OPERATION ANALYTICS AND INVESTIGATING METRIC SPIKE

By Vishalini Arul Devarajan

PROJECT DESCRIPTION

The purpose of the project is to understand sudden changes in key indicators and discover meaningful insights that will enrich the operation of the business. We look at this information from the following perspectives:

Case Study 1: Job Data Analysis

- Jobs Reviewed Over Time
- Throughput Analysis
- Language Share Analysis
- Duplicate Rows Detection

PROJECT DESCRIPTION

Case Study 2: **Investigating Metric SpikeWeekly User Engagement**

- User Growth Analysis
- Weekly Retention Analysis
- Weekly Engagement Per Device
- Email Engagement Analysis

APPROACH

- 1. Read the given data description and understand the problem.
- 2. Carefully go through the data and make sure that you understand the provided variable and attribute.
- 3. I imported files by using MySQL Workbench into the operation_analytics database and started making queries with the questions presented to get the desired outcome.
- 4. I ran the queries and, in case there is a mistake in the code, I cross check them by modifying the code and run the code without errors.
- 5. Once the questions are done using the queries, I take screenshots and save them in a file.
- 6. Finally, I added the code into the document.

TECH STACK USED IN THIS PROJECT

- Mysql Workbench 8.0.37
- Microsoft Excel To Visualize The Outputs

CASE STUDY 1: JOB DATA ANALYSIS

CASE STUDY 1: JOB DATA ANALYSIS

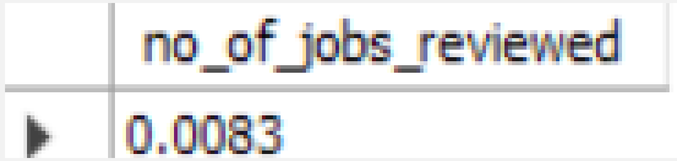
JOBS REVIEWED OVER TIME: CALCULATE THE NUMBER OF JOBS REVIEWEDPER HOUR FOR EACH DAY IN NOVEMBER 2020.

Task A: Create a SQL query to determine how many jobs are reviewed every hour on a daily basis in November 2020.

Query (distinct_job_id):

```
select count(distinct job_id)/(30*24) as no_of_jobs_reviewed

from job_data;
```

Output:



```
no_of_jobs_reviewed
0.0083
```

The number of jobs reviewed per hour for each day in November 2020 is 0.0083

CASE STUDY 1: JOB DATA ANALYSIS

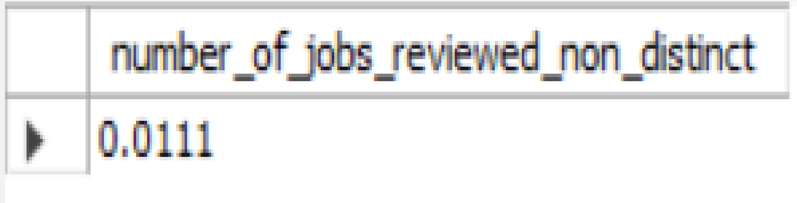
JOBS REVIEWED OVER TIME: CALCULATE THE NUMBER OF JOBS REVIEWEDPER HOUR FOR EACH DAY IN NOVEMBER 2020.

Task A: Create a SQL query to determine how many jobs are reviewed every hour on a daily basis in November 2020.

Query (non_distinct_job_id):

```
Select  count(job_id)/(30*24) as number_of_jobs_reviewed_non_distinct

from job_data;
```

Output:



	number_of_jobs_reviewed_non_distinct
▶	0.0111

The number of jobs reviewed per hour for each day in November 2020 is 0.0111

CASE STUDY 1: JOB DATA ANALYSIS

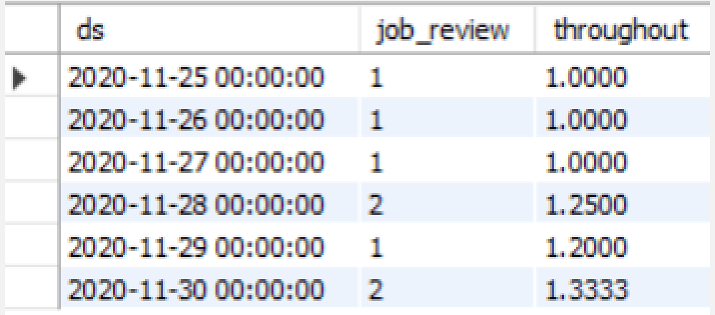
THROUGHPUT ANALYSIS: CALCULATE THE 7-DAY ROLLING AVERAGE OFTHROUGHPUT (NUMBER OF EVENTS PER SECOND).

Task B: 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.

Query:

```
select ds, job_review, avg(job_review) over

(order by ds rows between 5 preceding and current row) as throughout

from (select ds, count(distinct job_id) as job_review from job_data

where ds between "2020-11-01" and"2020-11-30“

group by ds order by ds)a;
```

Output:



ds	job_review	throughout
2020-11-25 00:00:00	1	1.0000
2020-11-26 00:00:00	1	1.0000
2020-11-27 00:00:00	1	1.0000
2020-11-28 00:00:00	2	1.2500
2020-11-29 00:00:00	1	1.2000
2020-11-30 00:00:00	2	1.3333

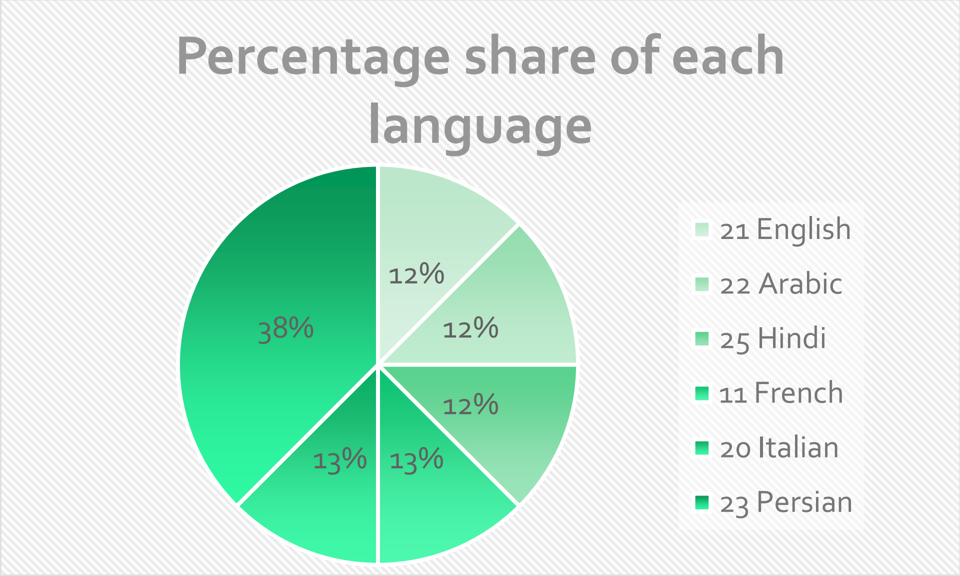
CASE STUDY 1: JOB DATA ANALYSIS

LANGUAGE SHARE ANALYSIS: CALCULATE THE PERCENTAGE SHARE OF EACHLANGUAGE IN THE LAST 30 DAYS.

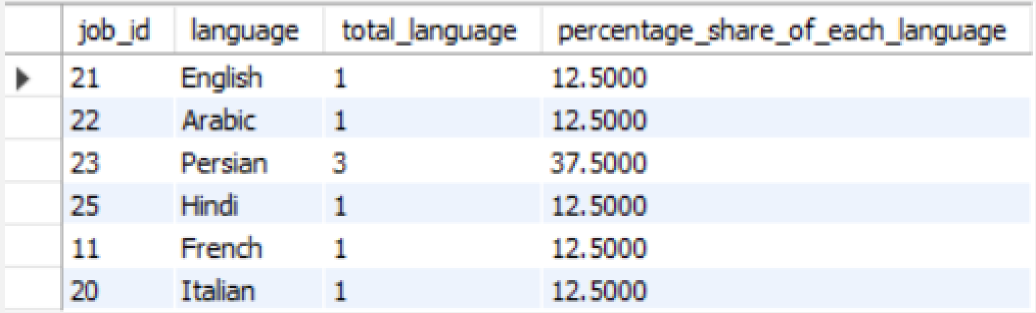
Task C: Write an SQL query to calculate the percentage share of each language over the last30 days.

Query:

```
Select job_data.job_id, job_data.language,

count(job_data.language) as total_language,

((count(job_data.language)/(select count(*) from job_data))*100) as

percentage_share_of_each_language

from job_data

group by job_data.job_id, job_data.language;
```


Percentage share of each language

Output:



job_id	language	total_language	percentage_share_of_each_language
21	English	1	12.5000
22	Arabic	1	12.5000
23	Persian	3	37.5000
25	Hindi	1	12.5000
11	French	1	12.5000
20	Italian	1	12.5000

CASE STUDY 1: JOB DATA ANALYSIS

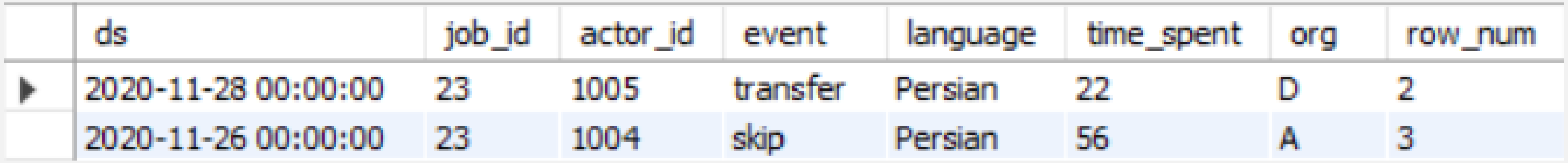
DUPLICATE ROWS DETECTION: IDENTIFY DUPLICATE ROWS IN THE DATA.

Task D: Write an SQL query to display duplicate rows from the job_data table.

Query:

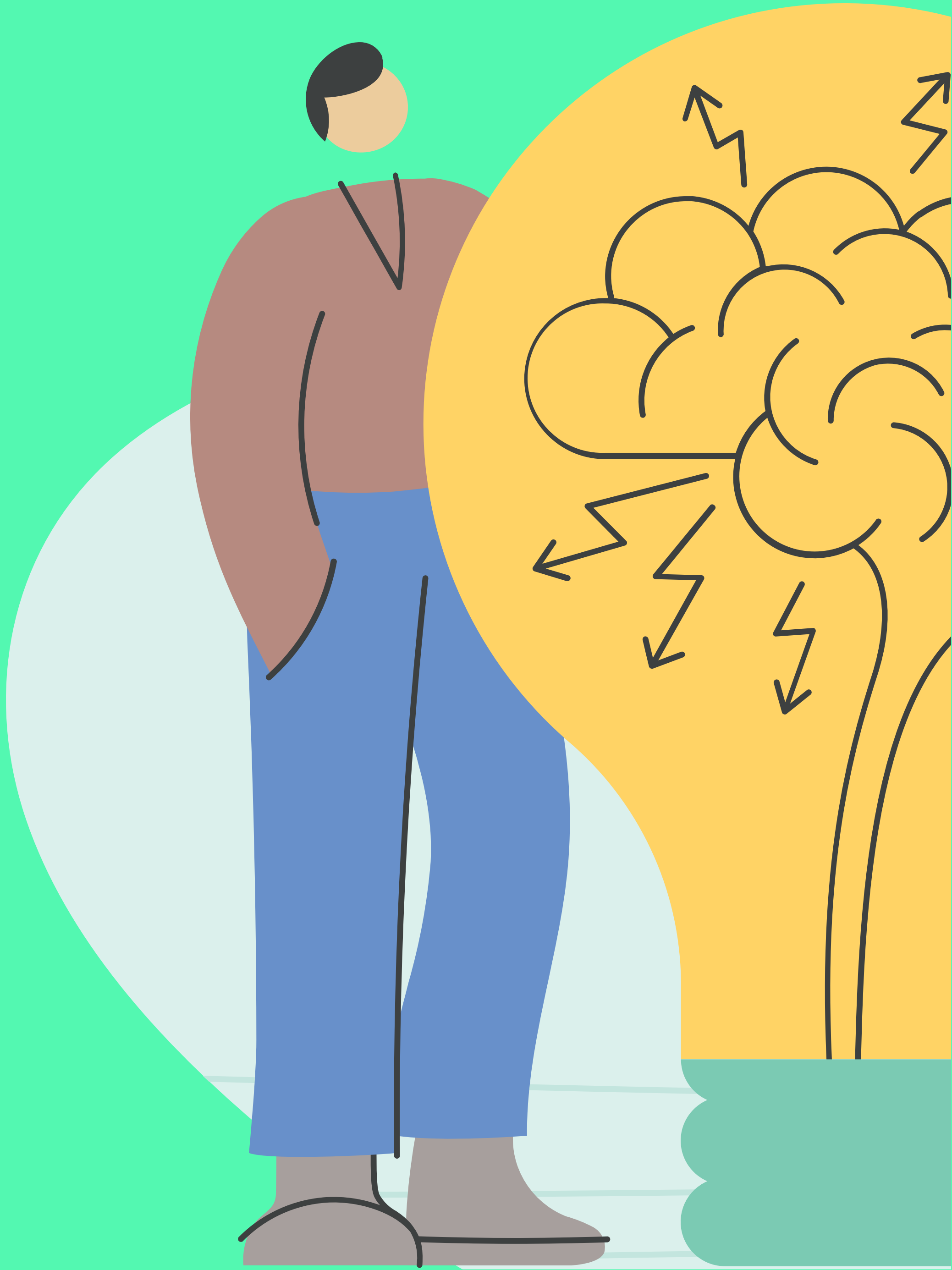
```
select * from(

select *, row_number()over(partition by job_id) as row_num

from job_data) a

where row_num>1;
```

Output:



	ds	job_id	actor_id	event	language	time_spent	org	row_num
▶	2020-11-28 00:00:00	23	1005	transfer	Persian	22	D	2
	2020-11-26 00:00:00	23	1004	skip	Persian	56	A	3

CASE STUDY 2: INVESTIGATING METRIC SPIKE



CASE STUDY 2: INVESTIGATING METRIC SPIKE

WEEKLY USER ENGAGEMENT: MEASURE THE ACTIVENESS OF USERS ON A WEEKLYBASIS.

Task A: Write an SQL query to calculate the weekly user engagement. This is a powerful tool in public speaking. It involves varying pitch, tone, and volume to convey emotion, emphasize points, and maintain interest.

Query:

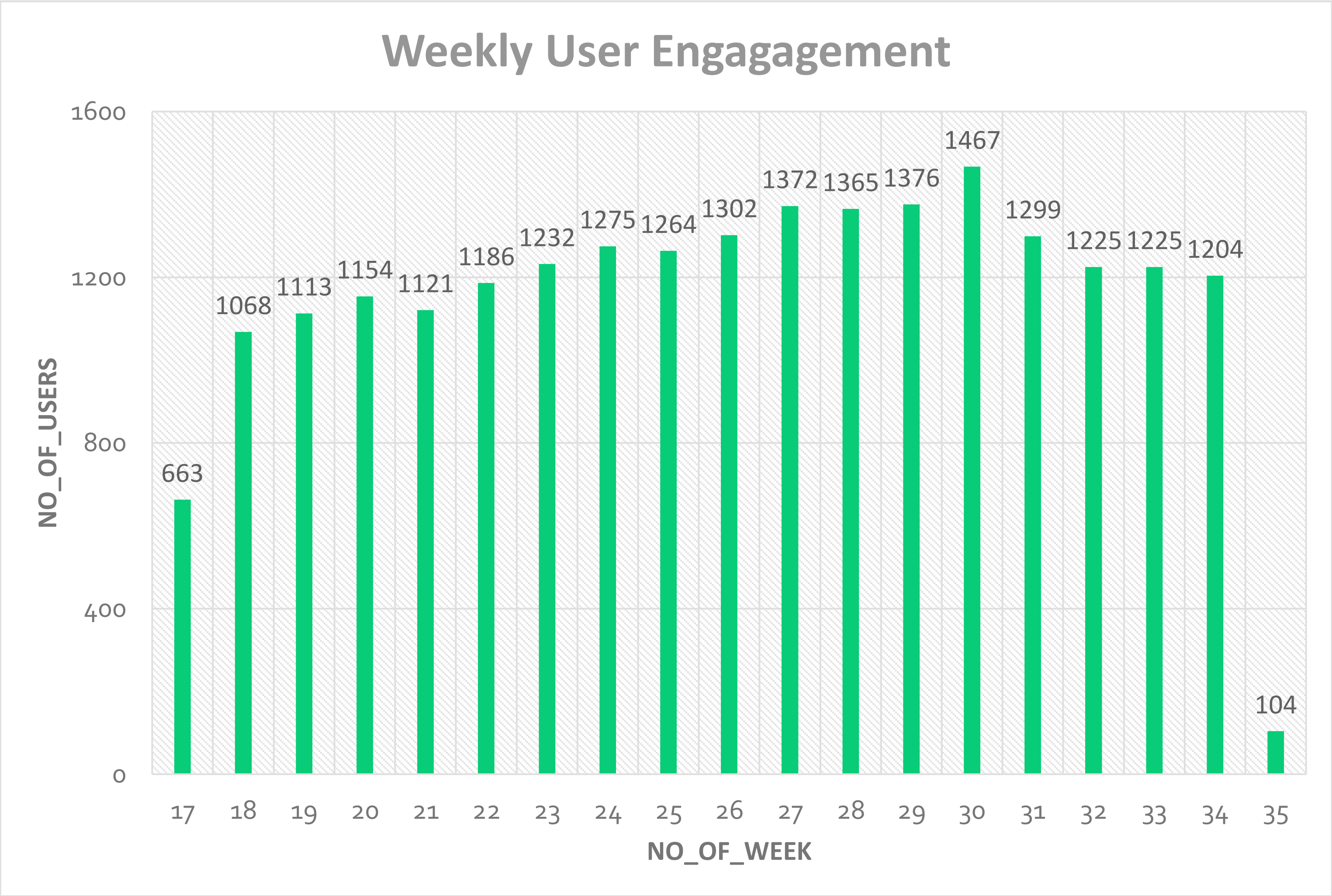
```
select week(occurred_at) as no_of_week,

count(distinct user_id) as no_of_usersfrom events

group by no_of_week;
```

CASE STUDY 2: INVESTIGATING METRIC SPIKE

WEEKLY USER ENGAGEMENT: MEASURE THE ACTIVENESS OF USERS ON A WEEKLYBASIS.

Output:

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



CASE STUDY 2: INVESTIGATING METRIC SPIKE

USER GROWTH ANALYSIS: ANALYZE THE GROWTH OF USERS OVER TIME FOR APRODUCT.

Task B: Write an SQL query to calculate the user growth for the product.

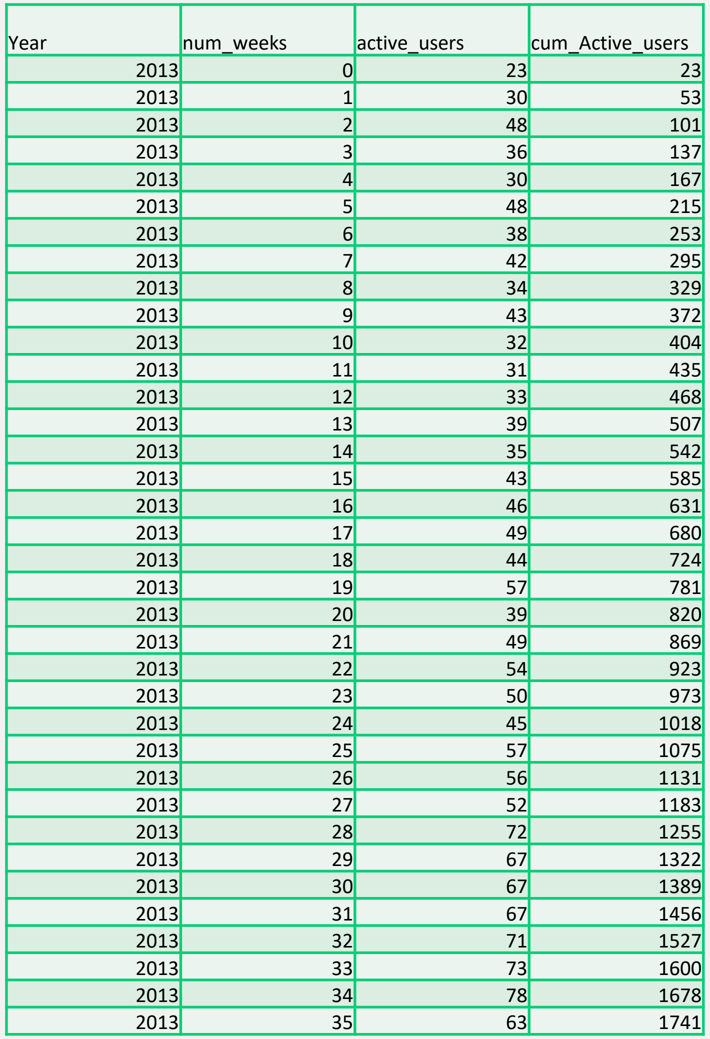
Query:

```
select year, num_weeks, active_users, sum(active_users) over(order by
year, num_weeks rows between unbounded preceding and current row) as
cum_active_users from (select extract(year from activated_at) as year,
extract(week from activated_at) as num_weeks,
count(distinct user_id) as active_users
from users
group by year, num_weeksorder by year, num_weeks) a;
```

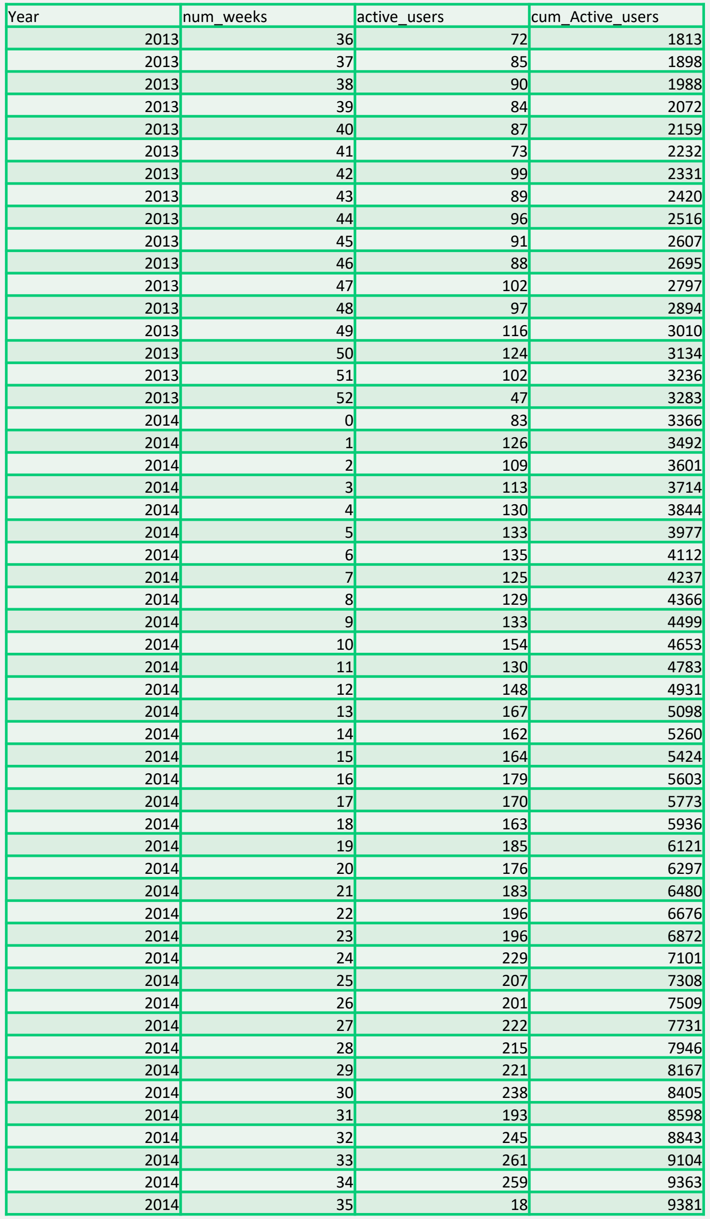
CASE STUDY 2: INVESTIGATING METRIC SPIKE

USER GROWTH ANALYSIS: ANALYZE THE GROWTH OF USERS OVER TIME FOR APRODUCT.

Output:



Year	num_weeks	active_users	cum_Active_users
2013	0	23	23
2013	1	30	53
2013	2	48	101
2013	3	36	137
2013	4	30	167
2013	5	48	215
2013	6	38	253
2013	7	42	295
2013	8	34	329
2013	9	43	372
2013	10	32	404
2013	11	31	435
2013	12	33	468
2013	13	39	507
2013	14	35	542
2013	15	43	585
2013	16	46	631
2013	17	49	680
2013	18	44	724
2013	19	57	781
2013	20	39	820
2013	21	49	869
2013	22	54	923
2013	23	50	973
2013	24	45	1018
2013	25	57	1075
2013	26	56	1131
2013	27	52	1183
2013	28	72	1255
2013	29	67	1322
2013	30	67	1389
2013	31	67	1456
2013	32	71	1527
2013	33	73	1600
2013	34	78	1678
2013	35	63	1741



Year	num_weeks	active_users	cum_Active_users
2013	36	72	1813
2013	37	85	1898
2013	38	90	1988
2013	39	84	2072
2013	40	87	2159
2013	41	73	2232
2013	42	99	2331
2013	43	89	2420
2013	44	96	2516
2013	45	91	2607
2013	46	88	2695
2013	47	102	2797
2013	48	97	2894
2013	49	116	3010
2013	50	124	3134
2013	51	102	3236
2013	52	47	3283
2014	0	83	3366
2014	1	126	3492
2014	2	109	3601
2014	3	113	3714
2014	4	130	3844
2014	5	133	3977
2014	6	135	4112
2014	7	125	4237
2014	8	129	4366
2014	9	133	4499
2014	10	154	4653
2014	11	130	4783
2014	12	148	4931
2014	13	167	5098
2014	14	162	5260
2014	15	164	5424
2014	16	179	5603
2014	17	170	5773
2014	18	163	5936
2014	19	185	6121
2014	20	176	6297
2014	21	183	6480
2014	22	196	6676
2014	23	196	6872
2014	24	229	7101
2014	25	207	7308
2014	26	201	7509
2014	27	222	7731
2014	28	215	7946
2014	29	221	8167
2014	30	238	8405
2014	31	193	8598
2014	32	245	8843
2014	33	261	9104
2014	34	259	9363
2014	35	18	9381

CASE STUDY 2: INVESTIGATING METRIC SPIKE

WEEKLY RETENTION ANALYSIS: ANALYZE THE RETENTION OF USERS ON A WEEKLYBASIS AFTER SIGNING UP FOR A PRODUCT.

Task C: Write an SQL query to calculate the weekly retention of users based on their sign-up cohort.

Query:

```
Select
distinct user_id,
count(user_id) as num_users,
sum(case when retention_week = 1 then 1 else 0 end) as per_week_retention
from(
select
a.user_id,
a.signup_week,
b.engagement_week,
b.engagement_week - a.signup_week as retention_week
from(
(select distinct user_id, extract(week from occurred_at) as signup_week from events
where event_type = 'signup_flow’
and event_name = 'complete_signup’)a
left join (select distinct user_id, extract(week from occurred_at) as engagement_week
from events
where event_type = 'engagement’
)b
on a.user_id = b.user_id)
)d
group by user_id
order by user_id;
```

CASE STUDY 2: INVESTIGATING METRIC SPIKE

WEEKLY RETENTION ANALYSIS: ANALYZE THE RETENTION OF USERS ON A WEEKLYBASIS AFTER SIGNING UP FOR A PRODUCT.

Output:

https://drive.google.com/file/d/1MxqCv8-lCOCaCO2aYz9Eu8cI0PFFJTyC/view?usp=sharing

CASE STUDY 2: INVESTIGATING METRIC SPIKE

WEEKLY ENGAGEMENT PER DEVICE: MEASURE THE ACTIVENESS OF USERS ON AWEEKLY BASIS PER DEVICE.

Task D: Write an SQL query to calculate the weekly engagement per device.

Query:

```
select year(occurred_at) as year,

week(occurred_at) as no_of_weeks,

device,

count(distinct user_id) as no_of_user

from events

where event_type='engagement’

group by 1,2,3

order by 1,2,3;
```

CASE STUDY 2: INVESTIGATING METRIC SPIKE

WEEKLY ENGAGEMENT PER DEVICE: MEASURE THE ACTIVENESS OF USERS ON AWEEKLY BASIS PER DEVICE.

Output:

https://drive.google.com/file/d/19odBL7SQ0mkWUm_SCxvsWt9vwDfx9yHw/view?usp=sharing

CASE STUDY 2: INVESTIGATING METRIC SPIKE

EMAIL ENGAGEMENT ANALYSIS: ANALYZE HOW USERS ARE ENGAGING WITH THEEMAIL SERVICE.

Task E: Write an SQL query to calculate the email engagement metrics.

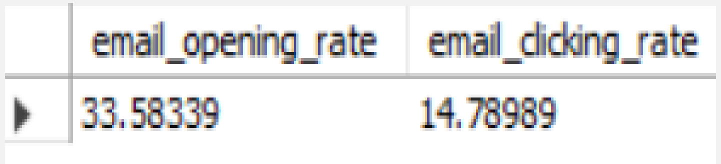
Query:

```
select
100.0 * sum(case when email_action = 'email_opened' then 1 else 0 end)
/sum(case when email_action = 'email_sent' then 1 else 0
end)as email_opening_rate,
100.0 * sum(case when email_action = 'email_clicked' then 1 else 0 end)
/sum(case when email_action = 'email_sent' then 1 else 0 end)
as email_clicking_rate
from(
select *,case when action in ('sent_weekly_digest', 'sent_reengagement_email')then 'email_sent' when action in
('email_open’)
then'email_opened' when action in
('email_clickthrough’) then’email_clicked’
end as email_action
from email_events
)a;
```

CASE STUDY 2: INVESTIGATING METRIC SPIKE

EMAIL ENGAGEMENT ANALYSIS: ANALYZE HOW USERS ARE ENGAGING WITH THEEMAIL SERVICE.

Output:



	email_opening_rate	email_clicking_rate
▶	33.58339	14.78989

THANK YOU