Operation Analytics and Investigating Metric Spike

Case Study 1: Job Data Analysis

Project description

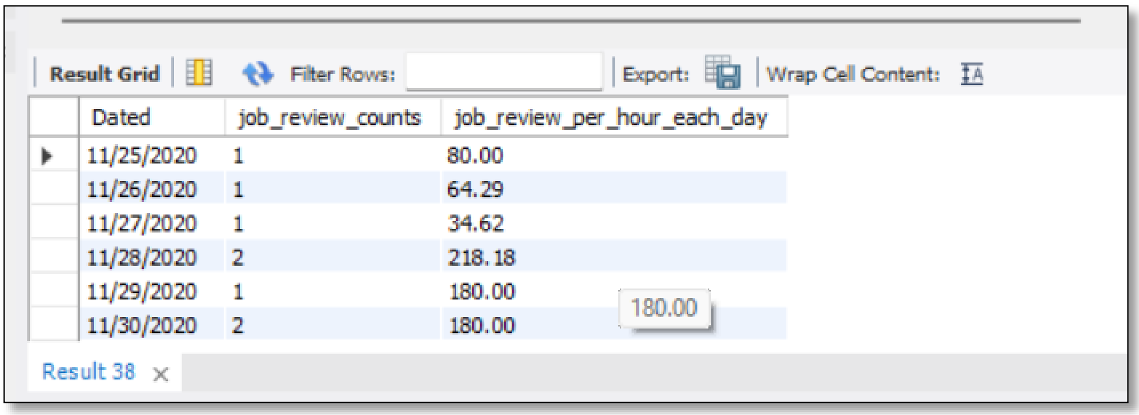
The dataset provided pertains to the review process of applicants for various job profiles. It encompasses data spanning from November 25th, 2020, to November 30th, 2020. Within this dataset, there are six unique job IDs (11, 20, 21, 22, 23, 25) and seven unique actor IDs ranging from 1001 to 1007. The events recorded include "skip," "transfer," and "decision." Additionally, reviews are conducted in six different languages: English, Arabic, Persian, Hindi, French, and Italian, across multiple organizations.

Approach

I started with understanding the characteristic of available data to find insightful information within metadata, like finding out span of time period, number of job id’s and time spent to review the application, and started looking for answers of some meaningful questions mentioned below

A. **Jobs Reviewed Over Time:**
   - Objective: Calculate the number of jobs reviewed per hour for each day in November 2020.
   - Your Task: Write an SQL query to calculate the number of jobs reviewed per hour for each day in November 2020.

```
SELECT ds AS Dated, COUNT(job_id) AS job_review_counts,
ROUND(COUNT(job_id)/(SUM(time_spent)/(60*60)),2) AS job_review_per_hour_each_day
FROM job_data
WHERE ds BETWEEN '01-11-2020' AND '30-11-2020'
GROUP BY ds
ORDER BY ds
```



Dated	job_review_counts	job_review_per_hour_each_day
11/25/2020	1	80.00
11/26/2020	1	64.29
11/27/2020	1	34.62
11/28/2020	2	218.18
11/29/2020	1	180.00
11/30/2020	2	180.00

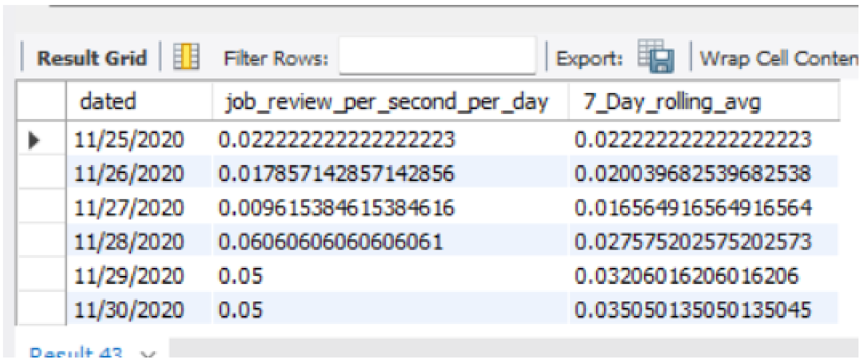
Insight:-

- Hourly rate of job reviews according to the table is ranging between 34.62 to 218.18
- From the result of query we can notice that hourly rate of job reviews is increasing

B. Throughput Analysis:

- Objective: Calculate the 7-day rolling average of throughput (number of events per second).
- Your Task: Write an SQL query to calculate the 7-day rolling average of throughput. Additionally, explain whether you prefer using the daily metric or the 7-day rolling average for throughput, and why.

```
with cte as
(select ds, cast(COUNT(job_id) as float) /cast(SUM(time_spent)as float) as t1
from job_data
WHERE ds BETWEEN '01-11-2020' AND '30-11-2020'
group by ds
)
select ds as dated, t1 as job_review_per_second_per_day,
avg(t1) over (order by ds rows between 6 preceding and current row) as 7_Day_rolling_avg
from cte
```



dated	job_review_per_second_per_day	7_Day_rolling_avg
11/25/2020	0.022222222222222223	0.022222222222222223
11/26/2020	0.017857142857142856	0.020039682539682538
11/27/2020	0.009615384615384616	0.016564916564916564
11/28/2020	0.06060606060606061	0.027575202575202573
11/29/2020	0.05	0.032060162060162060
11/30/2020	0.05	0.035050135050135045

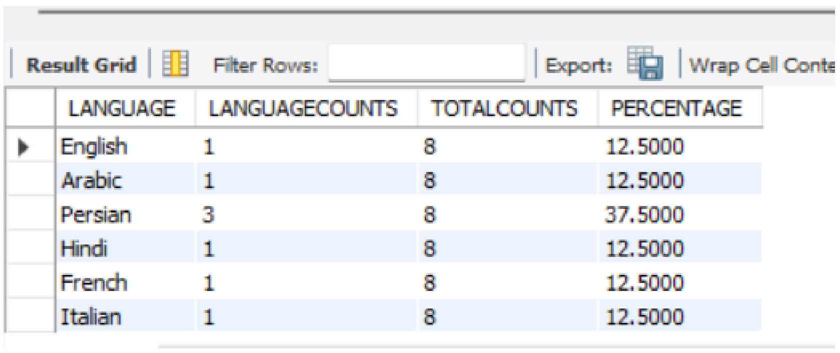
Insight: -

- Job reviews per second each day ranges between 0.0096 to 0.0606
- 7 day rolling average ranges between 0.0165 to 0.0350
- It is observed from the data that there is an upward trend in job reviews
- I would recommend 7 day rolling average data over daily data, because daily data can lead to misinformed decisions due to sudden spikes in data on the other hand 7 day rolling average data would be right fit as it is less impacted from sudden spikes and provides larger view to make realistic sense of data

C. **Language Share Analysis:**

- Objective: Calculate the percentage share of each language in the last 30 days.
- Your Task: Write an SQL query to calculate the percentage share of each language over the last 30 days.

```
WITH CTE AS (
SELECT COUNT(LANGUAGE) AS LANGUAGECOUNTS, LANGUAGE
FROM JOB_DATA
GROUP BY 2),
CTE2 AS (
SELECT COUNT(*) AS TOTALCOUNTS
FROM JOB_DATA)

SELECT LANGUAGE, LANGUAGECOUNTS, TOTALCOUNTS , (LANGUAGECOUNTS/TOTALCOUNTS)*100
AS PERCENTAGE
FROM CTE CROSS JOIN CTE2
```



LANGUAGE	LANGUAGECOUNTS	TOTALCOUNTS	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

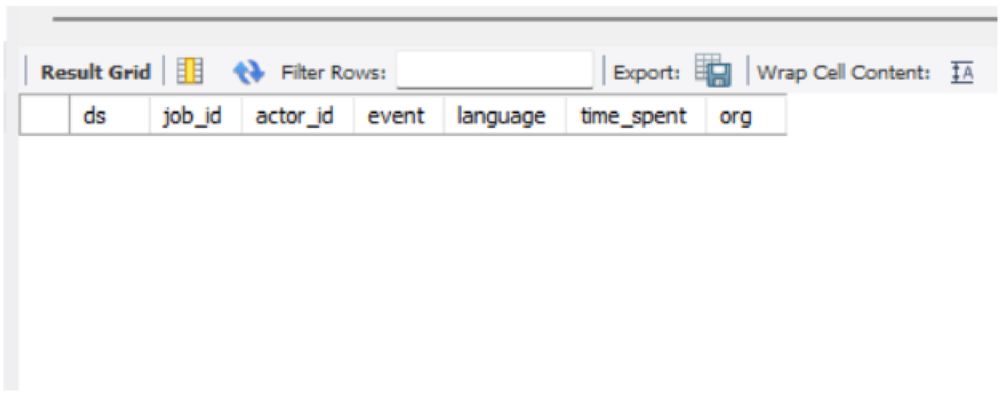
Insights:-

In the context of reviewing job applications, language share analysis is indispensable for gaining insights into applicant demographics and optimizing recruitment strategies. By understanding the distribution of languages used in applications, employers can tailor their hiring processes to better suit the linguistic preferences of their target candidates. This analysis also enables efficient allocation of resources for language-specific communication and ensures effective localization efforts in diverse markets. Moreover, it facilitates benchmarking against competitors' language practices and informs decisions regarding global talent acquisition strategies.

D. **Duplicate Rows Detection:**

- Objective: Identify duplicate rows in the data.
- Your Task: Write an SQL query to display duplicate rows from the job_data table.

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



	ds	job_id	actor_id	event	language	time_spent	org

Insights:-

- since the query statement is to find duplicate rows and not any particular column value, It is understood from the result of query that the data has no duplicate row

Tech-Stack Used

I used mysql workbench to analyse the data, Mysql workbench allows to query data with simpler inputs hence it was best efficient platform to perform queries against the provided data.

Insight

- The hourly rate of job reviews, as observed in the dataset, ranges from 34.62 to 218.18. Notably, an analysis of the query results reveals a discernible upward trend in the hourly rate of job reviews over time.

- Job reviews per second, calculated on a daily basis, exhibit a range between 0.0096 and 0.0606. Meanwhile, when considering a 7-day rolling average, the range extends from 0.0165 to 0.0350. This observation indicates a consistent increase in the frequency of job reviews, suggesting a growing workload or improved efficiency in the review process.

- The absence of duplicate rows in the dataset, as evidenced by the result of the query, underscores the integrity and cleanliness of the data. This finding instils confidence in the reliability of the dataset for analytical purposes and highlights the effectiveness of data management practices.

- In the realm of reviewing job applications, language share analysis emerges as a critical tool for understanding applicant demographics and optimizing recruitment strategies. By discerning the distribution of languages used in applications, employers can tailor their hiring processes to better align with the linguistic preferences of their target candidates. This strategic approach not only enhances communication effectiveness but also enables efficient resource allocation and informed decision-making in global talent acquisition endeavours.

Results

1. No. of job reviews per hour each day ranging from 34.65 to 218.18
2. Choosing 7 day rolling average against daily data to enhance decisions based on output
3. Language share analysis ~ out of 6 unique languages, Persian is the highest used language
4. Duplicate rows detection:- no entirely duplicate rows detected in the job_data table.

Case Study 2: Investigating Metric Spike

Operation Analytics and Investigating Metric Spike

Project description

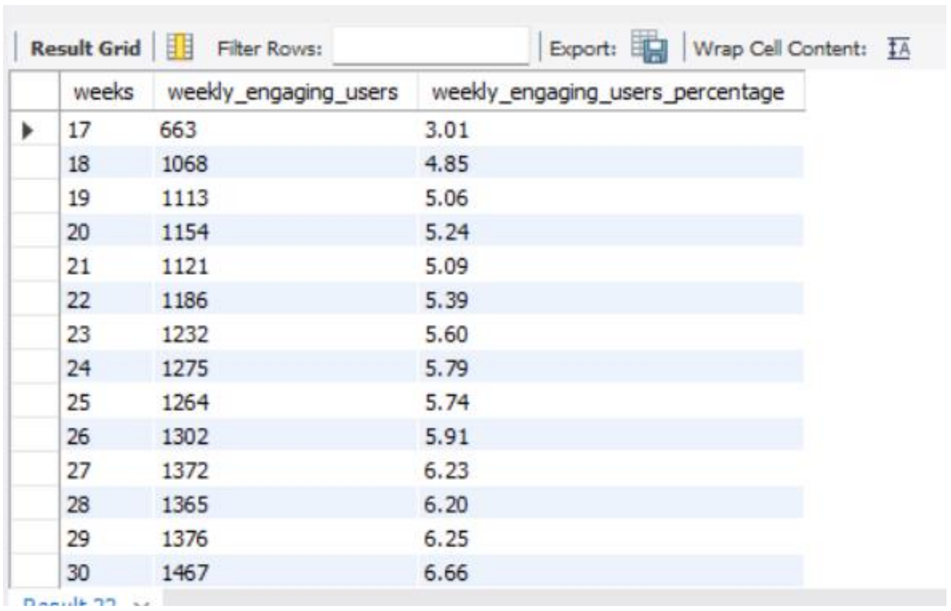
The data set contains three tables 1) users, 2) events, 3) email_events. These tables contain information of users engaging in events via different devices. The data set encompasses date range of Jan 2013 to Aug 2014. This data set can be related to a software company. Using these three tables we will be finding outputs of a specific questions

Approach

I used infile function to inject data to the platform and I started with understanding the characteristic of available data and found out events table has highest number of rows in all three tables 3 – 325255 , in users table data is ranging in dates jan 2013 and aug 2014 while events and email_events table contain data ranging in dates may 2014 to aug 2014, also changed all the columns in all three tables that contained data of dates in varchar to date time format. We will be using these tables to find results to below mentioned queries.

A. **Weekly User Engagement:**
- Objective: Measure the activeness of users on a weekly basis.
- Your Task: Write an SQL query to calculate the weekly user engagement.

```
with cte as (
select extract(week from occurred_at) weeks, count(distinct user_id)
weekly_engaging_users
from events
group by 1
order by 1
),
cte2 as (
select sum(weekly_engaging_users) as totalcounts
from cte
)
select weeks, weekly_engaging_users, round((weekly_engaging_users/totalcounts)*100,2)
as weekly_engaging_users_percentage
from cte2 cross join cte
```



weeks	weekly_engaging_users	weekly_engaging_users_percentage
17	663	3.01
18	1068	4.85
19	1113	5.06
20	1154	5.24
21	1121	5.09
22	1186	5.39
23	1232	5.60
24	1275	5.79
25	1264	5.74
26	1302	5.91
27	1372	6.23
28	1365	6.20
29	1376	6.25
30	1467	6.66

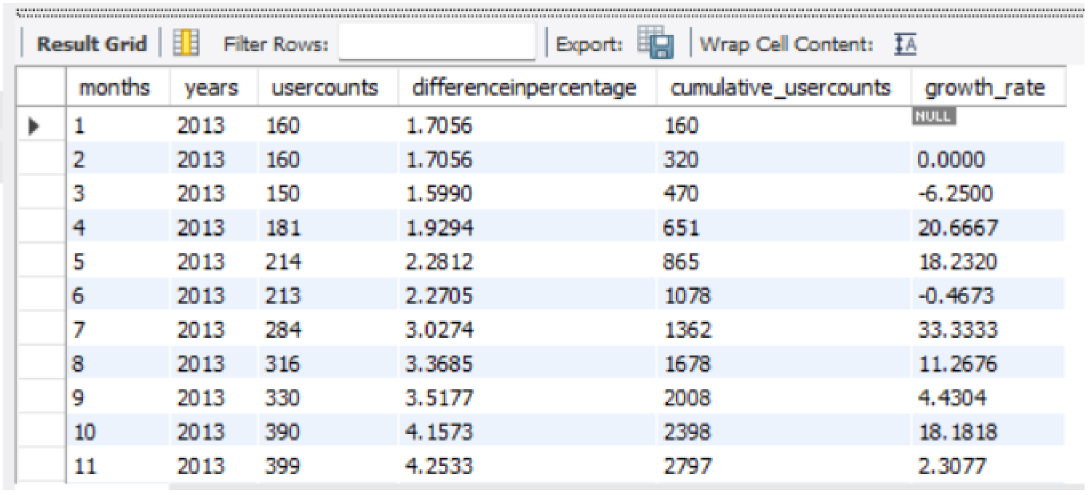
Insight:-

- Span of the event dataset ranges from 1st may 2014 to 31st aug 2014 i.e 4 months
- Weeks unique users engaging in events ranges from 104 to 1467 i.e 0.47% to 6.66%
- Where week No. 30 has the highest (i.e. 1467 engaging users) and week No. 35 has the lowest (i.e 104 engaging users)

B. **User Growth Analysis:**

- Objective: Analyze the growth of users over time for a product.
- Your Task: Write an SQL query to calculate the user growth for the product.

```
WITH cte AS (
   SELECT EXTRACT(MONTH FROM created_at) AS months,
      EXTRACT(YEAR FROM created_at) AS years,
      COUNT(DISTINCT user_id) AS usercounts
   FROM users
   GROUP BY 1, 2
   ORDER BY 2, 1),
cte2 AS (SELECT SUM(usercounts) AS totalusers
   FROM cte)
SELECT
   months,
   years,
   usercounts,
   (usercounts / totalusers) * 100 AS differenceinpercentage,
   SUM(usercounts) OVER (ORDER BY years, months ROWS BETWEEN UNBOUNDED
PRECEDING AND CURRENT ROW) AS cumulative_usercounts,
   ((usercounts - LAG(usercounts) OVER (ORDER BY years, months)) / LAG(usercounts)
OVER (ORDER BY years, months)) * 100 AS growth_rate
FROM cte CROSS JOIN cte2;
```



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

months	years	usercounts	differenceinpercentage	cumulative_usercounts	growth_rate
1	2013	160	1.7056	160	NULL
2	2013	160	1.7056	320	0.0000
3	2013	150	1.5990	470	-6.2500
4	2013	181	1.9294	651	20.6667
5	2013	214	2.2812	865	18.2320
6	2013	213	2.2705	1078	-0.4673
7	2013	284	3.0274	1362	33.3333
8	2013	316	3.3685	1678	11.2676
9	2013	330	3.5177	2008	4.4304
10	2013	390	4.1573	2398	18.1818
11	2013	399	4.2533	2797	2.3077

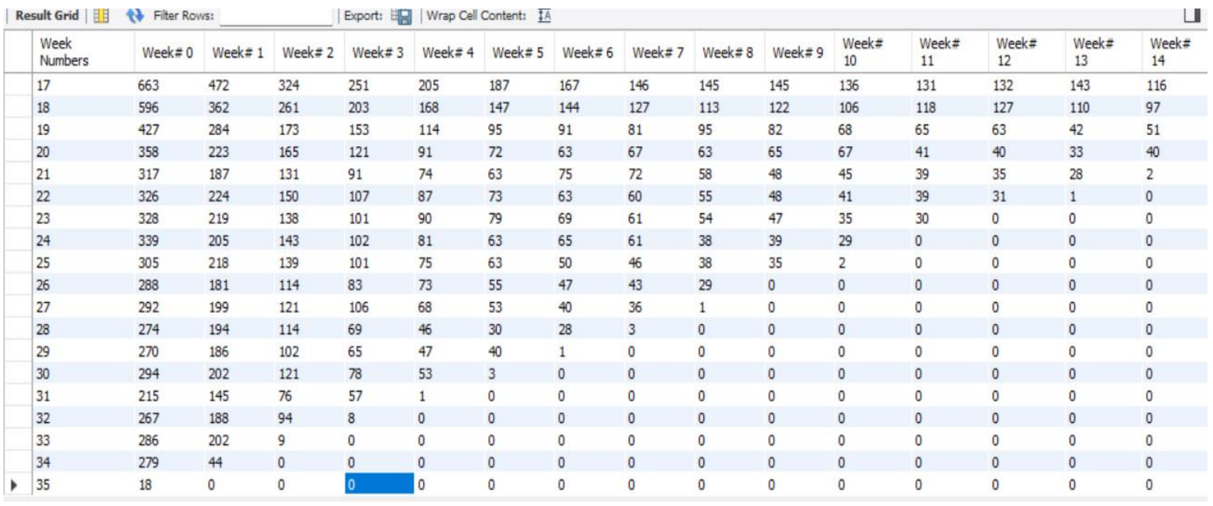
Insight:-

- Activation of user takes up to 3 minutes of time after creating.
- August 2014 recorded the highest user activation
- July 2013 recorded the highest growth rate of 33.33% from 213 to 284 users addition
- Overall the user counts are increasing continuously every month except there's been minute dip thrice over the period of 20 months 1) march 2013, 2) June 2013, 3) feb 2014

C. **Weekly Retention Analysis:**

- Objective: Analyze the retention of users on a weekly basis after signing up for a product.
- Your Task: Write an SQL query to calculate the weekly retention of users based on their sign-up cohort.

```
SELECT  first_login_week AS "Week Numbers",
SUM(CASE WHEN week_number = 0 THEN 1 ELSE 0 END) AS "Week# 0",

SUM(CASE WHEN week_number = 1 THEN 1 ELSE 0 END) AS "Week# 1",

SUM(CASE WHEN week_number = 2 THEN 1 ELSE 0 END) AS "Week# 2",

SUM(CASE WHEN week_number = 3 THEN 1 ELSE 0 END) AS "Week# 3",

SUM(CASE WHEN week_number = 4 THEN 1 ELSE 0 END) AS "Week# 4",

SUM(CASE WHEN week_number = 5 THEN 1 ELSE 0 END) AS "Week# 5",

SUM(CASE WHEN week_number = 6 THEN 1 ELSE 0 END) AS "Week# 6",

SUM(CASE WHEN week_number = 7 THEN 1 ELSE 0 END) AS "Week# 7",

SUM(CASE WHEN week_number = 8 THEN 1 ELSE 0 END) AS "Week# 8",

SUM(CASE WHEN week_number = 9 THEN 1 ELSE 0 END) AS "Week# 9",

SUM(CASE WHEN week_number = 10 THEN 1 ELSE 0 END) AS "Week# 10",

SUM(CASE WHEN week_number = 11 THEN 1 ELSE 0 END) AS "Week# 11",

SUM(CASE WHEN week_number = 12 THEN 1 ELSE 0 END) AS "Week# 12",

SUM(CASE WHEN week_number = 13 THEN 1 ELSE 0 END) AS "Week# 13",

SUM(CASE WHEN week_number = 14 THEN 1 ELSE 0 END) AS "Week# 14",

SUM(CASE WHEN week_number = 15 THEN 1 ELSE 0 END) AS "Week# 15",

SUM(CASE WHEN week_number = 16 THEN 1 ELSE 0 END) AS "Week# 16",

SUM(CASE WHEN week_number = 17 THEN 1 ELSE 0 END) AS "Week# 17",

SUM(CASE WHEN week_number = 18 THEN 1 ELSE 0 END) AS "Week# 18",


FROM (SELECT B.user_id,  B.current_week,  A.first_login_week, B.current_week -
A.first_login_week AS week_number
FROM (SELECT  user_id,WEEK(occurred_at) AS current_week
FROM events GROUP BY 1, 2) as  B ,
(SELECT user_id,MIN(WEEK(occurred_at)) AS first_login_week
 FROM events GROUP BY 1) as A WHERE B.user_id = A.user_id) T1 GROUP BY
first_login_week ORDER BY first_login_week
```


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

Week Numbers	Week# 0	Week# 1	Week# 2	Week# 3	Week# 4	Week# 5	Week# 6	Week# 7	Week# 8	Week# 9	Week# 10	Week# 11	Week# 12	Week# 13	Week# 14
17	663	472	324	251	205	187	167	146	145	145	136	131	132	143	116
18	596	362	261	203	168	147	144	127	113	122	106	118	127	110	97
19	427	284	173	153	114	95	91	81	95	82	68	65	63	42	51
20	358	223	165	121	91	72	63	67	63	65	67	41	40	33	40
21	317	187	131	91	74	63	75	72	58	48	45	39	35	28	2
22	326	224	150	107	87	73	63	60	55	48	41	39	31	1	0
23	328	219	138	101	90	79	69	61	54	47	35	30	0	0	0
24	339	205	143	102	81	63	65	61	38	39	29	0	0	0	0
25	305	218	139	101	75	63	50	46	38	35	2	0	0	0	0
26	288	181	114	83	73	55	47	43	29	0	0	0	0	0	0
27	292	199	121	106	68	53	40	36	1	0	0	0	0	0	0
28	274	194	114	69	46	30	28	3	0	0	0	0	0	0	0
29	270	186	102	65	47	40	1	0	0	0	0	0	0	0	0
30	294	202	121	78	53	3	0	0	0	0	0	0	0	0	0
31	215	145	76	57	1	0	0	0	0	0	0	0	0	0	0
32	267	188	94	8	0	0	0	0	0	0	0	0	0	0	0
33	286	202	9	0	0	0	0	0	0	0	0	0	0	0	0
34	279	44	0	0	0	0	0	0	0	0	0	0	0	0	0
35	18	0	0	0	0	0	0	0	0	0	0	0	0	0	0

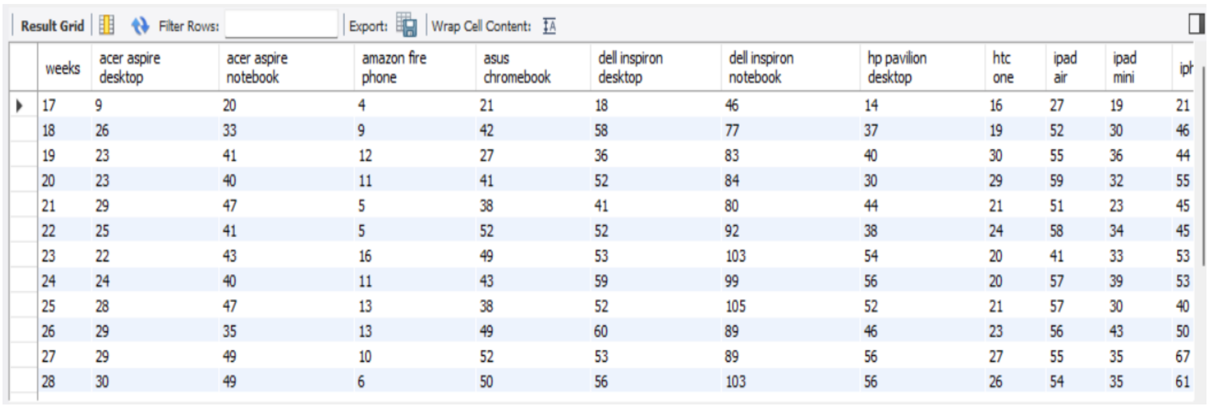
Insight:-

- There are 19 months in the data hence retention is accounted for 18 months starting from the first (i.e 17th) to the last (i.e 35th)
- If we were to take an example of the data set here, No. of users in first week (i.e. 17th week) is 663 and by last week the number went down to only 5 (company should consider these 5 users as loyal users their user_id's are 1) '3633' , 2) '4347', 3) '4972' 4) '10612', 5) '11687'
- The customer retention is very important metric for every business and improving the customer retention will extend the longevity of the company's life span
- No. of users using the platform after signing up is decreasing every week in this data set , mostly more than 50% of customers stops using after 3-4 months of signing up.

D. **Weekly Engagement Per Device:**

- Objective: Measure the activeness of users on a weekly basis per device.
- Your Task: Write an SQL query to calculate the weekly engagement per device.

```
select  week(occurred_at) as weeks ,
count(distinct case when device = "acer aspire desktop" then user_id else null end ) as 'acer
aspire desktop',
count(distinct case when device = "acer aspire notebook" then user_id else null end ) as
'acer aspire notebook',
count(distinct case when device = "amazon fire phone" then user_id else null end ) as
'amazon fire phone',
count(distinct case when device = "asus chromebook" then user_id else null end ) as 'asus
chromebook',
count(distinct case when device = "dell inspiron desktop" then user_id else null end ) as 'dell
inspiron desktop',
count(distinct case when device = "dell inspiron notebook" then user_id else null end ) as
'dell inspiron notebook',
count(distinct case when device = "hp pavilion desktop" then user_id else null end ) as 'hp
pavilion desktop',
count(distinct case when device = "htc one" then user_id else null end ) as 'htc one',
count(distinct case when device = "ipad air" then user_id else null end ) as 'ipad air',
count(distinct case when device = "ipad mini" then user_id else null end ) as 'ipad mini',
count(distinct case when device = "iphone 4s" then user_id else null end ) as 'iphone 4s',
count(distinct case when device = "iphone 5" then user_id else null end ) as 'iphone 5',
count(distinct case when device = "iphone 5s" then user_id else null end ) as 'iphone 5s',
count(distinct case when device = "kindle fire" then user_id else null end ) as 'kindle fire',
count(distinct case when device = "lenovo thinkpad" then user_id else null end ) as 'lenovo
thinkpad',
count(distinct case when device = "mac mini" then user_id else null end ) as 'mac mini',
count(distinct case when device = "macbook air" then user_id else null end ) as 'macbook
air',
count(distinct case when device = "macbook pro" then user_id else null end ) as 'macbook
pro',
count(distinct case when device = "nexus 10" then user_id else null end ) as 'nexus 10',
count(distinct case when device = "nexus 5" then user_id else null end ) as 'nexus 5',
count(distinct case when device = "nexus 7" then user_id else null end ) as 'nexus 7',
count(distinct case when device = "nokia lumia 635" then user_id else null end ) as 'nokia
lumia 635',
count(distinct case when device = "samsumg galaxy tablet" then user_id else null end ) as
'samsumg galaxy tablet',
count(distinct case when device = "samsung galaxy note" then user_id else null end ) as
'samsung galaxy note',
count(distinct case when device = "samsung galaxy s4" then user_id else null end ) as
'samsung galaxy s4',
count(distinct case when device = "windows surface" then user_id else null end ) as
'windows surface'
from events
where event_type = 'engagement'
group by 1
order by 1
```



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

weeks	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	iph
17	9	20	4	21	18	46	14	16	27	19	21
18	26	33	9	42	58	77	37	19	52	30	46
19	23	41	12	27	36	83	40	30	55	36	44
20	23	40	11	41	52	84	30	29	59	32	55
21	29	47	5	38	41	80	44	21	51	23	45
22	25	41	5	52	52	92	38	24	58	34	45
23	22	43	16	49	53	103	54	20	41	33	53
24	24	40	11	43	59	99	56	20	57	39	53
25	28	47	13	38	52	105	52	21	57	30	40
26	29	35	13	49	60	89	46	23	56	43	50
27	29	49	10	52	53	89	56	27	55	35	67
28	30	49	6	50	56	103	56	26	54	35	61

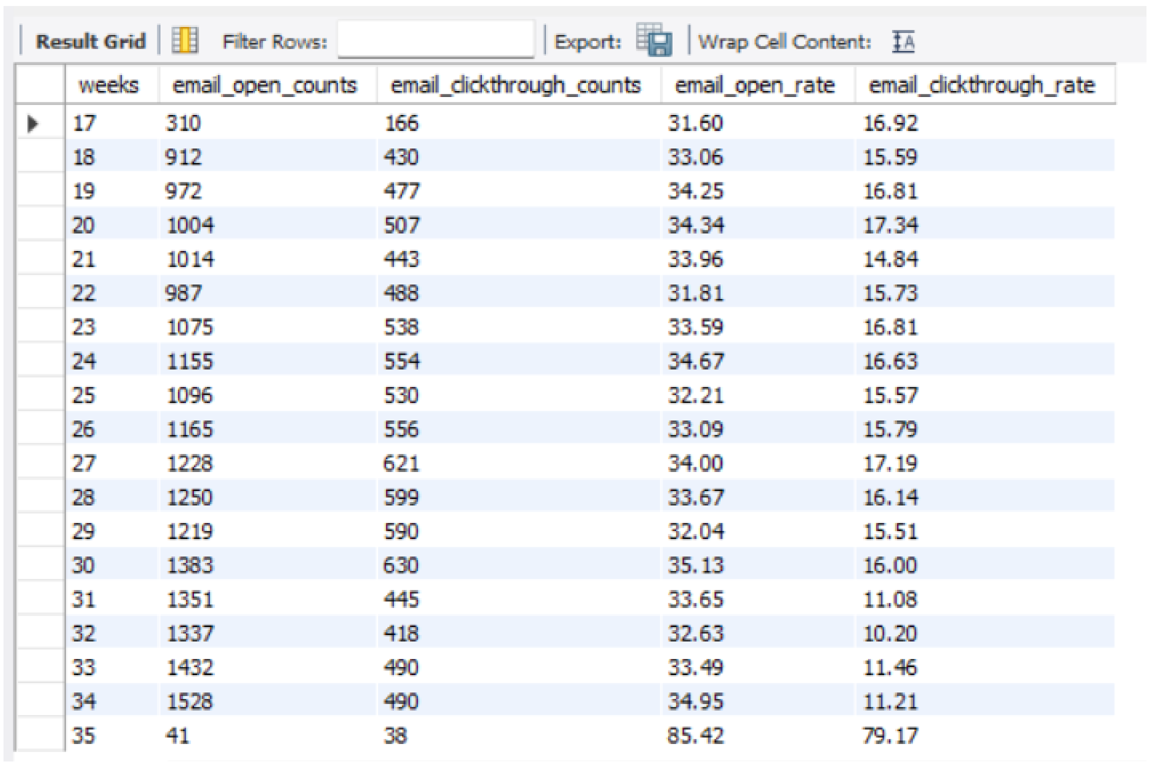
Insights:-

- Top 5 devices with highest unique users engagement
    1. MacBook Pro
    2. Lenovo Thinkpad
    3. I phone 5
    4. Macbook Air
    5. Samsung galaxy s4
- Samsung galaxy tablet is a device with least unique users
- Week no. 27,28,29,30 are the most engaging weeks with highest unique users engagement in week no.30

E. **Email Engagement Analysis:**

- Objective: Analyze how users are engaging with the email service.
- Your Task: Write an SQL query to calculate the email engagement metrics.

```
with cte as (
SELECT week(occurred_at) as weeks ,
count(case when action = 'email_open' then user_id else null end) as email_open_counts ,
count(case when action = 'email_clickthrough' then user_id else null end ) as
email_clickthrough_counts,
count(case when action = 'sent_weekly_digest' then user_id else null end ) as
sent_weekly_digest_counts,
count(case when action = 'sent_reengagement_email' then user_id else null end ) as
sent_reengagement_email_counts
FROM EMAIL_EVENTS
GROUP BY 1
order by 1)
select weeks, email_open_counts,email_clickthrough_counts,
round((email_open_counts/(sent_weekly_digest_counts+sent_reengagement_email_counts)
)*100,2) as  email_open_rate,
round((email_clickthrough_counts/(sent_weekly_digest_counts+sent_reengagement_email_
counts))*100,2) as email_clickthrough_rate
from cte
group by 1
order by 1
```



weeks	email_open_counts	email_clickthrough_counts	email_open_rate	email_clickthrough_rate
17	310	166	31.60	16.92
18	912	430	33.06	15.59
19	972	477	34.25	16.81
20	1004	507	34.34	17.34
21	1014	443	33.96	14.84
22	987	488	31.81	15.73
23	1075	538	33.59	16.81
24	1155	554	34.67	16.63
25	1096	530	32.21	15.57
26	1165	556	33.09	15.79
27	1228	621	34.00	17.19
28	1250	599	33.67	16.14
29	1219	590	32.04	15.51
30	1383	630	35.13	16.00
31	1351	445	33.65	11.08
32	1337	418	32.63	10.20
33	1432	490	33.49	11.46
34	1528	490	34.95	11.21
35	41	38	85.42	79.17

Insights:

- It is understood from the dataset that weekly digest is sent to users and reengagement is sent to select users.
- It is observed that 30 to 35% weekly digest emails are opened by users and 10 to 16% clickthrough is done by users.

Tech-Stack Used

I used mysql workbench to analyse the data, Mysql workbench allows to query data with simpler inputs hence it was best efficient platform to perform queries against the provided data.

Insight

The analysis of the event dataset spanning from May 1st, 2014 to August 31st, 2014, unveils several significant observations regarding user engagement and retention. Over the four-month period, weekly engagement fluctuated notably, ranging from 104 to 1467 unique users, representing 0.47% to 6.66% of the total user base. Week 30 emerged as the peak engagement period with 1467 users, while week 35 recorded the lowest engagement with only 104 users actively participating in events.

Furthermore, user activation, occurring within approximately three minutes post-creation, witnessed its peak in August 2014. Notably, July 2013 exhibited the highest growth rate, marking a substantial 33.33% increase in users from 213 to 284. Despite minor dips in March 2013, June 2013, and February 2014, overall user counts demonstrated a consistent monthly increase over the 20-month period under consideration.

Moreover, retention analysis indicates a critical aspect of user loyalty. For instance, starting with 663 users in the first week (17th week), the count dwindled to a mere 5 users by the last week. These remaining users (identified by user IDs: 3633, 4347, 4972, 10612, 11687) are considered loyal users, highlighting the importance of customer retention in sustaining business longevity. Additionally, data regarding device usage revealed the top five devices with the highest engagement, including MacBook Pro, Lenovo Thinkpad, iPhone 5, MacBook Air, and Samsung Galaxy S4, while Samsung Galaxy Tablet exhibited the least engagement. The analysis also identified weeks 27 to 30 as the most engaging, aligning with the highest unique user engagement observed in week 30. Finally, insights into email engagement revealed that approximately 30 to 35% of weekly digest emails are opened by users, with 10 to 16% engaging further through clickthrough actions, underlining the significance of personalized reengagement strategies to enhance user interaction and retention rates.

Results

1. weekly user engagement - The event dataset, spanning May 1st to August 31st, 2014, showed weekly engagement ranging from 104 to 1467 unique users, indicating varying levels of user participation over time.

2. user growth analysis - User activation, occurring within three minutes post-creation, peaked in August 2014, while July 2013 experienced the highest growth rate, showcasing a 33.33% increase in user numbers.

3. weekly retention analysis - Retention analysis revealed a decline in user numbers over time, with only a small fraction remaining loyal after 18 months, emphasizing the importance of effective retention strategies for long-term sustainability.

4. weekly engagement per device - Device usage data identified MacBook Pro, Lenovo Thinkpad, iPhone 5, MacBook Air, and Samsung Galaxy S4 as the top five devices with the highest engagement, while Samsung Galaxy Tablet showed the least engagement.

5. Email Engagement Analysis- Insights from the dataset indicate the implementation of weekly digest emails and targeted reengagement efforts, with approximately 30 to 35% of weekly digest emails being opened by users and 10 to 16% resulting in clickthrough actions, suggesting varying levels of user engagement with these communication channels.