Operation Analytics and Investigating Metric Spike

Report By – Siva Sankari H

1.Introduction

Operational Analytics is a business analytics technique that uses data to improve a company's day to day operations. It is also called as operational intelligence or real time business visibility. It uses both data analytics and business intelligence to improve efficiency and streamline everyday operations in real time Unlike traditional methods that rely on quarterly or annual data, it enables businesses to identify and address issues in real time, enhancing efficiency, reducing waste, and increasing profitability.

Investigating metric spike is also an important part of operational analysis – used to root cause analysis. As a data analyst our responsibility lies in analysing the historical and real time data to spot trends, sudden spikes or anomalies in key metrics that could indicate issues in operations. Creating reports to provide clear, actionable insights to stakeholders, suggesting improvements or adjustments to optimise processes based on findings and monitoring the operational performance also falls under the responsibility of data analyst.

1.1 Project Description

In this project we will be working as a Lead Data Analyst at a company like Microsoft. We will be provided with various datasets and tables, and our task will be to derive insights from this data to answer questions posed by different departments within the company. The goal is to use your advanced SQL skills to analyse the data and provide valuable insights that can help improve the company's operations and understand sudden changes in key metrics.

2. Database Design

The project is divided into two case studies – Job Data Analysis and Investigating Metric Spikes. We will be using 4 tables in this project – job_data, users, events, and email_events. Create a new schema called **project3**. All the necessary tables are created within this DB.

3. Case Study – 1: Job Data Analysis

We will be working on table named **job_data** table with the following columns:

job_id	Unique identifier of jobs
actor_id	Unique identifier of actor
event	The type of event (decision/skip/transfer).
language	The Language of the content
time_spent	Time spent to review the job in seconds.
org	The Organization of the actor
ds	The date in the format YYYY/mm/dd (stored as text).

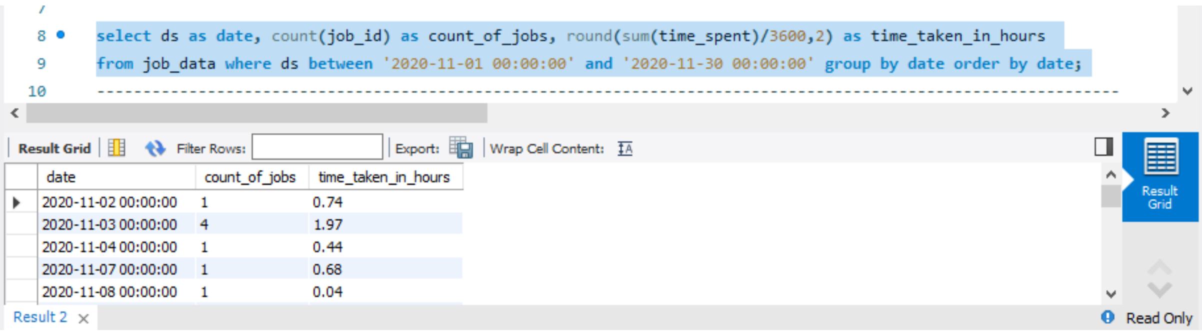
3.1 Approach and Analysis and Insights Derived

Task – A: Jobs Reviewed Over Time

The main objective of the task is to calculate the number of jobs reviewed per hour for each day in November 2020.

Query and Output –

```
SELECT
        DS AS DATE,
        COUNT(JOB_ID) AS COUNT_OF_JOBS,
        ROUND(SUM(TIME_SPENT)/3600,2) AS TIME_TAKEN_IN_HOURS
FROM
        JOB_DATA
WHERE
        DS BETWEEN '2020-11-01 00:00:00' AND '2020-11-30 00:00:00'
GROUP BY
        DATE
ORDER BY
        DATE;
```



Output 1a): Jobs Reviewed over time

SQL processes queries in the order: FROM, JOIN, WHERE, GROUP BY, HAVING, SELECT, DISTINCT, ORDER BY, and finally, LIMIT/OFFSET.

The output obtained shows – total number of jobs occurred in a day and the time taken to complete (– converted to hours) along with the target date. We have filtered the data only for November ,2020 as mentioned in the question. We can see the output starts from 3rd November, 2020. 4 events have occurred on that day and it took around 1.97 hours to complete (i.e. 2129+3274+287+1400 = 7090 secs).

```
#task 1 = Calculate the number of jobs reviewed per hour for each day in November 2020.
select * from job_data where ds = '2020-11-03 00:00:00'; # on nov 1st, we reviewed 3 jobs
```

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

job_id	actor_id	event	language	time_spent	org	ds
9	1951	skip	French	2129	A	2020-11-03 00:00:00
44	1237	transfer	French	3274	C	2020-11-03 00:00:00
10	1406	decision	English	287	D	2020-11-03 00:00:00
16	1420	decision	French	1400	E	2020-11-03 00:00:00

Output 21b): Jobs Reviewed on Nov 3rd,2020.

Task – B: Throughput Analysis

The task is to calculate the average number of events that occur per second over a period of seven days. The metric used to measure this is called "*throughput*." The question asks to determine the 7-day rolling average of throughput. Additionally, it asks to compare the preference between daily metrics and 7-day rolling and explain why.

Query and Output –

```
SELECT
        DS AS DATE,
        COUNT(EVENT) AS EVENT_PER_DAY,
        ROUND(SUM(TIME_SPENT)/86400,2) AS TIME_SPENT_DAY_HR ,
        ROUND(COUNT(EVENT)/(SUM(TIME_SPENT)/86400),3) AS THROUGHPUT_PER_DAY
FROM JOB_DATA
WHERE
        DS BETWEEN '2020-11-01 00:00:00' AND '2020-11-30 00:00:00'
GROUP BY
        DATE
ORDER BY
       DATE;
```

```
select ds as date, count(event) as event_per_day, round(sum(time_spent)/86400,2) as time_spent_day_hr ,
round(count(event)/(sum(time_spent)/86400),3) as throughput_per_day
from job_data
where ds between '2020-11-01 00:00:00' and '2020-11-30 00:00:00' group by date order by date;
```

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

date	event_per_day	time_spent_day_hr	throughput_per_day
2020-11-02 00:00:00	1	0.03	32.481
2020-11-03 00:00:00	4	0.08	48.745
2020-11-04 00:00:00	1	0.02	54.511
2020-11-07 00:00:00	1	0.03	35.323
2020-11-08 00:00:00	1	0.00	546.836

Output 3: Throughput Analysis

The output shows the calculated throughput for each day – converted to hours. For simplifying the calculations, we have rounded off the results to 3 decimal places.

On Nov 3rd, 2020 – 4 events have occurred which collectively took 0.8 hours to complete. The throughput for that day is calculated as

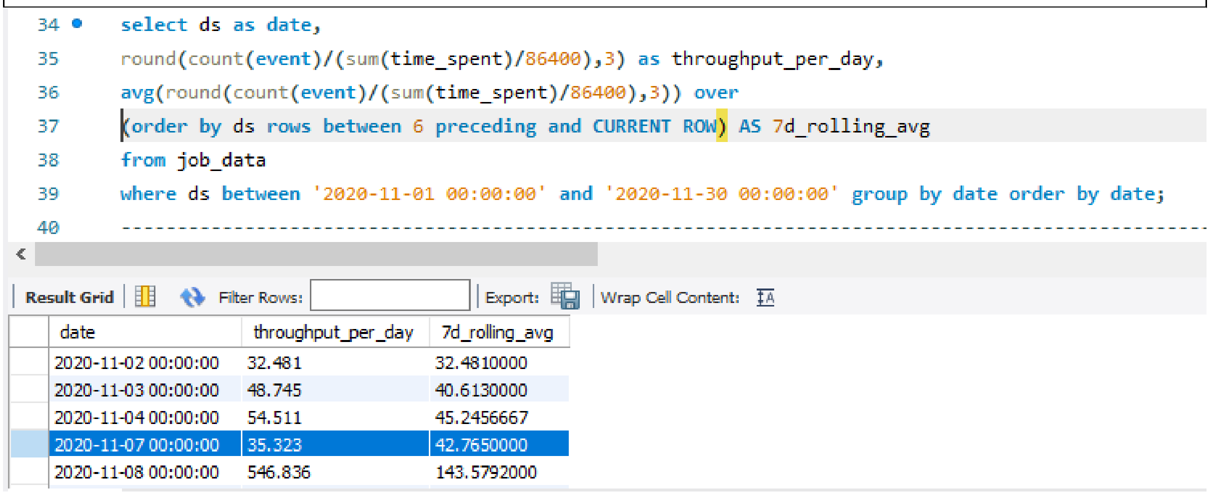
$$Throughput = \frac{Total\ no.of\ events\ occured\ in\ a\ day}{Time\ spent\ on\ completing\ it\ in\ hour}$$

We are dividing the resultant by 24 hours to calculate throughput per day. So, on Nov 3rd, 2020 we have got a throughput of 48.745.

<u>7-day rolling average –</u>

Query and Output –

```
SELECT
    DS AS DATE,
    ROUND(COUNT(EVENT)/(SUM(TIME_SPENT)/86400),3) AS THROUGHPUT_PER_DAY,
    AVG(ROUND(COUNT(EVENT)/(SUM(TIME_SPENT)/86400),3))
    OVER
    (ORDER BY DS ROWS BETWEEN 6 PRECEDING AND CURRENT ROW)
    AS 7D_ROLLING_AVG
FROM
    JOB_DATA
WHERE
    DS BETWEEN '2020-11-01 00:00:00' AND '2020-11-30 00:00:00'
GROUP BY
    DATE
ORDER BY
    DATE;
```



```
select ds as date,
round(count(event)/(sum(time_spent)/86400),3) as throughput_per_day,
avg(round(count(event)/(sum(time_spent)/86400),3)) over
(order by ds rows between 6 preceding and CURRENT ROW) AS 7d_rolling_avg
from job_data
where ds between '2020-11-01 00:00:00' and '2020-11-30 00:00:00' group by date order by date;
```

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

date	throughput_per_day	7d_rolling_avg
2020-11-02 00:00:00	32.481	32.4810000
2020-11-03 00:00:00	48.745	40.6130000
2020-11-04 00:00:00	54.511	45.2456667
2020-11-07 00:00:00	35.323	42.7650000
2020-11-08 00:00:00	546.836	143.5792000

Output 4: 7 Day Rolling Average Throughput

Rolling Averages sometimes called as *moving average* is a metric used to calculate trends over a short period of time. This technique is mostly employed when the data points move up or down more drastically. This means the spike in the data is unpredictable. Calculating averages for a short period of time and rolling it over for a new period for few turns allows the analysts to understand the trends more accurately. There are 5 missing dates in our dataset. We have ignored those dates. The query in the table above shows the 7-day rolling average calculations.

The figure below shows the pivot chart and table. we have started our calculations from day 7 of our dataset (i.e., 10/11/2020). From the chart it can be seen that, there is down trend on Nov 16th, 2020 – 24th Nov, 2020. There is up trend from 27th Nov, 2020.

Row Labels	Sum of throughput_per_day	Sum of 7day average
02-11-2020 00:00	32.481	
03-11-2020 00:00	48.745	
04-11-2020 00:00	54.511	
07-11-2020 00:00	35.323	
08-11-2020 00:00	546.836	
09-11-2020 00:00	333.591	
10-11-2020 00:00	70.244	160.2472857
11-11-2020 00:00	75.923	166.4532857
12-11-2020 00:00	48.41	166.4054286
13-11-2020 00:00	47.025	165.336
14-11-2020 00:00	56.73	168.3941429
15-11-2020 00:00	36.166	95.44128571
16-11-2020 00:00	71.405	57.98614286
19-11-2020 00:00	78.108	59.10957143
20-11-2020 00:00	59.848	56.81314286
21-11-2020 00:00	40.336	55.65971429
22-11-2020 00:00	145.823	69.77371429
23-11-2020 00:00	26.174	65.40857143
24-11-2020 00:00	49.343	67.291
25-11-2020 00:00	1920.001	331.3761429
26-11-2020 00:00	105.366	335.2701429
27-11-2020 00:00	40.32	332.4804286
28-11-2020 00:00	5236.37	1074.771
29-11-2020 00:00	85.913	1066.212429
30-11-2020 00:00	146.11	1083.346143
Grand Total	**9391.102**	**5577.775571**

Table 1Pivot Table showing daily throughput metrics and 7-day average rollout

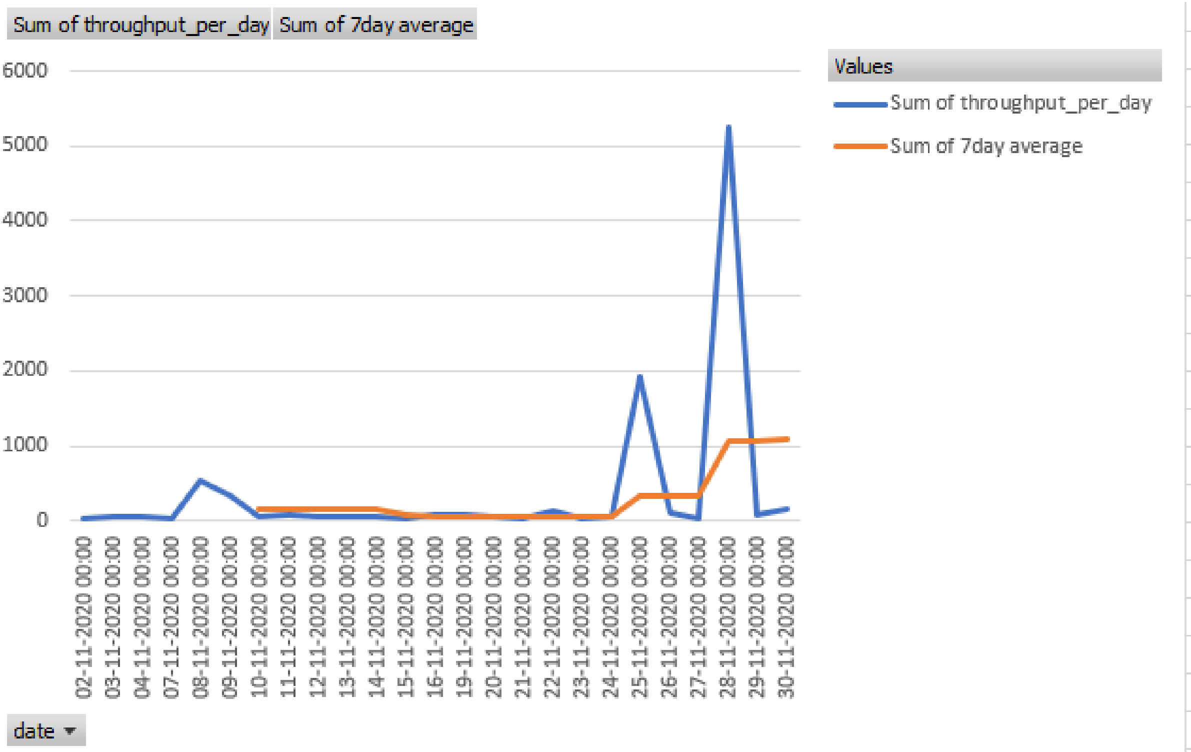


Table 2: Pivot chart representation for average throughput rollout

Task – C: Language Share Analysis

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

Query and Output –

```
SELECT
    LANGUAGE,
    ROUND ((COUNT (*)/56) *100,2) AS PERCENTAGE_SHARE
FROM JOB_DATA
WHERE
    DS BETWEEN '2020-11-01 00:00:00' AND '2020-11-30 00:00:00'
GROUP BY LANGUAGE;
```

```
select language, round((count(*)/56)*100,2) as percentage_share from job_data
where ds between '2020-11-01 00:00:00' and '2020-11-30 00:00:00' group by language;
```

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

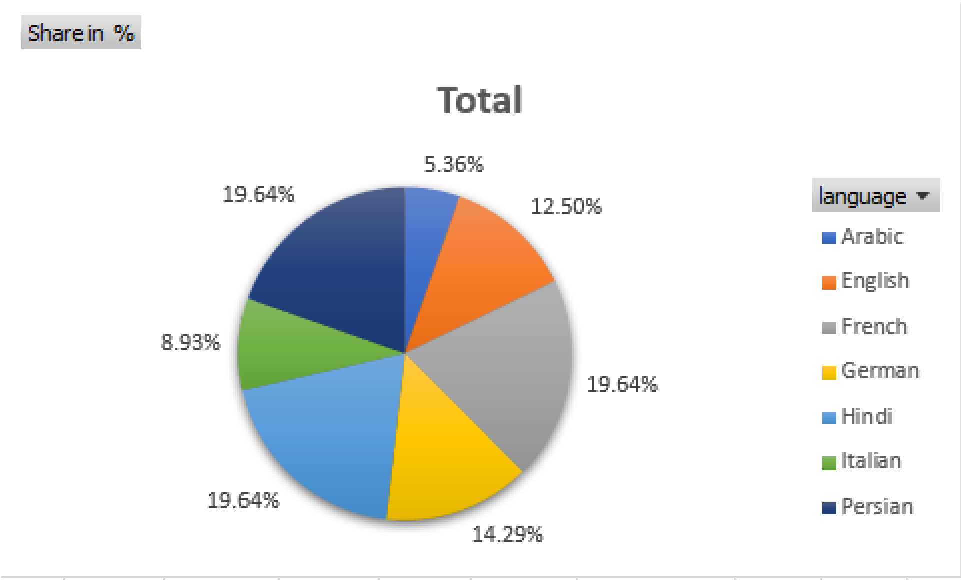
language	percentage_share
Italian	8.93
English	12.50
French	19.64
Hindi	19.64
German	14.29
Persian	19.64
Arabic	5.36

Output 5: Language Share Analysis

The query above groups all the languages existing in the table and calculates their percentage share.

We have total 7 distinct languages in our dataset - Italian, French, English, Hindi, German, Persian, Arabic. We have 56 entries for November 2020. Percentage language is calculated as follows.

$$Percentage\ share = \frac{Count(distinct\ Language)}{Total\ Count} * 100$$



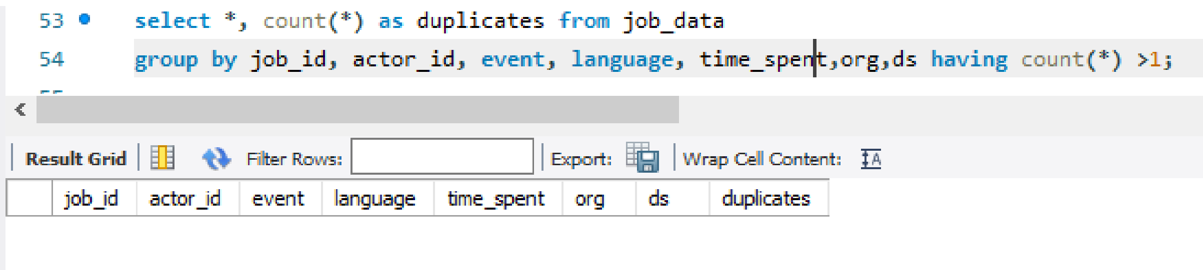
Output 6: Language Share

Task – D: Duplicate Rows Detection

Objective is to identify duplicate row if any in our dataset.

Query and Output –

```
SELECT *, COUNT (*) AS DUPLICATES
FROM JOB_DATA
GROUP BY
        JOB_ID, ACTOR_ID, EVENT, LANGUAGE, TIME_SPENT, ORG, DS
HAVING COUNT (*) >1;
```



Output 7: Duplicate rows detection

There are no duplicate rows in our dataset.

4. Case Study – 2: Investigating Metric Spike

4.1 Table Info -

We will be working with 3 tables in the project3 schema. Sample tables is given. Table details follow as below –

Table – 1: users: Contains one row per user, with descriptive information about that user's account.

user_id - PK	Unique id given for each user
company_id	Id given from the company like Microsoft to user.
language	Language Preference of the user
activated_at	Date of account activation
state	Status of the account
created_at	Date of account creation

Table – 2: events: Contains one row per event, where an event is an action that a user has taken (e.g., login, messaging, search).

User_id	Unique id given for each user
Event_type	Type of event a user engaged in - engagement and signup flow
Event_name	Name of the event
Location	Location where the device is present
Device	Type of device being used
User_type - PK	-
Occurred_at	Time of event occurance

Table – 3: email_events: Contains events specific to the sending of emails.

User_id	Unique id given for each user
Action	Action taken for the event - sent_weekly_digest, email_open, email_clickthrough, sent_reengagement_email
User_type	-
Occurred_at	Time of action

4.2 Approach and Analysis and Insights Derived

Task – A: Weekly User Engagement

Objective – Measure the activeness of users on a weekly basis

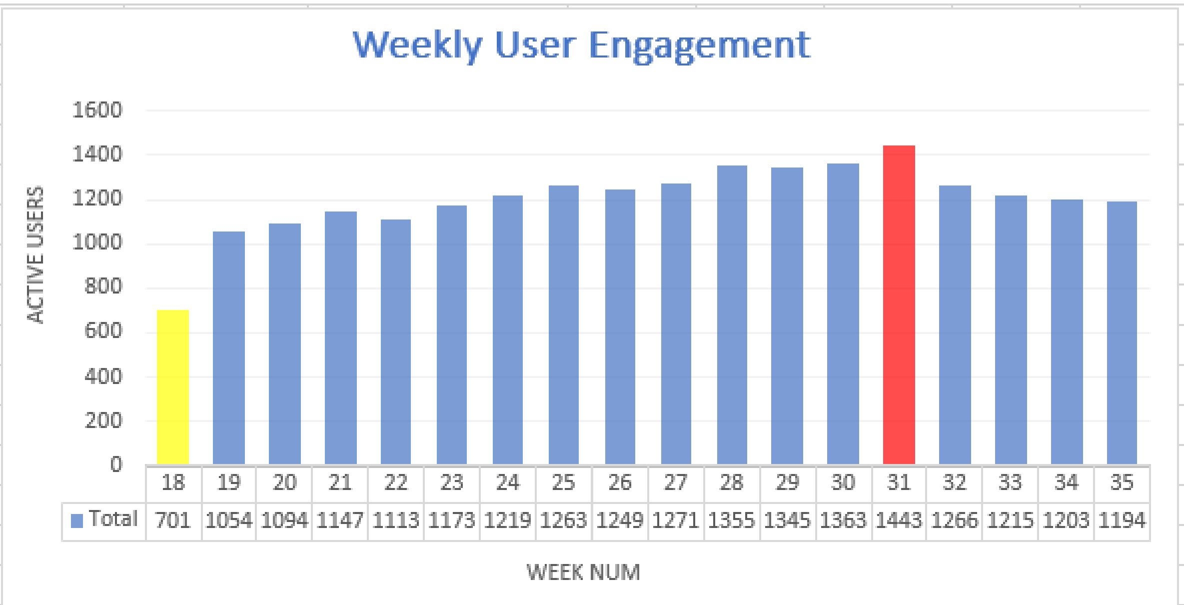
We are going use the events table for this task. An active user will have event_type = 'engagement' and event_name being anything but "complete_signup". An inactive user will have event_name = "complete_signup" and event_type = "signup_flow".

"Engagement" and "signup flow" indicates two different stages of user activity. *"Signup flow"* indicates a user is still in his login page or registration stage. This is considered as *inactive status*. *"Engagement"* state indicates a user has logged in successfully and is in *active status*. The state at which the user currently is depends on his activities.

We have used "WEEKOFYEAR()" function that returns the week number in the year assuming Monday to be the start of the week and the year under consideration has more than 3 days in 1st week. The dataset contains entries from May 1st, 2014 – the year 2014 has more than 3 days in its 1st week.

Query and Output –

```
SELECT
  WEEKOFYEAR(OCCURED_AT) AS WEEK_NUM,
  COUNT(DISTINCT USER_ID) AS ACTIVE_USERS
FROM
  EVENTS WHERE EVENT_TYPE = 'ENGAGEMENT'
GROUP BY WEEK_NUM ORDER BY WEEK_NUM;
```



	18	19	20	21	22	23	24	25	26	27	28	29	30	31	32	33	34	35
Total	701	1054	1094	1147	1113	1173	1219	1263	1249	1271	1355	1345	1363	1443	1266	1215	1203	1194

Excel Chart 1: Weekly Active Users Engagement

```
SELECT
  weekofyear(occured_at) AS week_num,
  COUNT(DISTINCT user_id) AS Active_users
FROM
  events WHERE event_type = 'engagement'
GROUP BY week_num ORDER BY week_num;
```

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

week_num	Active_users
18	701
19	1054
20	1094
21	1147
22	1113
23	1173
24	1219
25	1263
26	1249
27	1271
28	1355
29	1345
30	1363
31	1443
32	1266
33	1215
34	1203
35	1194

Output 1: Investigating Weekly User Engagement

Insights – Maximum user activity can be seen on week 31 with a rate of 1443. Minimum being 701 that occurred on week 18. It can be seen that we have a average of 1203 users active.

Task – B: User growth Analysis

Objective: Analyse the growth of users over time for a product. The plot below shows the users activity in each month – May to August.

Query and Output -

```
SELECT
        MONTH(OCCURED_AT) AS MONTH_NUMBER,
        DEVICE, COUNT (DISTINCT USER_ID) AS ACTIVE_USERS
FROM EVENTS
WHERE
         DEVICE IN
         ('DELL INSPIRON NOTEBOOK','IPHONE 5','IPHONE 4S','WINDOWS
SURFACE','MACBOOK AIR','IPHONE 5S','MACBOOK PRO','KINDLE FIRE','IPAD MINI','NEXUS
7','NEXUS 5','SAMSUNG GALAXY S4','LENOVO THINKPAD','SAMSUMG GALAXY TABLET',
```

```
'ACER ASPIRE NOTEBOOK','ASUS CHROMEBOOK','SAMSUNG GALAXY NOTE','MAC
MINI','HP PAVILION DESKTOP','IPAD AIR','HTC ONE','DELL INSPIRON DESKTOP','AMAZON
FIRE PHONE','ACER ASPIRE DESKTOP','NOKIA LUMIA 635','NEXUS 10')
GROUP BY
        MONTH_NUMBER, DEVICE
ORDER BY
        MONTH_NUMBER ASC;
```

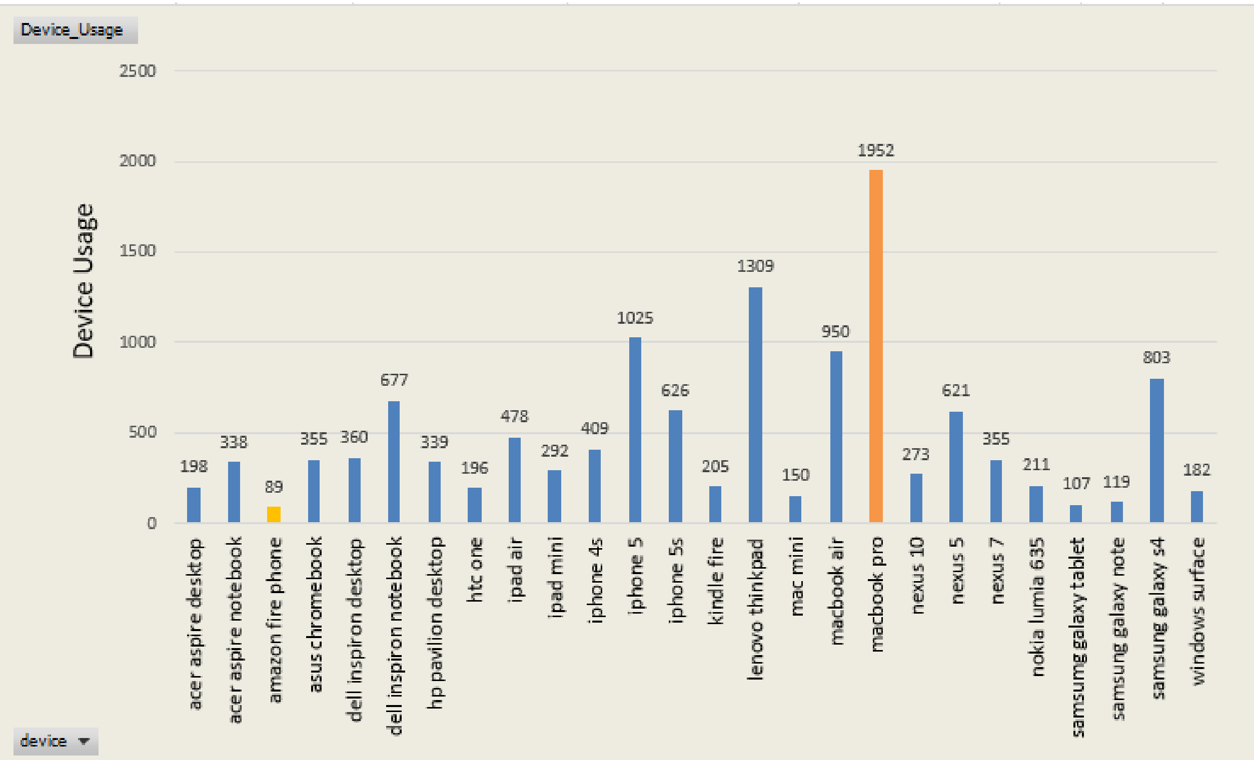
```
select month(occured_at) as Month_number, device, count(distinct user_id) as Active_users from events
where device in ('dell inspiron notebook','iphone 5','iphone 4s','windows surface','macbook air','iphone 5s',
group by Month_number, device order by Month_number asc;
```

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

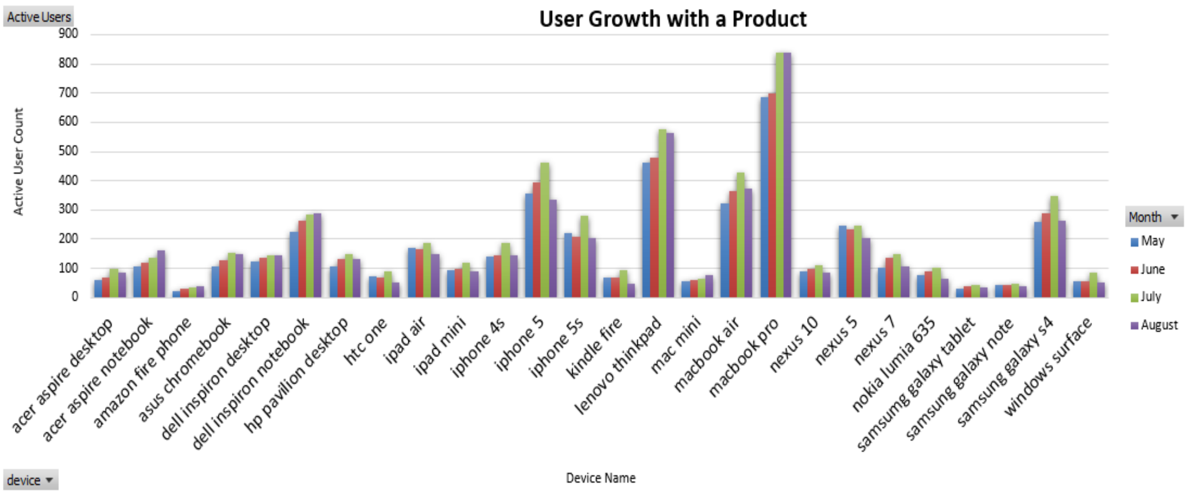
Month_number	device	Active_users
5	acer aspire desktop	61
5	acer aspire notebook	108
5	amazon fire phone	21
5	asus chromebook	107
5	dell inspiron desktop	122
5	dell inspiron notebook	225
5	hp pavilion desktop	108
5	htc one	75
5	ipad air	171

Output 2: User Growth with Product

We are grouping the data based on the device registered in the db. We have total 26 distinct devices registered. The above query selects the count of distinct users according to the device name. for the ease of visualization, we have bifurcated the results based on month. The figure below shows the results obtained along with the distribution chart.



Insights – It can be seen that “MacBook pro” (1952) device has been used most and “Amazon Fire Phone” (89) being the least.



Excel Chart 2: Device Growth Analysis

Active Users	Month				
Row Labels	**May**	**June**	**July**	**August**	**Grand Total**
acer aspire desktop	61	69	100	87	317
acer aspire notebook	108	118	137	160	523
amazon fire phone	21	31	33	38	123
asus chromebook	107	127	153	150	537
dell inspiron desktop	122	138	145	145	550
dell inspiron notebook	225	263	285	290	1063
hp pavilion desktop	108	132	148	131	519
htc one	75	67	88	50	280
ipad air	171	164	187	148	670
ipad mini	94	97	121	91	403
iphone 4s	142	143	187	143	615
iphone 5	358	393	460	336	1547
iphone 5s	221	210	278	204	913
kindle fire	68	70	92	48	278
lenovo thinkpad	461	480	576	562	2079
mac mini	54	59	63	76	252
macbook air	321	365	428	375	1489
macbook pro	688	700	839	837	3064
nexus 10	89	99	110	86	384
nexus 5	245	233	245	202	925
nexus 7	101	135	149	108	493
nokia lumia 635	79	88	101	65	333
samsumg galaxy tablet	31	37	43	33	144
samsung galaxy note	45	42	46	38	171
samsung galaxy s4	257	290	346	265	1158
windows surface	56	55	85	53	249
Grand Total	**4308**	**4605**	**5445**	**4721**	**19079**

Output 3: Month wise device engagement

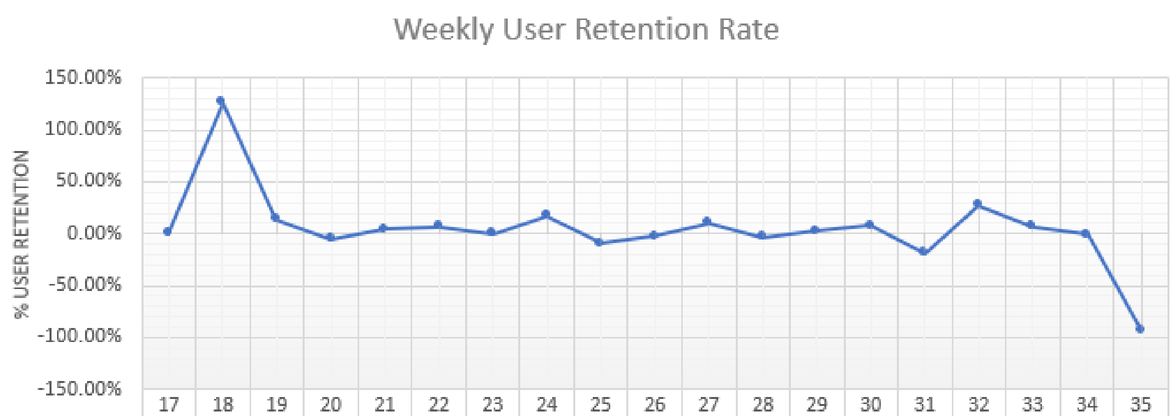
The pivot table shows the month wise device usage. Because of data granularity some count differences can be seen in the pivot table and the previous output. The user growth is maximum in July month. It can be seen that the user growth trend raises with an average rate of 4770.

Task – C: Weekly Retention Analysis

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

Query and Output –

```
SELECT
      WEEK (TABLE1.OCCURED_AT) AS WEEK_NUM,
      COUNT (DISTINCT TABLE1.USER_ID) AS ACTIVE_USER,
      COUNT (DISTINCT TABLE1.USER_ID) – LAG (COUNT (DISTINCT TABLE1.USER_ID), 1,
NULL) OVER (ORDER BY WEEK (TABLE1.OCCURED_AT)) AS RETAINTED_USERS
FROM
       (SELECT DISTINCT USER_ID, EVENT_TYPE, OCCURED_AT
        FROM EVENTS
        WHERE EVENT_TYPE = "SIGNUP_FLOW") AS TABLE1
LEFT JOIN EVENTS AS TABLE2 ON TABLE1.USER_ID = TABLE2.USER_ID
WHERE
        TABLE2.EVENT_TYPE = "ENGAGEMENT" AND
        (WEEK (TABLE1.OCCURED_AT) = WEEK (TABLE2.OCCURED_AT))
GROUP BY
         WEEK (TABLE1.OCCURED_AT);
```



	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31	32	33	34	35
Total	0.00%	126.3	13.50	-4.86	3.98%	7.10%	0.00%	16.84	-9.61	-2.90	10.45	-3.15	2.79%	7.69%	-18.9	26.94	6.53%	-0.77	-93.0

Insights gathered – Retention drops over time due to factors like loss of initial excitement, poor user experience, lack of ongoing value or personalization, infrequent updates, inadequate support, or stronger competition.

Task – D: Weekly Engagement Per Device

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

In this task, we have to identify the count of users engaging in our various products. We have extracted week number from the “occurred_at” field in the events table and grouping the data by device. The output obtained is exported to excel for visualization. The resultant charts are attached below.

Query and Output –

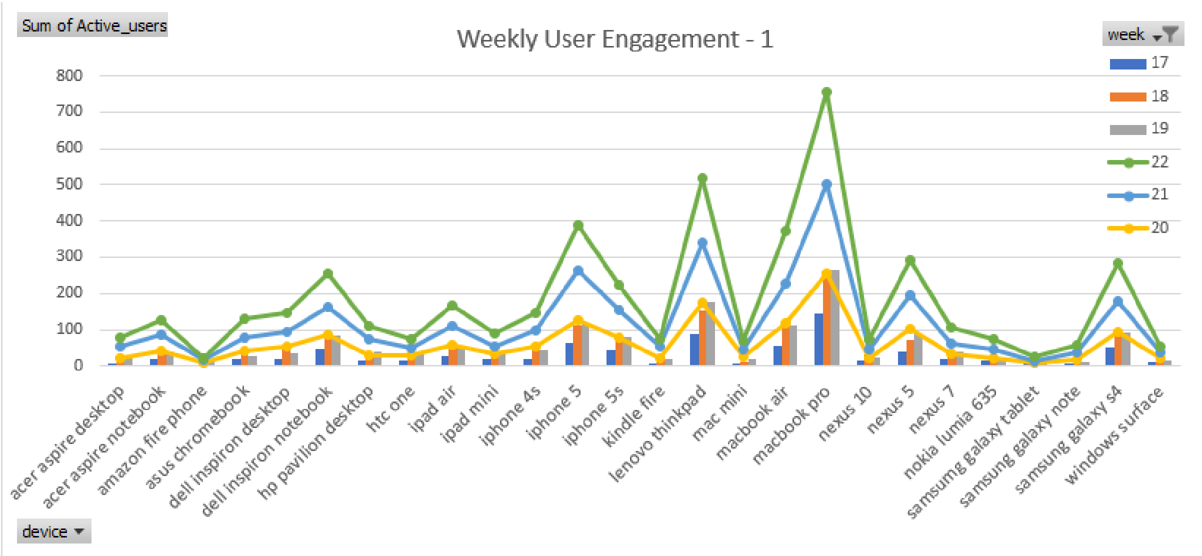
```
SELECT
      WEEK (OCCURED_AT) AS WEEK,
      DEVICE,
      COUNT (DISTINCT USER_ID) AS ACTIVE_USERS
FROM EVENTS
WHERE
      DEVICE IN
      ('DELL INSPIRON NOTEBOOK','IPHONE 5','IPHONE 4S','WINDOWS
SURFACE','MACBOOK AIR','IPHONE 5S', 'MACBOOK PRO','KINDLE FIRE','IPAD MINI','NEXUS
7','NEXUS 5','SAMSUNG GALAXY S4','LENOVO THINKPAD','SAMSUMG GALAXY
TABLET','ACER ASPIRE NOTEBOOK','ASUS CHROMEBOOK','SAMSUNG GALAXY NOTE','MAC
MINI','HP PAVILION DESKTOP','IPAD AIR','HTC ONE','DELL INSPIRON DESKTOP','AMAZON
FIRE PHONE','ACER ASPIRE DESKTOP','NOKIA LUMIA 635','NEXUS 10')
GROUP BY
       WEEK, DEVICE
ORDER BY WEEK ASC;
```

We have split the data into 4 charts for the ease of understanding.

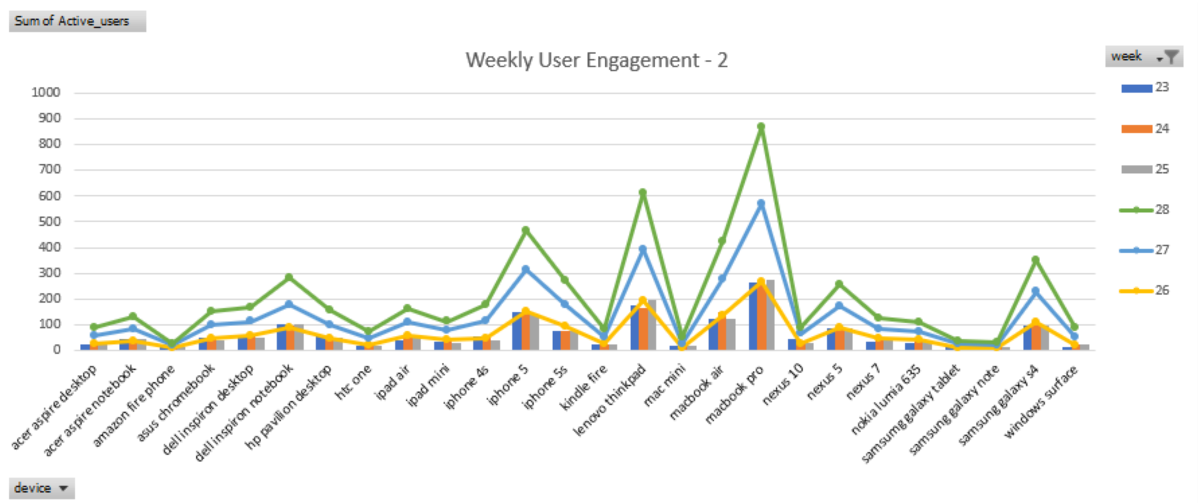
Chart – 1: week 17 to week 22 - Least used device in this week is “amazon fire phone” and most favored device was “macbook pro”. Iphone series, ipad air, mackbook series, dell series and Lenovo thinkpad are the most commonly used devices in this week.

Chart – 2: week 23 to week 28 - Least used device in this week is “amazon fire phone” and most favored device was “macbook pro”.

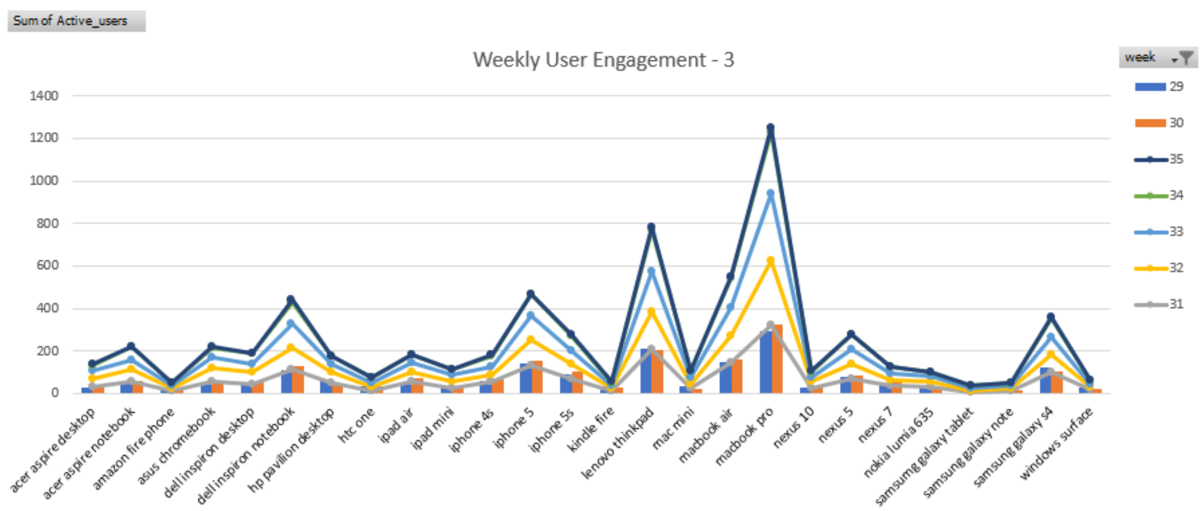
Chart – 3: Week 29 to week 35 - Least used device is “amazon fire phone” and “Samsung galaxy note” and most favored device are “macbook pro” and “Lenovo thinkpad”.



Sum of Active_users
Weekly User Engagement - 1
week
17
18
19
22
21
20
800
700
600
500
400
300
200
100
0
acer aspire desktop
acer aspire notebook
amazon fire phone
asus chromebook
dell inspiron desktop
dell inspiron notebook
hp pavilion desktop
htc one
ipad air
ipad mini
iphone 4s
iphone 5
iphone 5s
kindle fire
lenovo thinkpad
mac mini
macbook air
macbook pro
nexus 10
nexus 5
nexus 7
nokia lumia 635
samsumg galaxy tablet
samsung galaxy note
samsung galaxy s4
windows surface
device



Sum of Active_users
Weekly User Engagement - 2
week
23
24
25
28
27
26
1000
900
800
700
600
500
400
300
200
100
0
acer aspire desktop
acer aspire notebook
amazon fire phone
asus chromebook
dell inspiron desktop
dell inspiron notebook
hp pavilion desktop
htc one
ipad air
ipad mini
iphone 4s
iphone 5
iphone 5s
kindle fire
lenovo thinkpad
mac mini
macbook air
macbook pro
nexus 10
nexus 5
nexus 7
nokia lumia 635
samsumg galaxy tablet
samsung galaxy note
samsung galaxy s4
windows surface
device



Sum of Active_users
Weekly User Engagement - 3
week
29
30
35
34
33
32
31
1400
1200
1000
800
600
400
200
0
acer aspire desktop
acer aspire notebook
amazon fire phone
asus chromebook
dell inspiron desktop
dell inspiron notebook
hp pavilion desktop
htc one
ipad air
ipad mini
iphone 4s
iphone 5
iphone 5s
kindle fire
lenovo thinkpad
mac mini
macbook air
macbook pro
nexus 10
nexus 5
nexus 7
nokia lumia 635
samsumg galaxy tablet
samsung galaxy note
samsung galaxy s4
windows surface

Insights gathered –

1) On analysing the graphs above it can be seen that most the users favour MacBook series. It can be understood that people favour the MacBook Pro for its high performance, premium build quality, excellent Retina display, long battery life, seamless integration with other Apple devices, and macOS software. It's also portable, retains value well, and is known for its great keyboard, trackpad, and brand reputation.
2) The least favoured device is amazon fire phone. On further study of the product, we can come to conclusion that The Amazon Fire Phone was least favoured due to its limited app selection, poor software (Fire OS), weak hardware, lack of ecosystem integration, high price, uninspiring design, and the failure of its gimmicky 3D features.

Task – E: Email Engagement Analysis

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

The task is to analyse the email engagements. We can use ***"email_events"*** table for this.

Overall User Engagement via Emails –

We have a total of 6179 unique records. The below query shows the overall email engagements.

```
select action, count(distinct user_id) from email_events
where action in ("sent_weekly_digest","email_open","email_clickthrough","sent_reengagement_email")
group by action;
```

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

action	count(distinct user_id)
sent_reengagement_email	3653
sent_weekly_digest	4111
email_clickthrough	5277
email_open	5927

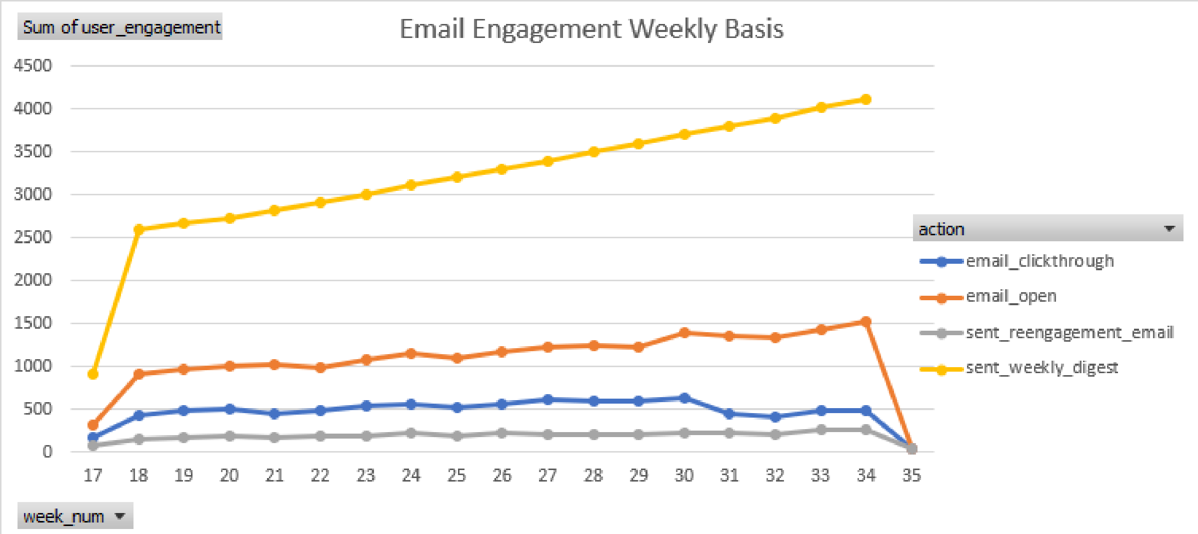
It can be seen that 5927 users out of total have opened the email and only 3653 users have responded back to it.

Weekly Email Engagement -

The below query shows the email engagement details weekly.

```
SELECT WEEK(OCCURED_AT) AS WEEK_NUM, ACTION, COUNT(USER_ID) AS USER_ENGAGEMENT FROM EMAIL_EVENTS GROUP BY WEEK_NUM, ACTION ORDER BY WEEK_NUM;
```

The output of the query is represented via pivot table and chart.



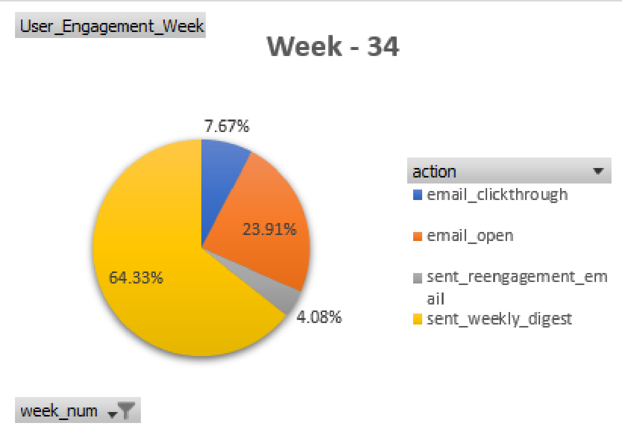
Sum of user_engagement				
	email_clickthrough	email_open	sent_reengagement_email	sent_weekly_digest
17	166	310	73	908
18	430	912	157	2602
19	477	972	173	2665
20	507	1004	191	2733
21	443	1014	164	2822
22	488	987	192	2911
23	538	1075	197	3003
24	554	1155	226	3105
25	530	1096	196	3207
26	556	1165	219	3302
27	621	1228	213	3399
28	599	1250	213	3499
29	590	1219	213	3592
30	630	1383	231	3706
31	445	1351	222	3793
32	418	1337	200	3897
33	490	1432	264	4012
34	490	1528	261	4111
35	38	41	48	

Table 3: Pivot Chart for Email Engagement Weekly

Insights from the chart –

1. Peak interaction can be seen from week 18. Most of the users are actively using the applications – this could be attributed to various factors like – It takes at least a week for the user to completely understand the platform and its interface. A reminder, notifications or updates about the new features or content during this initial period triggers re-engagement from the users
2. The pie chart below shows the user engagement in week 34 where there is peak engagement – “email_open” and “sent_weekly_digest”.

User_Engagement_Week	Column Labels	
Row Labels	34	Grand Total
email_clickthrough	7.67%	7.67%
email_open	23.91%	23.91%
sent_reengagement_email	4.08%	4.08%
sent_weekly_digest	64.33%	64.33%
Grand Total	100.00%	100.00%



3. On the whole, we have an average of 474 users/week engaging in "email_clickthrough" activity, 1076 users/week in "email_open" , 192 users/week in "sent_reengagement" and 3181 users/week in "sent_weekly_digest" activity.

User_Engagement_Week	Colum																		
Row Labels	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31	32	33	34	35	Average
email_clickthrough	166	430	477	507	443	488	538	554	530	556	621	599	590	630	445	418	490	490	38	474.210526
email_open	310	912	972	1004	1014	987	1075	1155	1096	1165	1228	1250	1219	1383	1351	1337	1432	1528	41	1076.78947
sent_reengagement_email	73	157	173	191	164	192	197	226	196	219	213	213	213	231	222	200	264	261	48	192.263158
sent_weekly_digest	908	2602	2665	2733	2822	2911	3003	3105	3207	3302	3399	3499	3592	3706	3793	3897	4012	4111		3181.5
Grand Total	1457	4101	4287	4435	4443	4578	4813	5040	5029	5242	5461	5561	5614	5950	5811	5852	6198	6390	127	

Table 4: Average User Engagement

5.Tech Stack Used

1) MySQL Workbench –
   a) We have used latest version v 8.0.40.
   b) MySQL Workbench is a powerful visual database design tool specifically designed for MySQL databases. It offers a user-friendly interface and a comprehensive suite of tools for database administration, development, and modelling.
   c) We have used for writing, execute and debug SQL queries. It's database design allows us to create and edit ER diagrams to visually model database structures.
2) SQL Server Management Studio –
   a) SQL SMS is used to interact with the SQL server database.
   b) The key features include – Query Editor – used for write, execute and debug queries, Database Engine – for connecting and managing the server instances. Data import and Export Wizard – for importing and exporting various sources like csv, excel and other databases.
3) MS Excel – used for visualisation. I have used pivot tables and pivot chart to analyse my data output and visualise it.