Operation Analytics and Investigating Metric Spike

- By Pilli Bhavana

PROJECT DESCRIPTION

Operation Analytics is the analysis of a company's end-to-end operations to identify areas for improvement. Data Analysts must be able to investigate metric spikes and derive insights from data sets to answer questions asked by different departments. This is an important part of a company's success, as it means better automation, understanding between cross-functional teams, and more effective workflows.

Case Study 1 (Job Data)

A. **Number of jobs reviewed:** Amount of jobs reviewed over time.
   Your task: Calculate the number of jobs reviewed per hour per day for November 2020?

B. **Throughput:** It is the no. of events happening per second.
   Your task: Let's say the above metric is called throughput. Calculate 7 day rolling average of throughput? For throughput, do you prefer daily metric or 7-day rolling and why?

C. **Percentage share of each language:** Share of each language for different contents.
   Your task: Calculate the percentage share of each language in the last 30 days?

D. **Duplicate rows:** Rows that have the same value present in them.
   Your task: Let's say you see some duplicate rows in the data. How will you display duplicates from the table?

Case Study 2 (Investigating metric spike)

A. **User Engagement:** To measure the activeness of a user. Measuring if the user finds quality in a product/service.
   Your task: Calculate the weekly user engagement?

B. **User Growth:** Amount of users growing over time for a product.
   Your task: Calculate the user growth for product?

C. **Weekly Retention:** Users getting retained weekly after signing-up for a product.
   Your task: Calculate the weekly retention of users-sign up cohort?

D. **Weekly Engagement:** To measure the activeness of a user. Measuring if the user finds quality in a product/service weekly.
   Your task: Calculate the weekly engagement per device?

E. **Email Engagement:** Users engaging with the email service.
   Your task: Calculate the email engagement metrics?

APPROACH

- I created the data for job_data using MS-Excel.
- I imported the data into MySQL and then ran several queries to acquire the response required and to get familiar with the data.
- I worked in PowerPoint after completing the MySQL query portion of the report submission.
- After writing down the information to be included on each slide in a notebook, I inserted the information to the PPT after reviewing it.

TECHSTACK USED

MICROSOFT EXCEL (VERSION: MICROSOFT 360): FOR CREATING THE DATA

MYSQL WORKBENCH (VERSION:8.0 CE): FOR PERFORMING THE ANALYSIS

MICROSOFT POWERPOINT (VERSION: MICROSOFT 360): FOR CREATING THE REPORT

INSIGHTS

I learned how to use the tools I was using, such as PowerPoint, Excel and MySQL, in practical job situations as I used them.

I created 25 rows data where I learned how to do that, how you can only use the conditions given and create data.

I experienced the challenges that a real data analyst encounters when working on projects while I was working on the data analysis part of the project as there were Null values and while creating table you must have a deep knowledge of the data you are working with.

I now understand how the industry operates to use data to find any answers they desire and how cleverly they use the tools.

RESULT

Case Study 1: Job Data

Number of jobs Reviewed

```
SELECT COUNT(DISTINCT job_id)/SUM(time_spent/3600)/COUNT(ds)
AS jobs_reviewed_per_hour_per_day_in_nov_2020
FROM job_data
WHERE ds BETWEEN "2020-11-01" AND "2020-11-30";
```

jobs_reviewed_per_hour_per_day_in_nov_2020
0.32099458

Throughput

```
SELECT job_id, `date`, event_per_day,
AVG(event_per_day)OVER(ORDER BY `date` ROWS BETWEEN 6 PRECEDING AND CURRENT
ROW)AS 7_day_rolling_avg
FROM
(SELECT job_id, ds AS `date`,
COUNT(DISTINCT `event`) AS event_per_day
FROM job_data
GROUP BY `date`
ORDER BY `date`)a;
```

job_id	date	event_per_day	7_day_rolling_avg
21	2020-10-30	1	1.0000
22	2020-11-01	1	1.0000
25	2020-11-02	1	1.0000
23	2020-11-03	1	1.0000
11	2020-11-04	1	1.0000
22	2020-11-05	1	1.0000
24	2020-11-06	1	1.0000
21	2020-11-07	1	1.0000
25	2020-11-08	1	1.0000
11	2020-11-09	2	1.1429
25	2020-11-10	1	1.1429
21	2020-11-11	1	1.1429
21	2020-11-12	1	1.1429
24	2020-11-13	1	1.1429
23	2020-11-14	1	1.1429
11	2020-11-15	1	1.1429
11	2020-11-16	1	1.0000
25	2020-11-17	1	1.0000
22	2020-11-18	1	1.0000
21	2020-11-19	1	1.0000
22	2020-11-20	1	1.0000
24	2020-11-21	1	1.0000
20	2020-11-22	2	1.1429
11	2020-11-23	1	1.1429
24	2020-11-24	1	1.1429
20	2020-11-25	1	1.1429
23	2020-11-26	1	1.1429
11	2020-11-27	1	1.1429
25	2020-11-28	2	1.2857
23	2020-11-29	1	1.1429
22	2020-11-30	2	1.2857

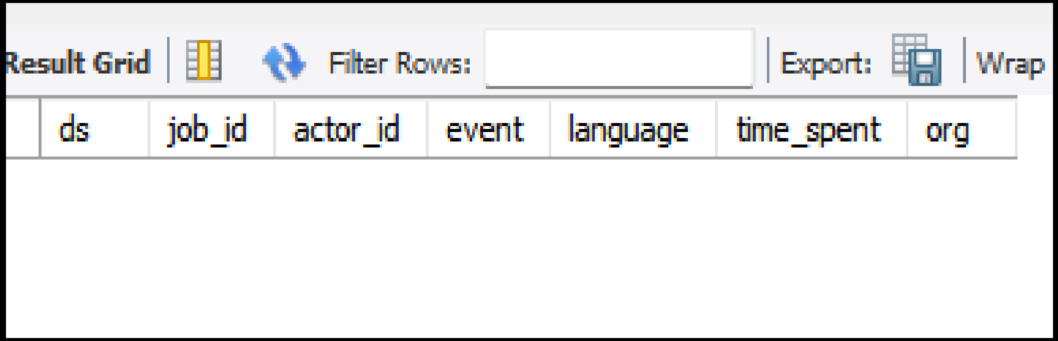
Percentage share of each language

```
SELECT `language`,
COUNT(*) * 100.0 / (SELECT COUNT(*)
FROM job_data ) AS
percentage_share_of_language
FROM job_dataGROUP BY `language`;
```

language	percentage_share_of_language
Arabic	14.28571
English	17.14286
French	17.14286
Hindi	17.14286
Italian	5.71429
Korean	11.42857
Persian	17.14286

Duplicate rows

```
SELECT *
FROM job_data
GROUP BY ds , job_id, actor_id, `event`, `language`, time_spent, org
HAVING COUNT(ds) >1
AND COUNT(job_id) >1
AND COUNT(actor_id) >1
AND COUNT(`event`) >1
AND COUNT(`language`) >1
AND COUNT(time_spent) >1
AND COUNT(org) >1;
```



ds	job_id	actor_id	event	language	time_spent	org

NOTE: the data I created has no duplicate rows so no values are showing after running the query.

Case Study 2: Investigating Metric Spike

User Engagement

```
SELECT COUNT(DISTINCT user_id) AS
active_users,
WEEK(occurred_at) AS `week`
FROM `events`
WHERE event_type= 'engagement’
GROUP BY `week`;
```

active_users	week
663	17
1068	18
1113	19
1154	20
1121	21
1186	22
1232	23
1275	24
1264	25
1302	26

active_users	week
1372	27
1365	28
1376	29
1467	30
1299	31
1225	32
1225	33
1204	34
104	35

User Growth

```
SELECT COUNT(user_id) AS increase_of_user,
YEARWEEK(created_at) AS week_of_year
FROM users
GROUP BY week_of_year;
```

increase_of_user	week_of_year
52	201253
68	201301
76	201302
77	201303
75	201304
87	201305
80	201306
83	201307
81	201308
84	201309
88	201310
95	201311
92	201312
86	201313
96	201314
93	201315
100	201316
102	201317
105	201318
108	201319
104	201320
113	201321
112	201322
116	201323
118	201324
127	201325
127	201326
132	201327
141	201328
130	201329
141	201330
131	201331
148	201332
151	201333
149	201334
164	201335
164	201336
164	201337
166	201338
180	201339
174	201340
172	201341
191	201342
195	201343
194	201344
191	201345
179	201346
207	201347
213	201348
216	201349
221	201350
235	201351
232	201352
232	201401
223	201402
248	201403
247	201404
254	201405
264	201406
270	201407
269	201408
269	201409
289	201410
287	201411
299	201412
310	201413
315	201414
302	201415
365	201416
348	201417
355	201418
359	201419
373	201420
361	201421
391	201422
410	201423
425	201424
402	201425
408	201426
423	201427
427	201428
452	201429
477	201430
408	201431
474	201432
474	201433
498	201434
32	201435

Weekly Retention

```
SELECT COUNT(DISTINCT(users.user_id)) AS retained_users,
YEARWEEK(users.created_at) AS `week`
FROM `events`
INNER JOIN users ON
users.user_id=`events`.user_id
WHERE event_type='signup_flow’
GROUP BY `week`;
```

retained_users	week
149	201417
355	201418
359	201419
373	201420
361	201421
391	201422
410	201423
425	201424
402	201425
408	201426
423	201427
427	201428
452	201429
477	201430
408	201431
474	201432
474	201433
498	201434
32	201435

Weekly Engagement

```
SELECT COUNT(user_id) AS no_of_users,
YEAR(occurred_at) AS `year`,
WEEK(occurred_at) AS `week`,
device
FROM `events`
WHERE `events`.event_type= 'engagement’
GROUP BY `year`, `week`, device
ORDER BY `year`, `week`, device;
```

NOTE: This query returns 491 rows in the next slide I attached the first 20 rows

no_of_users	year	week	device
67	2014	17	acer aspire desktop
206	2014	17	acer aspire notebook
83	2014	17	amazon fire phone
251	2014	17	asus chromebook
187	2014	17	dell inspiron desktop
503	2014	17	dell inspiron notebook
132	2014	17	hp pavilion desktop
190	2014	17	htc one
330	2014	17	ipad air
205	2014	17	ipad mini
217	2014	17	iphone 4s
706	2014	17	iphone 5
473	2014	17	iphone 5s
57	2014	17	kindle fire
793	2014	17	lenovo thinkpad
59	2014	17	mac mini
490	2014	17	macbook air
1516	2014	17	macbook pro
145	2014	17	nexus 10
382	2014	17	nexus 5

no_of_users	year	week	device
177	2014	17	nexus 7
128	2014	17	nokia lumia 635
70	2014	17	samsumg galaxy tablet
116	2014	17	samsung galaxy note
449	2014	17	samsung galaxy s4
87	2014	17	windows surface
295	2014	18	acer aspire desktop
363	2014	18	acer aspire notebook
177	2014	18	amazon fire phone
523	2014	18	asus chromebook
683	2014	18	dell inspiron desktop
953	2014	18	dell inspiron notebook
373	2014	18	hp pavilion desktop
174	2014	18	htc one
520	2014	18	ipad air
309	2014	18	ipad mini
448	2014	18	iphone 4s
1328	2014	18	iphone 5
778	2014	18	iphone 5s
265	2014	18	kindle fire

Email Engagement

```
SELECT `action`,
YEARWEEK(occurred_at) AS `week`,
COUNT(distinct user_id) AS users_engaging
FROM email_events
GROUP BY `action`, `week`
ORDER BY `action`, `week`;
```

NOTE: This query returns 75 rows in the next slide I attached the first 60 rows

action	week	users_engaging
email_clickthrough	201417	166
email_clickthrough	201418	425
email_clickthrough	201419	476
email_clickthrough	201420	501
email_clickthrough	201421	436
email_clickthrough	201422	478
email_clickthrough	201423	529
email_clickthrough	201424	549
email_clickthrough	201425	524
email_clickthrough	201426	550
email_clickthrough	201427	613
email_clickthrough	201428	594
email_clickthrough	201429	583
email_clickthrough	201430	625
email_clickthrough	201431	444
email_clickthrough	201432	416
email_clickthrough	201433	490
email_clickthrough	201434	481
email_clickthrough	201435	38
email_open	201417	310
email_open	201418	900
email_open	201419	961
email_open	201420	989
email_open	201421	996
email_open	201422	965
email_open	201423	1057
email_open	201424	1136
email_open	201425	1084
email_open	201426	1149
email_open	201427	1207
email_open	201428	1228
email_open	201429	1201
email_open	201430	1363
email_open	201431	1338
email_open	201432	1318
email_open	201433	1417
email_open	201434	1502
email_open	201435	41
sent_reengagement_email	201417	73
sent_reengagement_email	201418	157
sent_reengagement_email	201419	173
sent_reengagement_email	201420	191
sent_reengagement_email	201421	164
sent_reengagement_email	201422	192
sent_reengagement_email	201423	197
sent_reengagement_email	201424	226
sent_reengagement_email	201425	196
sent_reengagement_email	201426	219
sent_reengagement_email	201427	213
sent_reengagement_email	201428	213
sent_reengagement_email	201429	213
sent_reengagement_email	201430	231
sent_reengagement_email	201431	222
sent_reengagement_email	201432	200
sent_reengagement_email	201433	264
sent_reengagement_email	201434	261
sent_reengagement_email	201435	48
sent_weekly_digest	201417	908
sent_weekly_digest	201418	2602
sent_weekly_digest	201419	2665

DRIVE LINK

Drive link of the csv file of MySQL query I wrote.

https://drive.google.com/file/d/1ZFxEf5yNzSu2WsfkyL0nxqSmPy6ntqzl/view?usp=share_link

THANK YOU