Operations Analytics and Investigating Metric Spike

Submitted by – Akhilesh Chaturvedi
Date: 10 Jan 2023

Project Description

Operations Analytics and Investigating metric spike both are crucial techniques for analysing any Social Media business.
In Operations Analytics overall functioning of a company is investigated for taking better performance decision in future. For any Social media platform how users of the product interact with their product is essential for the success of their product.
For this purpose, User Engagement analysis, Ad-performance analysis, Social listening and Content moderation are some important metric. In Metric Spike, User engagement, customer retention rate is something very important metrics for any business.

Approach

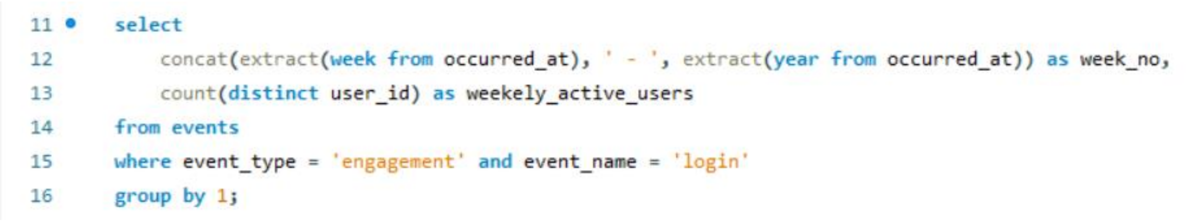
To solve the following case study I used MySQL workbench. Three tables having names, "users", "events", "email_events" were uploaded to MySQL. And queried the data for respective problems accordingly.

Tech-Stack Used: MySQL Workbench 8

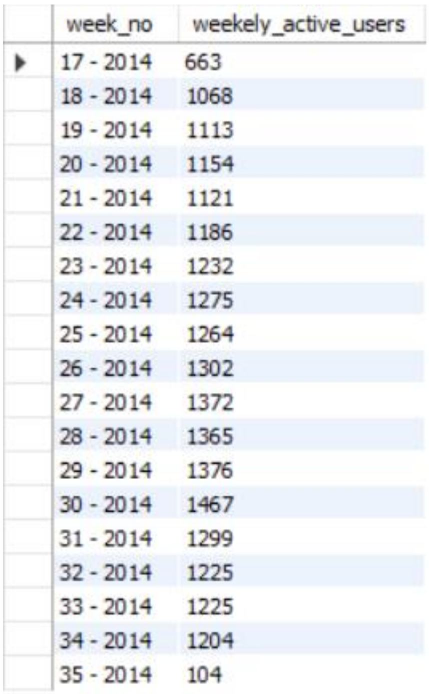
Problem Statements

Case Study (Investigating metric spike)

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

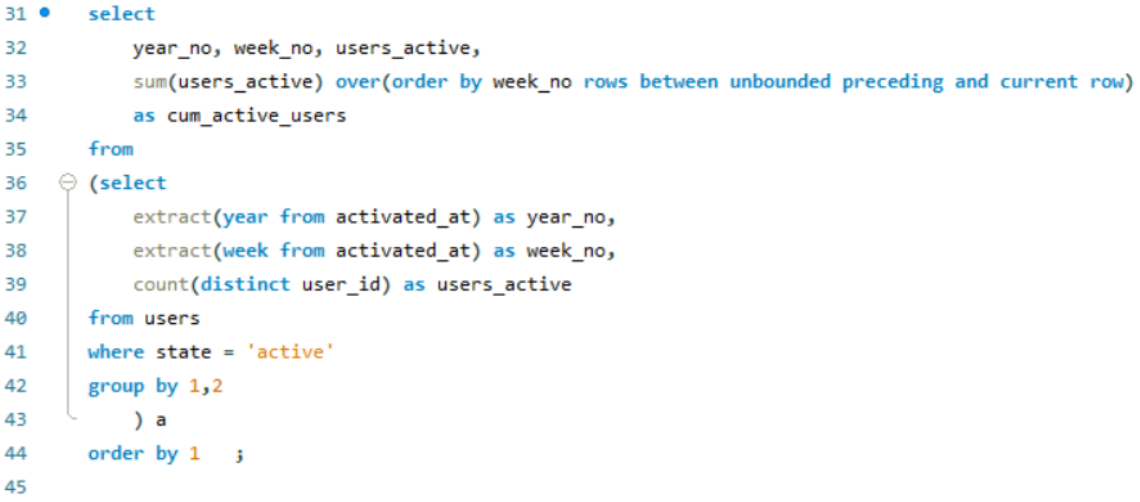


```
select
    concat(extract(week from occurred_at), ' - ', extract(year from occurred_at)) as week_no,
    count(distinct user_id) as weekely_active_users
from events
where event_type = 'engagement' and event_name = 'login'
group by 1;
```



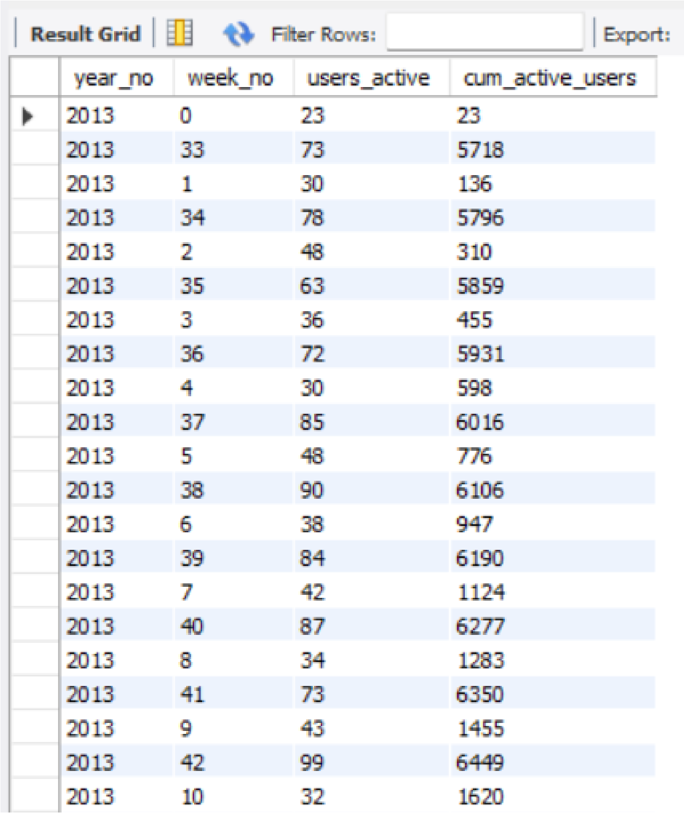
week_no	weekely_active_users
17 - 2014	663
18 - 2014	1068
19 - 2014	1113
20 - 2014	1154
21 - 2014	1121
22 - 2014	1186
23 - 2014	1232
24 - 2014	1275
25 - 2014	1264
26 - 2014	1302
27 - 2014	1372
28 - 2014	1365
29 - 2014	1376
30 - 2014	1467
31 - 2014	1299
32 - 2014	1225
33 - 2014	1225
34 - 2014	1204
35 - 2014	104

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



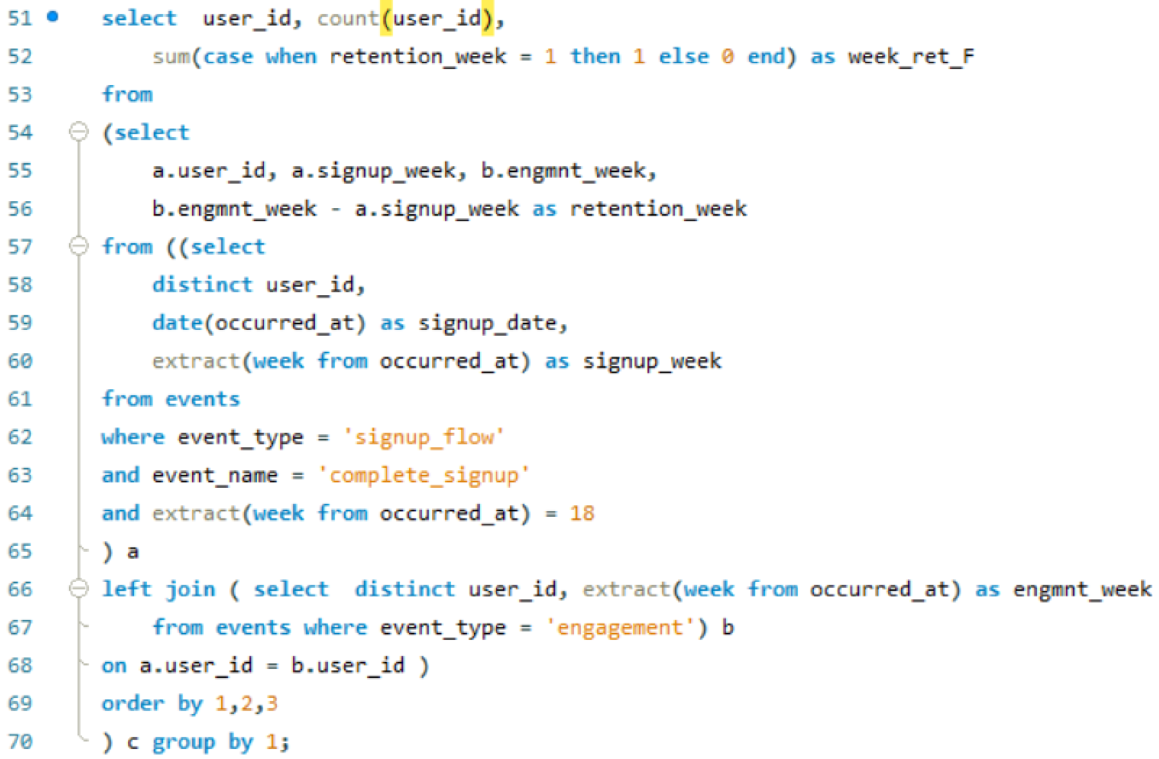
```
select
    year_no, week_no, users_active,
    sum(users_active) over(order by week_no rows between unbounded preceding and current row)
    as cum_active_users
from
(select
    extract(year from activated_at) as year_no,
    extract(week from activated_at) as week_no,
    count(distinct user_id) as users_active
from users
where state = 'active'
group by 1,2
    ) a
order by 1    ;
```

#Output



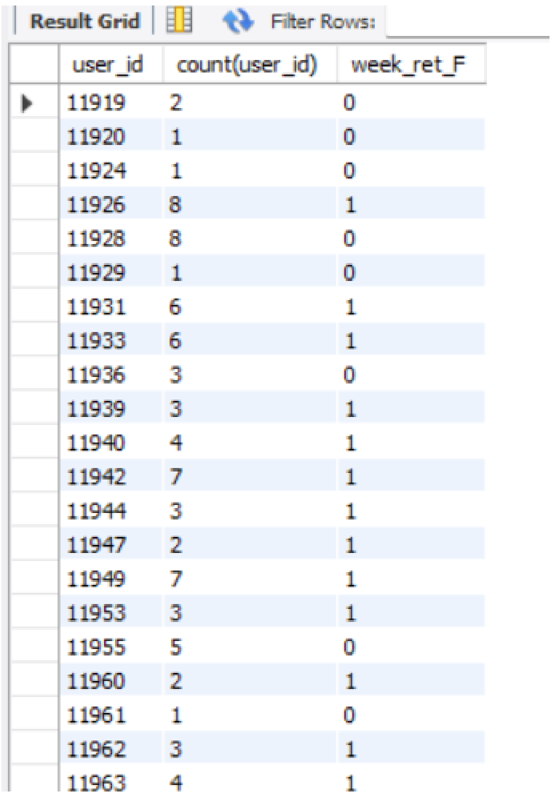
year_no	week_no	users_active	cum_active_users
2013	0	23	23
2013	33	73	5718
2013	1	30	136
2013	34	78	5796
2013	2	48	310
2013	35	63	5859
2013	3	36	455
2013	36	72	5931
2013	4	30	598
2013	37	85	6016
2013	5	48	776
2013	38	90	6106
2013	6	38	947
2013	39	84	6190
2013	7	42	1124
2013	40	87	6277
2013	8	34	1283
2013	41	73	6350
2013	9	43	1455
2013	42	99	6449
2013	10	32	1620

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



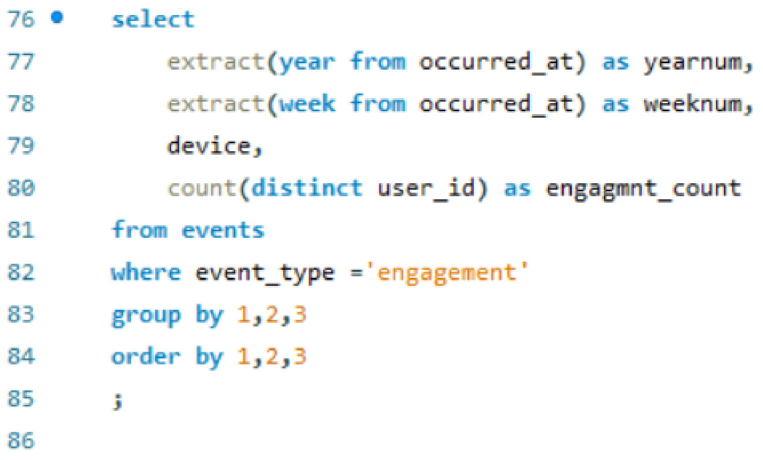
```
select  user_id, count(user_id),
    sum(case when retention_week = 1 then 1 else 0 end) as week_ret_F
from
(select
    a.user_id, a.signup_week, b.engmnt_week,
    b.engmnt_week - a.signup_week as retention_week
from ((select
    distinct user_id,
    date(occurred_at) as signup_date,
    extract(week from occurred_at) as signup_week
from events
where event_type = 'signup_flow'
and event_name = 'complete_signup'
and extract(week from occurred_at) = 18
) a
left join ( select  distinct user_id, extract(week from occurred_at) as engmnt_week
    from events where event_type = 'engagement') b
on a.user_id = b.user_id )
order by 1,2,3
) c group by 1;
```

Output



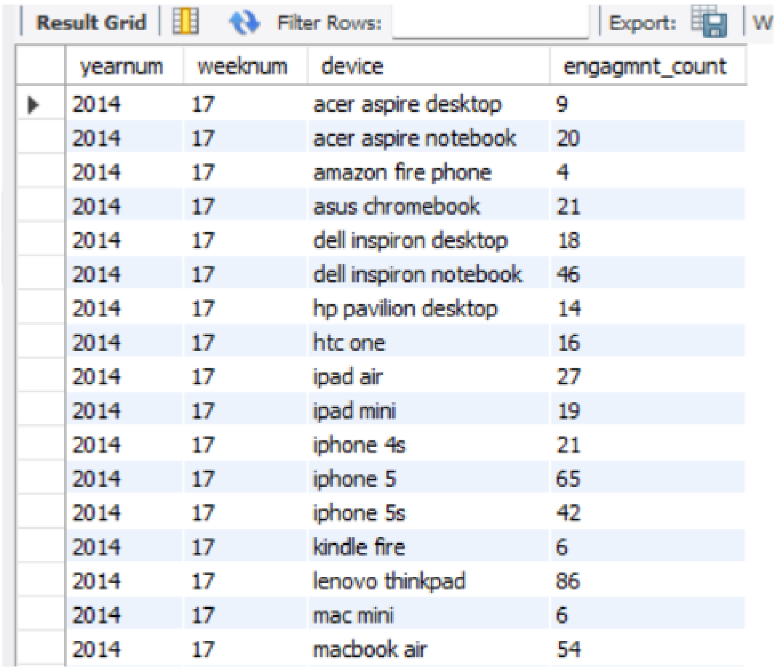
user_id	count(user_id)	week_ret_F
11919	2	0
11920	1	0
11924	1	0
11926	8	1
11928	8	0
11929	1	0
11931	6	1
11933	6	1
11936	3	0
11939	3	1
11940	4	1
11942	7	1
11944	3	1
11947	2	1
11949	7	1
11953	3	1
11955	5	0
11960	2	1
11961	1	0
11962	3	1
11963	4	1

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



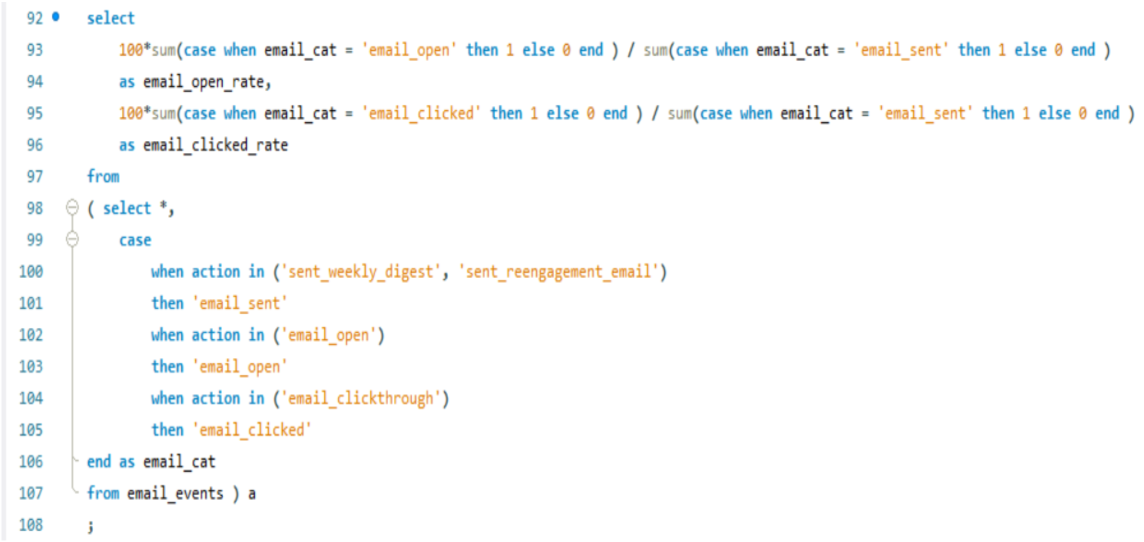
```
select
    extract(year from occurred_at) as yearnum,
    extract(week from occurred_at) as weeknum,
    device,
    count(distinct user_id) as engagmnt_count
from events
where event_type ='engagement'
group by 1,2,3
order by 1,2,3
;
```

Output



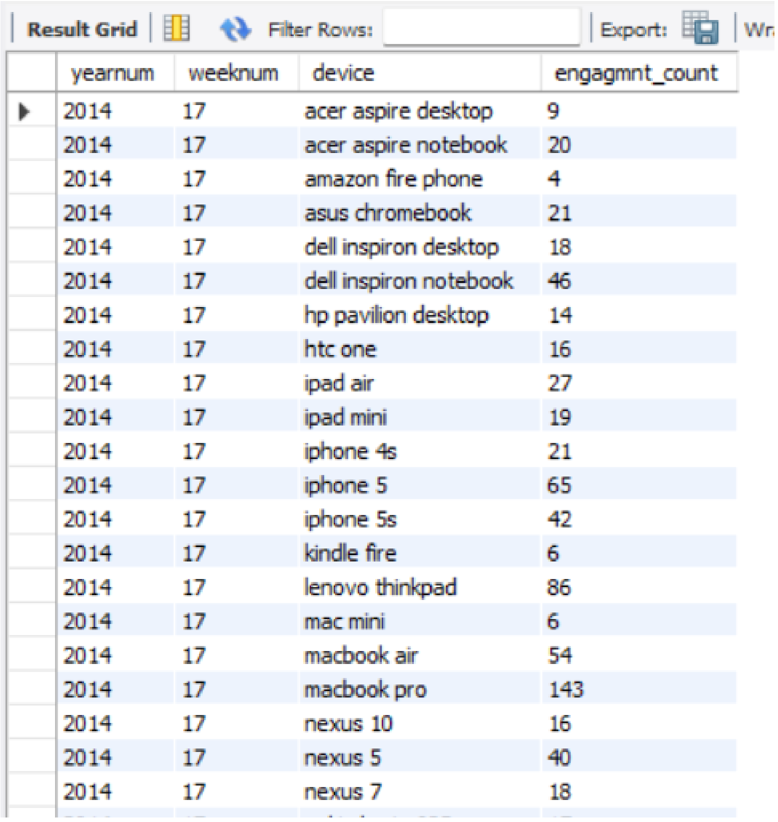
yearnum	weeknum	device	engagmnt_count
2014	17	acer aspire desktop	9
2014	17	acer aspire notebook	20
2014	17	amazon fire phone	4
2014	17	asus chromebook	21
2014	17	dell inspiron desktop	18
2014	17	dell inspiron notebook	46
2014	17	hp pavilion desktop	14
2014	17	htc one	16
2014	17	ipad air	27
2014	17	ipad 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

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



```
select
    100*sum(case when email_cat = 'email_open' then 1 else 0 end ) / sum(case when email_cat = 'email_sent' then 1 else 0 end )
    as email_open_rate,
    100*sum(case when email_cat = 'email_clicked' then 1 else 0 end ) / sum(case when email_cat = 'email_sent' then 1 else 0 end )
    as email_clicked_rate
from
( select *,
    case
        when action in ('sent_weekly_digest', 'sent_reengagement_email')
        then 'email_sent'
        when action in ('email_open')
        then 'email_open'
        when action in ('email_clickthrough')
        then 'email_clicked'
end as email_cat
from email_events ) a
;
```

# Output



yearnum	weeknum	device	engagmnt_count
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
2014	17	nexus 7	18