3
11775	2014-05-01 16:38:00	signup_flow	complete_signup	United Kingdom	lenovo thinkpad	2
11778	2014-05-01 18:49:00	signup_flow	complete_signup	Indonesia	iphone 4s	3
11779	2014-05-01 18:24:00	signup_flow	complete_signup	Germany	samsung galaxy s4	1
11780	2014-05-01 10:34:00	signup_flow	complete_signup	United States	iphone 4s	3

The above table gives the list of users who have completed sign-up.

Now to calculate the weekly retention of signed-up users, we use the query below:

Query:

```
select week(occurred_at) as Week, count(distinct user_id) as "Users Weekly Retented" from
dskdb.events
where event_type="engagement"
and user_id in
(
SELECT user_id FROM dskdb.events
where event_type="signup_flow" and event_name="complete_signup"
)
group by week(occurred_at)
;
```

The output:

Week	Users Weekly Retented
17	72
18	222
19	323
20	407
21	444
22	515
23	541
24	600
25	616
26	644
27	716
28	725
29	743
30	809
31	734
32	763

The above table shows the numbers of users getting retained weekly after signing-up for a product.

4. **Weekly Engagement:**
   The question is to find the weekly engagement per device. For this we use the query below:

   Query:
   SELECT week(occurred_at) as Week, device, count(distinct user_id) as "Engagement"
    FROM dskdb.events
    group by Week, device;

   The output:

Week	device	Engagement
17	acer aspire desktop	12
17	acer aspire notebook	23
17	amazon fire phone	4
17	asus chromebook	23
17	dell inspiron desktop	20
17	dell inspiron notebook	48
17	hp pavilion desktop	17
17	htc one	19
17	ipad air	29
17	ipad mini	19
17	iphone 4s	27
17	iphone 5	69
17	iphone 5s	47
17	kindle fire	6
17	lenovo thinkpad	94
17	mac mini	7
17	macbook air	61

   The above table shows the weekly engagement per device, device macbook has the highest engagement.

5. **Email Engagement:**
   For this task the question is to find the users engaging with email service. Below query is used for that:

   Query:
   SELECT action, week(occurred_at) as week,
   count(distinct user_id) as "Users_Engagement" FROM dskdb.email_events
   group by action, week(occurred_at)
   order by action, week(occurred_at)
   ;

   The output:

action	week	Users_Engagement
email_clickthrough	17	166
email_clickthrough	18	425
email_clickthrough	19	476
email_clickthrough	20	501
email_clickthrough	21	436
email_clickthrough	22	478
email_clickthrough	23	529
email_clickthrough	24	549
email_clickthrough	25	524
email_clickthrough	26	550
email_clickthrough	27	613

action	week	Users_Engagement
sent_weekly_digest	34	4111
sent_weekly_digest	33	4012
sent_weekly_digest	32	3897
sent_weekly_digest	31	3793
sent_weekly_digest	30	3706
sent_weekly_digest	29	3592
sent_weekly_digest	28	3499
sent_weekly_digest	27	3399
sent_weekly_digest	26	3302
sent_weekly_digest	25	3207
sent_weekly_digest	24	3105

The above output shows the email enagagement metrics, as shown in the table sent_weekly_digest has highest engegement

Insights:

This project is based on 2 datasets. The first dataset is the data for actors with details on the jobs assigned to actors, time spent, language and other details. The outputs derived from this dataset help understand throughput of the events, the share of language and number of jobs reviewed.

The second dataset is the data of an email service used by the users. The tasks and questions answered above help gain the insights on the user behaviour, user engagement with the service and service performance.

The graph derived above on user growth shows the growth rate over the period of 2 years.

Result:

This project has helped me learn how to import huge dataset, clean the dataset and how to get insights from the data. It has helped me practice the queries and get hands on experience on analysing the data as well as how to derive meaningful graphs from the tables.