*** Project ***

Operation Analytics And Investigating Metric Spike

(Insights)

Project Description

- The given project consists of 2 case studies:-
- First is regarding Operation Analytics where job data is provided and number of jobs reviewed , 7day rolling average of throughput, percentage share of language used and duplicates are found out.
- Second is Investigating Metric Spike where user engagement, user growth, weekly retention, weekly engagement and email engagement is determined.
- The following information is found with the help of SQL queries.

Case Study 1

Operation Analytics

--- Insights ---

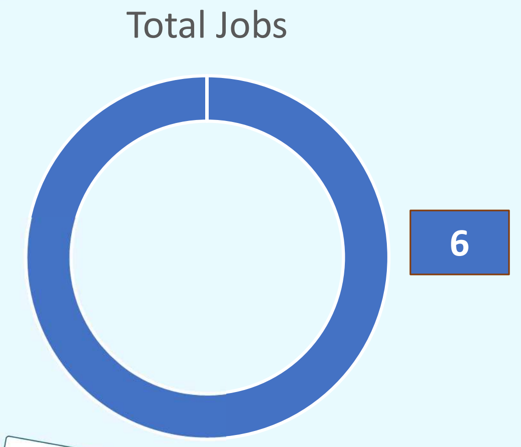
Approach

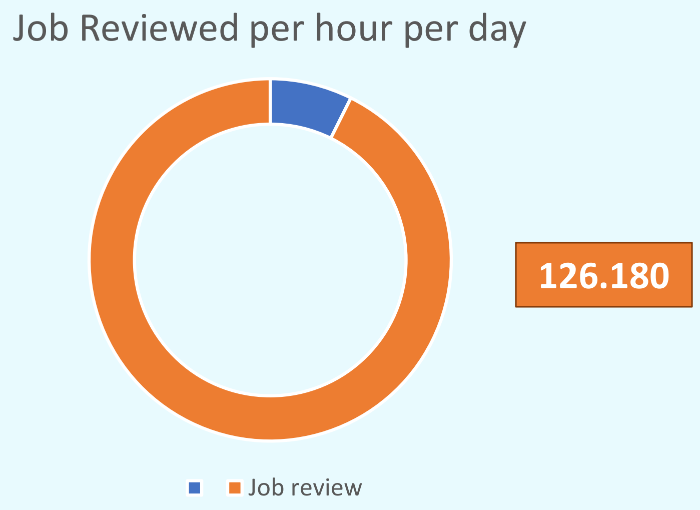
The required information was determined via SQL queries where the data base was created first in SQL and moreover for the second case study due to the size of the data excel was used to make charts for better visualisation.

Tech stack used

- **MySQL** was used to run the queries.
- The language was selected because of comfort and experience in the same.
- **MS Excel** was used in the second case study for better visualisation.
- As I am currently learning this tool, it was utilised to get more hands on experience.

Jobs Reviewed Over Time





Jobs Reviewed Over Time

Queries:

```
SELECT     COUNT(*) AS total_jobs,
            avg(t) as job_reviewed_per_hr_per_day
FROM
   (SELECT
      ds,
     (COUNT(job_id) * 3600) / (SUM(time_spent)) AS t
   FROM
     job_data
   WHERE
     month(ds) = 11
   GROUP BY ds) a;
```

Throughput Analysis

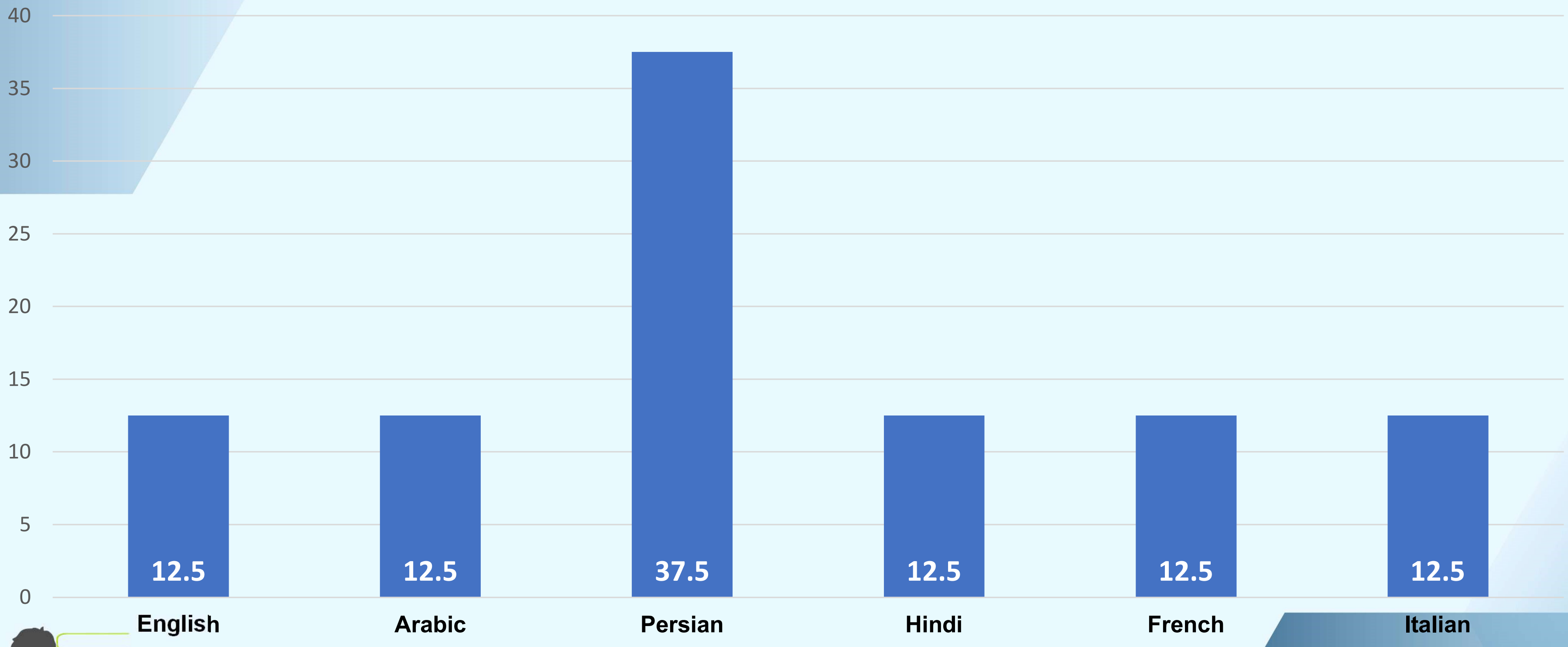
Day	Date	Average Daily Throughput
1	11/25/2020	0.02
2	11/26/2020	0.02
3	11/27/2020	0.01
4	11/28/2020	0.06
5	11/29/2020	0.05
6	11/30/2020	0.05

Throughput Analysis

Queries:

```
SELECT
              row_number() over(order by ds) AS Day,
  ds AS Date,
  ROUND(COUNT(event) / SUM(time_spent),2) AS 'Avg Daily Throughout’
FROM
   job_data
GROUP BY ds
ORDER BY ds;
```

Language Share Analysis



Language Share Analysis

Queries:

```
SELECT
    language, sub.total, (COUNT(*) * 100.0 / total) AS percentage_of_use
FROM
   job_data
CROSS JOIN
   (SELECT
       COUNT(*) AS total
   FROM
      job_data) AS sub
GROUP BY language , total;
```

Duplicate Rows Detection

Actor Id	Duplicate No
1003 →	2

Duplicate Rows Detection

Queries:

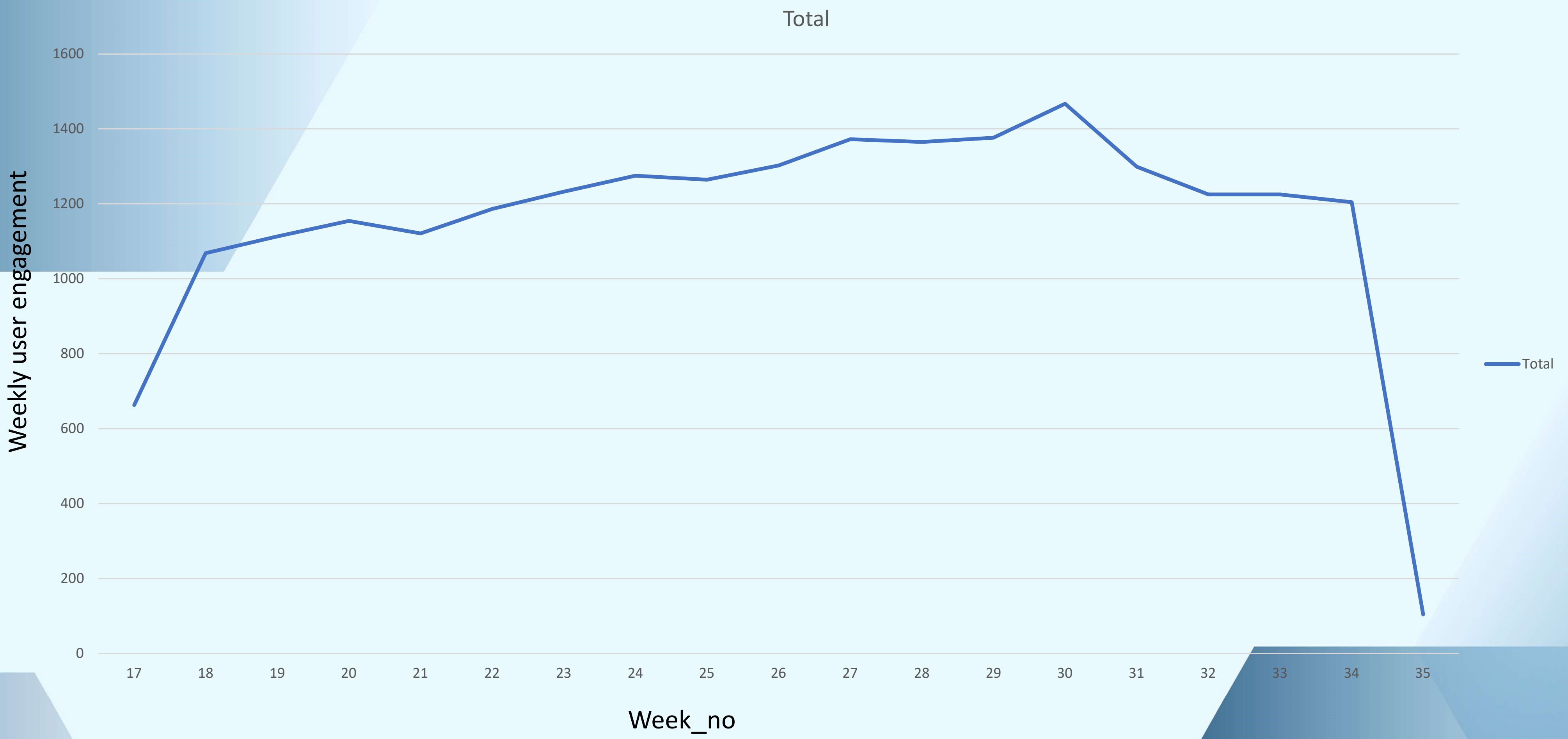
```
SELECT
    actor_id, COUNT(*) AS duplicates
FROM    job_data
GROUP BY actor_id
HAVING COUNT(*) > 1;
```

Case Study 2

Investigating Metric Spike

--- Insights ---

Weekly User Engagement



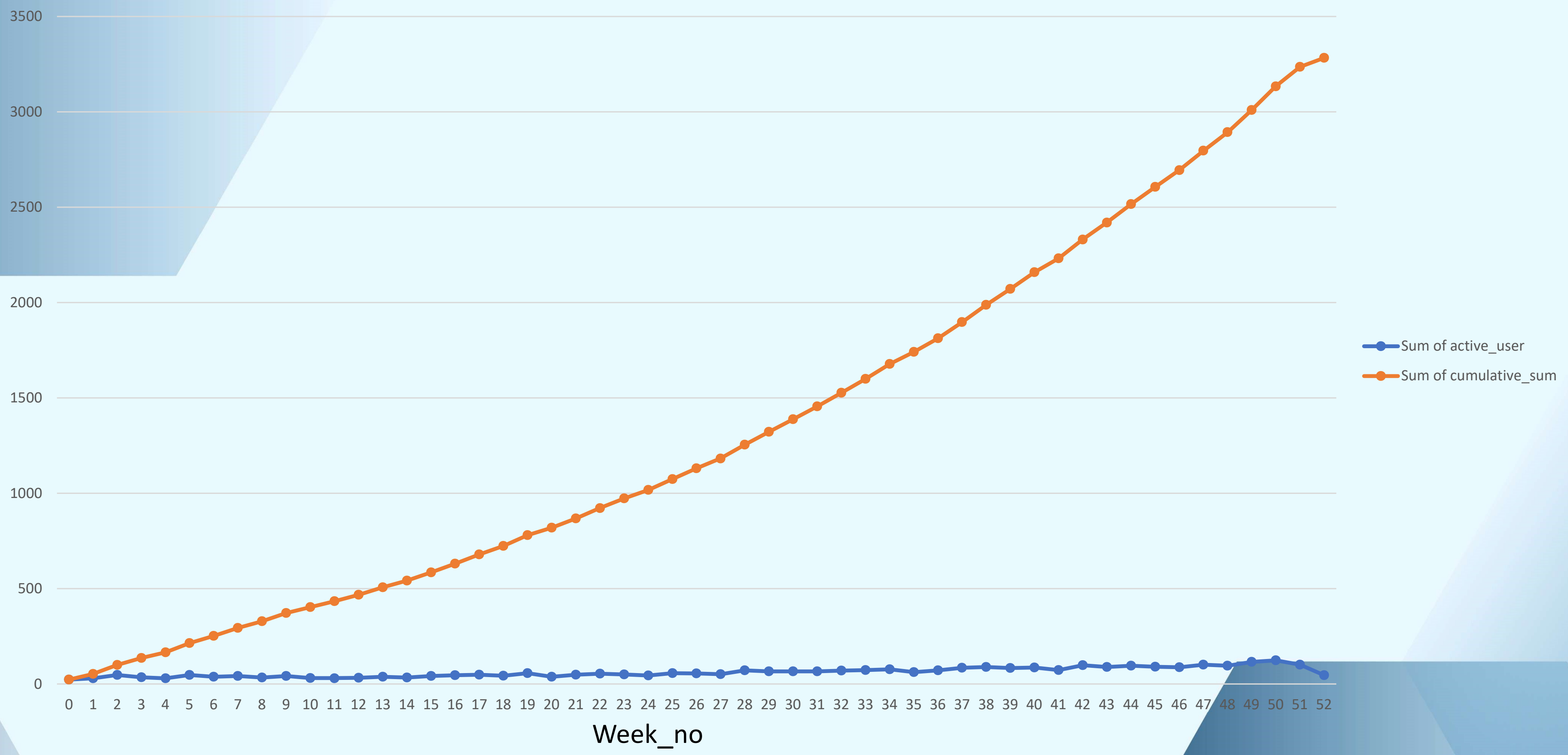
Weekly User Engagement

Queries:

```
Select
extract(week from occurred_at) as week_no,
 count(distinct user_id) as weekly_user_engagement
from events
group by week_no;
```

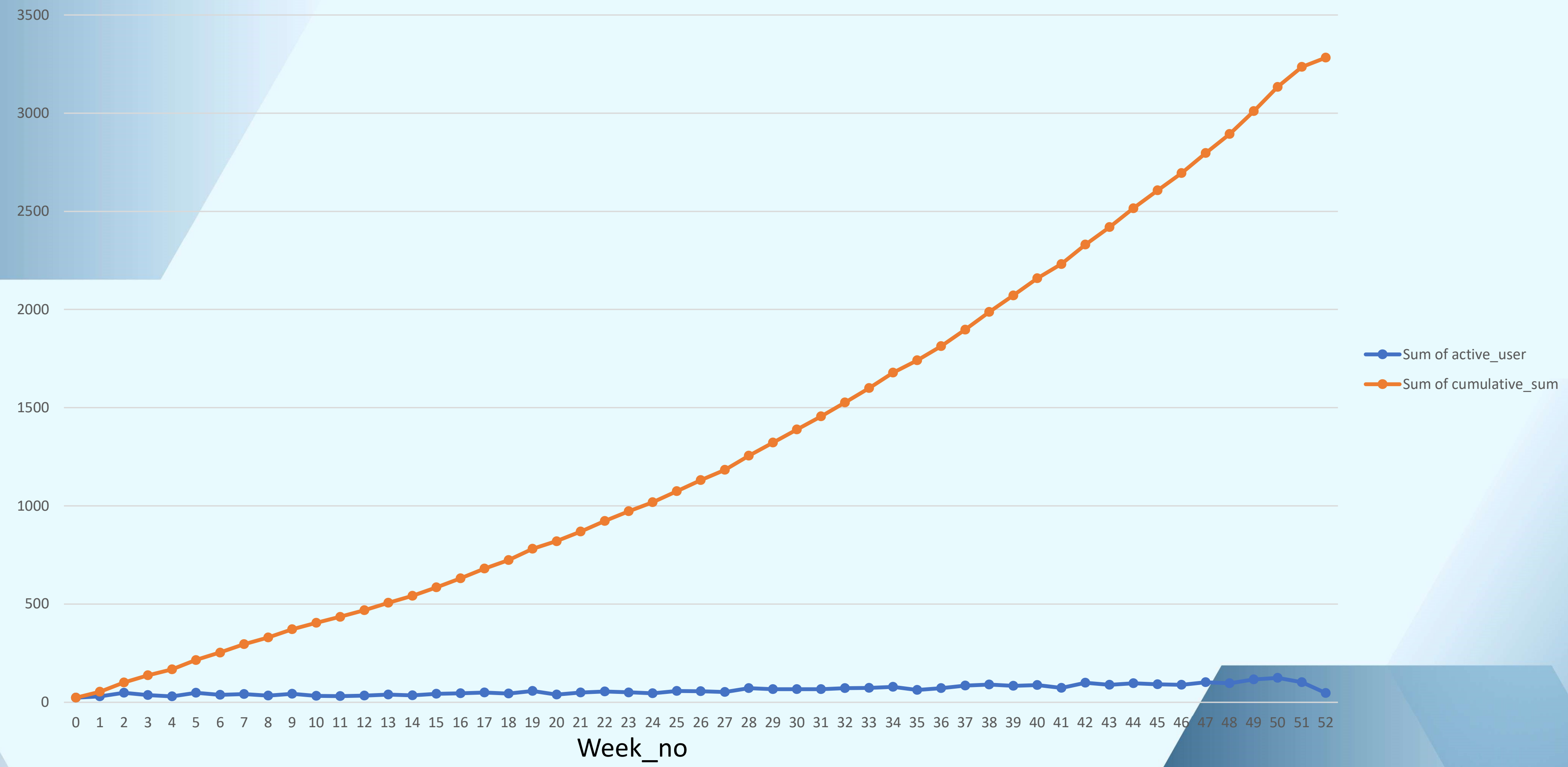
Year: 2013

User Growth Analysis



Year: 2014

User Growth Analysis

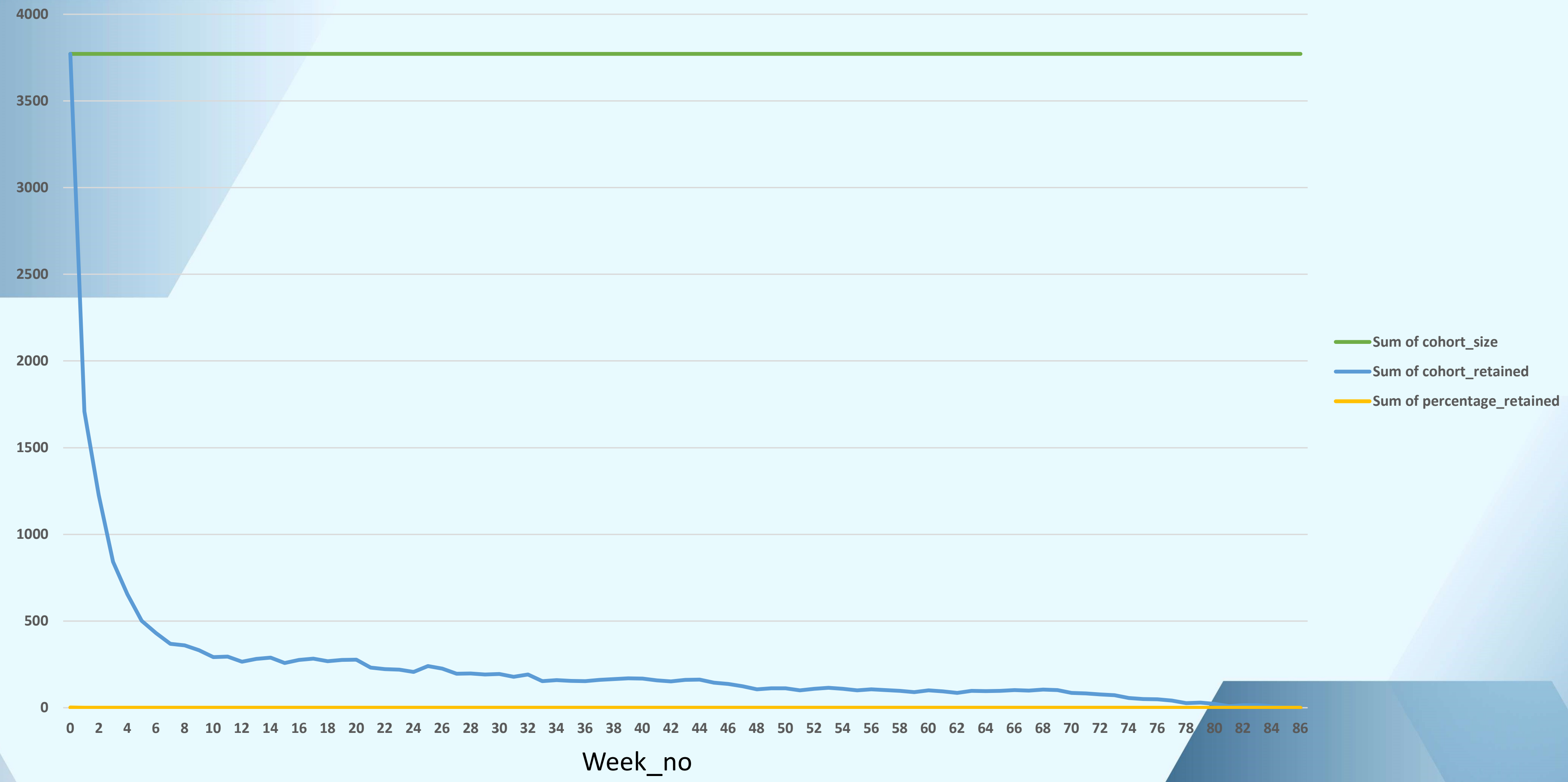


User Growth Analysis

Queries:

```
select
week_no,Years,active_user,sum(active_user) over(order by Years, week_no) as
cumulative_sum
From
(select extract(week from activated_at) as week_no,
extract(year from activated_at) as Years,
count(distinct user_id) as active_user
from users
where state= 'active’
group by Years,week_no
order by Years,week_no)a;
```

Weekly Retention Analysis

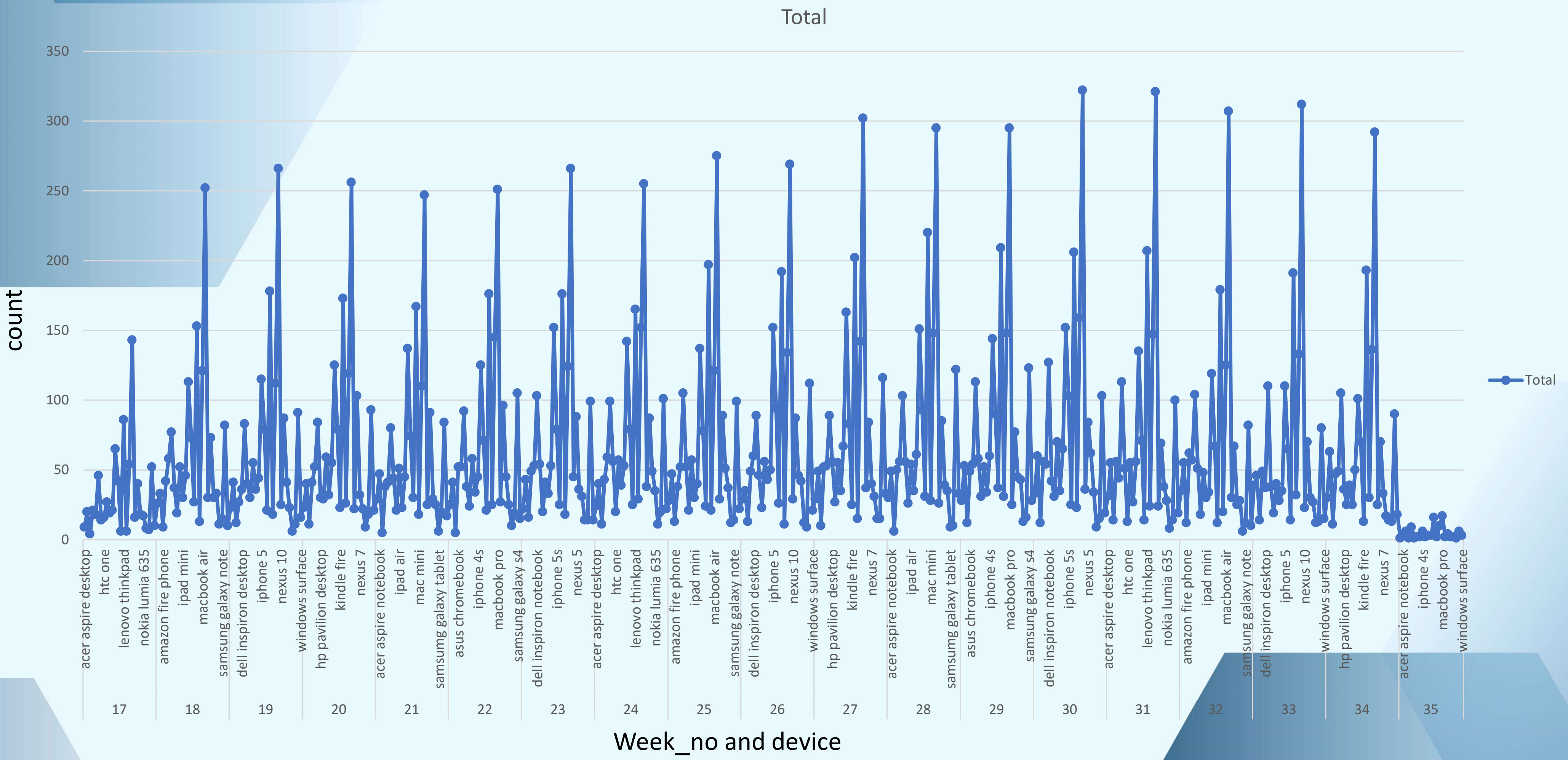


Weekly Retention Analysis

Queries:

```
Select
week_period,first_value(cohort_retained) over (order by week_period) as cohort_size,
cohort_retained,cohort_retained / first_value(cohort_retained) over (order by week_period)
as percentage_retained From
(select
timestampdiff(week,a.activated_at,b.occurred_at) as week_period,
count(distinct a.user_id) as cohort_retained
From
(select user_id, activated_at
from users
where state='active’) a
inner join
(select user_id,occurred_at from events )b
On
   a.user_id=b.user_id
    group by 1) c;
```

Weekly Engagement Per Device



Weekly Engagement Per Device

Queries:

```
SELECT
     EXTRACT(WEEK FROM occurred_at) AS week,
    EXTRACT(YEAR FROM occurred_at) AS year,
    device,
    COUNT(DISTINCT user_id) AS count
FROM
    events
WHERE
    event_type = 'engagement’
GROUP BY week, year,3
ORDER BY week , year,3;
```

Email Engagement Analysis

week	num_users	time_weekly_digest_sent	time_weekly_digest_sent_growth	time_email_open	time_email_open_growth	time_email_clickthrough	time_email_clickthrough_growth
17	981	908	0	310	0	166	0
18	2714	2602	1694	912	602	430	264
19	2787	2665	63	972	60	477	47
20	2874	2733	68	1004	32	507	30
21	2926	2822	89	1014	10	443	-64
22	3029	2911	89	987	-27	488	45
23	3134	3003	92	1075	88	538	50
24	3254	3105	102	1155	80	554	16
25	3343	3207	102	1096	-59	530	-24
26	3439	3302	95	1165	69	556	26
27	3543	3399	97	1228	63	621	65
28	3641	3499	100	1250	22	599	-22
29	3734	3592	93	1219	-31	590	-9
30	3866	3706	114	1383	164	630	40
31	3950	3793	87	1351	-32	445	-185
32	4023	3897	104	1337	-14	418	-27
33	4200	4012	115	1432	95	490	72
34	4294	4111	99	1528	96	490	0
35	48	0	-4111	41	-1487	38	-452

Email Engagement Analysis

Queries:

```
Select
week,num_users,time_weekly_digest_sent,
time_weekly_digest_sent-lag(time_weekly_digest_sent) over(order by week) as
time_weekly_digest_sent_growth,
time_email_open,time_email_open-lag(time_email_open) over(order by week) as
time_email_open_growth,
time_email_clickthrough,time_email_clickthrough-lag(time_email_clickthrough) over(order
by week) as time_email_clickthrough_growth
From(select week(occurred_at)as week,
count(distinct user_id) as num_users,
sum(if(action='sent_weekly_digest',1,0)) as time_weekly_digest_sent,
sum(if(action='email_open',1,0)) as time_email_open,
sum(if(action='email_clickthrough',1,0)) as time_email_clickthrough from email_events
 group by 1
order by 1) a;
```

Result

Really engaging project, difficulty of the project makes it more fulfilling to execute.

Learnt a lot of new things like rolling average, cohort retention analysis.

Tired to insert excel charts wherever I could, hopefully would be able to use excel more efficiently next time.

Became better in using windows function.

Thank You