OPERATION ANALYTICS AND INVESTIGATING METRIC SPIKE

-ADVANCED SQL

PROJECT DESCRIPTION:

- The project is all about analyzing company operations to gain insights that can drive operational improvements and enhance decision-making.

- This project focuses on two areas of analysis:
  A) **Job Data Analysis**.
  B) **Investigating Metric Spikes**.

- The main purpose of analyzing operational data is to understand performance trends,detect sudden changes in key metrics and improve overall business.

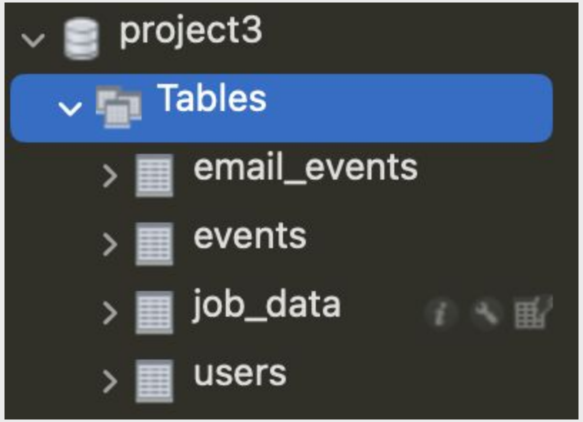
TECH-STACK USED:

- **SOFTWARE: MySQL Workbench 8.0.40**
  - I used MySQl Workbench software because it can run multiple queries at a time compared to mysql server and also It is easy to use and extract the queries in detailed way.

- **OPERATING SYSTEM: MacOS**
  - My device is MacOs.I have used it as it is stable and efficient

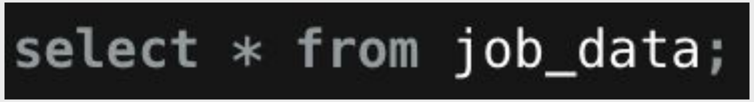
APPROACH:

- To do the project,The team has given us 1 dataset for "Case Study-1" and 3 datasets for "Case Study-2".The files are csv files.
- First I created a database with name:**Project3** then I have loaded the tables using **"Table data import wizard".** and adjusted column datatypes and their names.



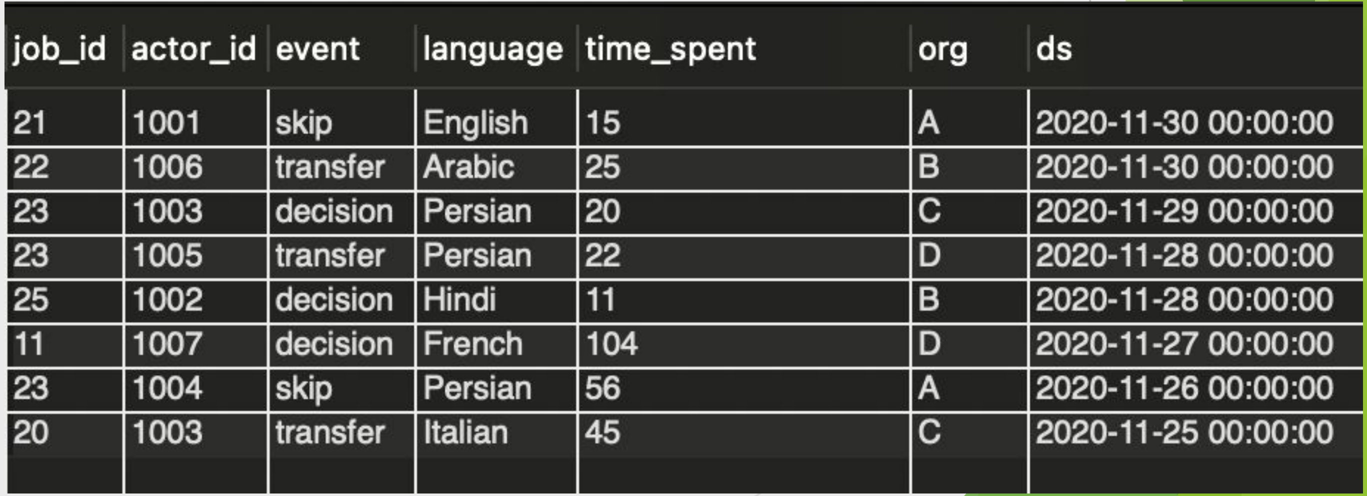
CASE STUDY 1: JOB DATA ANALYSIS

In case study 1, We are given with 1 dataset named job_data.The table structure is:



```
select * from job_data;
```

TABLE STRUCTURE:



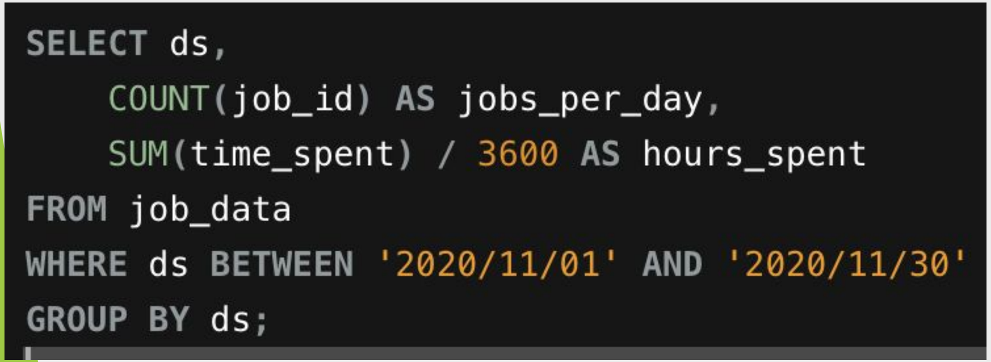
job_id	actor_id	event	language	time_spent	org	ds
21	1001	skip	English	15	A	2020-11-30 00:00:00
22	1006	transfer	Arabic	25	B	2020-11-30 00:00:00
23	1003	decision	Persian	20	C	2020-11-29 00:00:00
23	1005	transfer	Persian	22	D	2020-11-28 00:00:00
25	1002	decision	Hindi	11	B	2020-11-28 00:00:00
11	1007	decision	French	104	D	2020-11-27 00:00:00
23	1004	skip	Persian	56	A	2020-11-26 00:00:00
20	1003	transfer	Italian	45	C	2020-11-25 00:00:00

CASE STUDY 1: JOB DATA ANALYSIS

A.Jobs Reviewed Over Time: Calculate the number of jobs reviewed per hour for each day in November 2020.

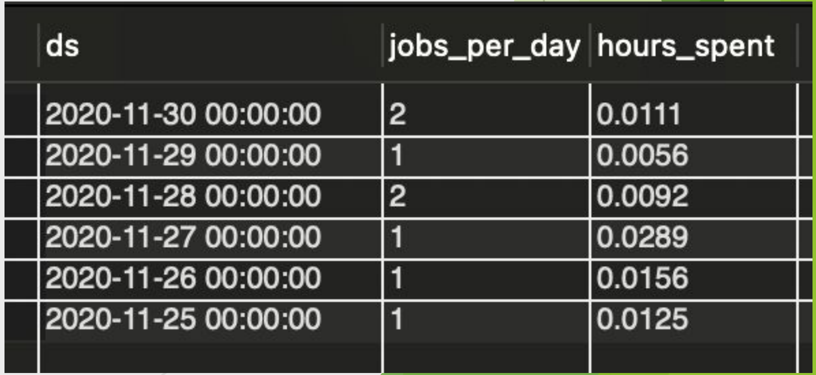
To solve the given task,I have used count(),sum() aggregate functions.As the question is jobs reviewed per hour,I have divided count(jobs) by sum(timespent) in seconds and used the condition between 1 nov to 31 nov 2020.

QUERY:



```
SELECT ds,
    COUNT(job_id) AS jobs_per_day,
    SUM(time_spent) / 3600 AS hours_spent
FROM job_data
WHERE ds BETWEEN '2020/11/01' AND '2020/11/30'
GROUP BY ds;
```

OUTPUT:

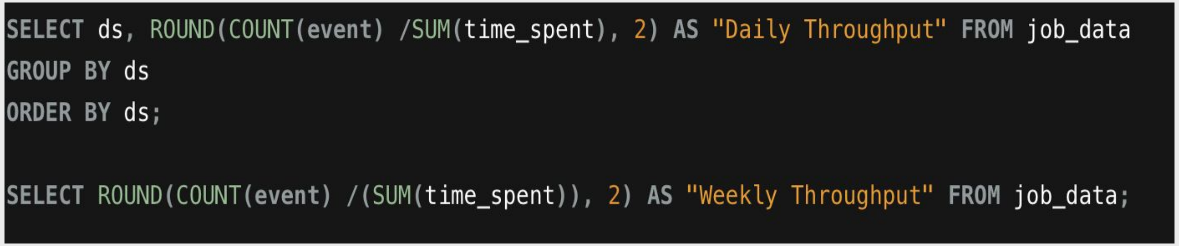


ds	jobs_per_day	hours_spent
2020-11-30 00:00:00	2	0.0111
2020-11-29 00:00:00	1	0.0056
2020-11-28 00:00:00	2	0.0092
2020-11-27 00:00:00	1	0.0289
2020-11-26 00:00:00	1	0.0156
2020-11-25 00:00:00	1	0.0125

B.Throughput Analysis:Calculate the 7-day rolling average of throughput(number of events per second).

The task was to find the 7-day rolling average of throughput and also daily metric.I would prefer the 7-day rolling average because it gives proper insights for the company future than the daily metric.

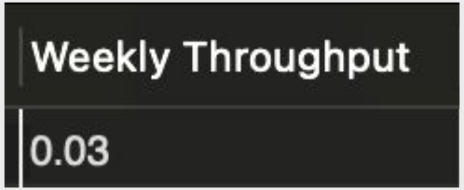
QUERY:



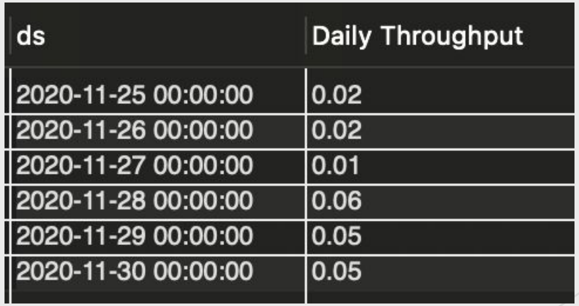
```
SELECT ds, ROUND(COUNT(event) /SUM(time_spent), 2) AS "Daily Throughput" FROM job_data
GROUP BY ds
ORDER BY ds;

SELECT ROUND(COUNT(event) /(SUM(time_spent)), 2) AS "Weekly Throughput" FROM job_data;
```

OUTPUT:



Weekly Throughput
0.03

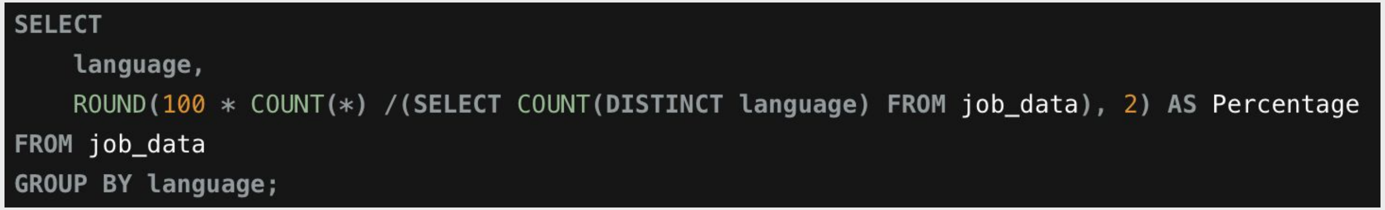


ds	Daily Throughput
2020-11-25 00:00:00	0.02
2020-11-26 00:00:00	0.02
2020-11-27 00:00:00	0.01
2020-11-28 00:00:00	0.06
2020-11-29 00:00:00	0.05
2020-11-30 00:00:00	0.05

C.Language Share Analysis: Calculate the percentage share of each language in the last 30 days.

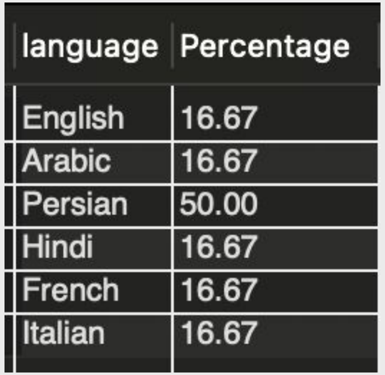
The Query is about calculating share of languages so I divided the language used by all languages used and multiplied with 100 to get the percentage of each language.

QUERY:



```
SELECT
    language,
    ROUND(100 * COUNT(*) /(SELECT COUNT(DISTINCT language) FROM job_data), 2) AS Percentage
FROM job_data
GROUP BY language;
```

OUTPUT:

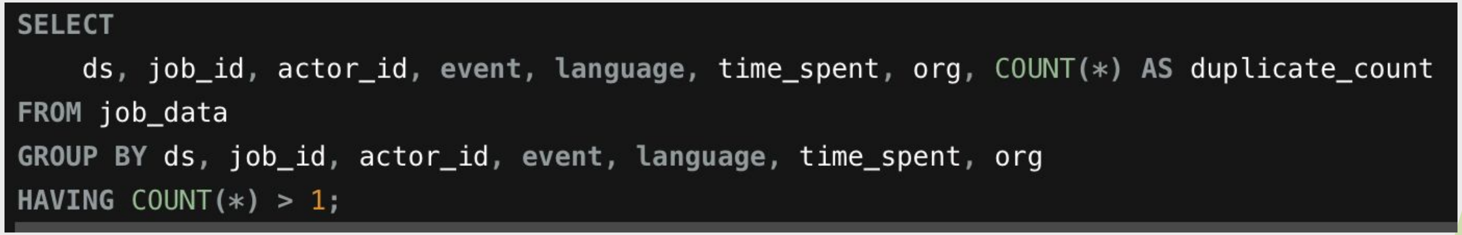


language	Percentage
English	16.67
Arabic	16.67
Persian	50.00
Hindi	16.67
French	16.67
Italian	16.67

D.Duplicate Rows Detection: Identify duplicate rows in the data.

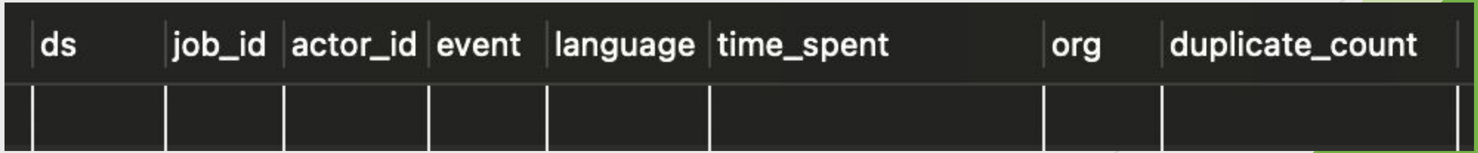
The task is to find the duplicate rows.we can find the duplicate rows by counting all columns from the table rowwise and checking the count()>1 for every individual row.If count() > 1 then duplicates exists,else no.The given table doesn’t consist any duplicate rows.

QUERY:



```
SELECT
    ds, job_id, actor_id, event, language, time_spent, org, COUNT(*) AS duplicate_count
FROM job_data
GROUP BY ds, job_id, actor_id, event, language, time_spent, org
HAVING COUNT(*) > 1;
```

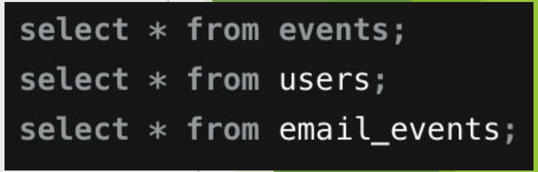
OUTPUT:



ds	job_id	actor_id	event	language	time_spent	org	duplicate_count

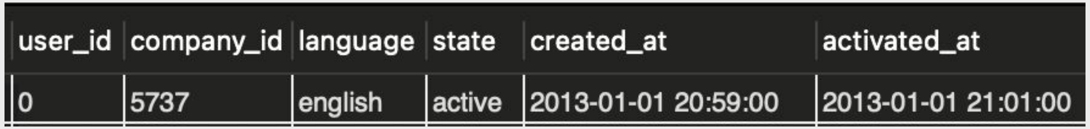
CASE STUDY-2:INVESTIGATING METRIC SPIKE:

In this Case Study,We are given with 3 datasets,they are users,events,email_events,The table structures are:



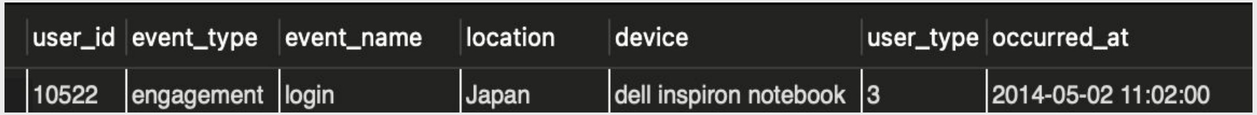
```
select * from events;
select * from users;
select * from email_events;
```

USERS:



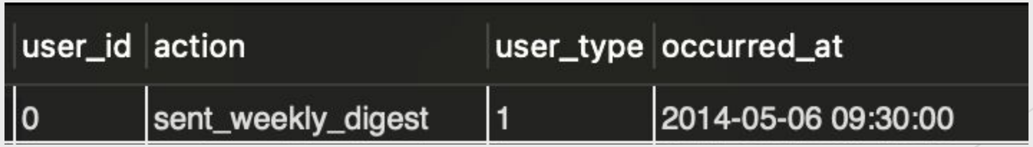
user_id	company_id	language	state	created_at	activated_at
0	5737	english	active	2013-01-01 20:59:00	2013-01-01 21:01:00

EVENTS:



user_id	event_type	event_name	location	device	user_type	occurred_at
10522	engagement	login	Japan	dell inspiron notebook	3	2014-05-02 11:02:00

EMAIL_EVENTS:



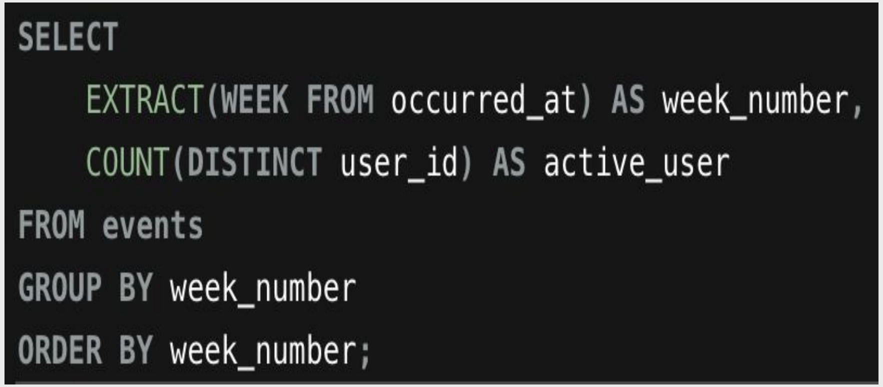
user_id	action	user_type	occurred_at
0	sent_weekly_digest	1	2014-05-06 09:30:00

CASE STUDY-2:INVESTIGATING METRIC SPIKE:

A.Weekly User Engagement:Measure the activeness of users on a weekly basis.

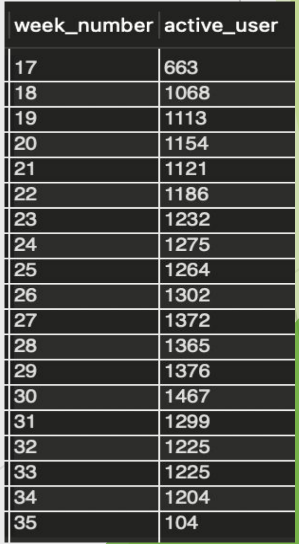
To solve the given task,I have used the extract(),count() aggregate function to find week number and users engaged in that week.

QUERY:



```
SELECT
    EXTRACT(WEEK FROM occurred_at) AS week_number,
    COUNT(DISTINCT user_id) AS active_user
FROM events
GROUP BY week_number
ORDER BY week_number;
```

OUTPUT:

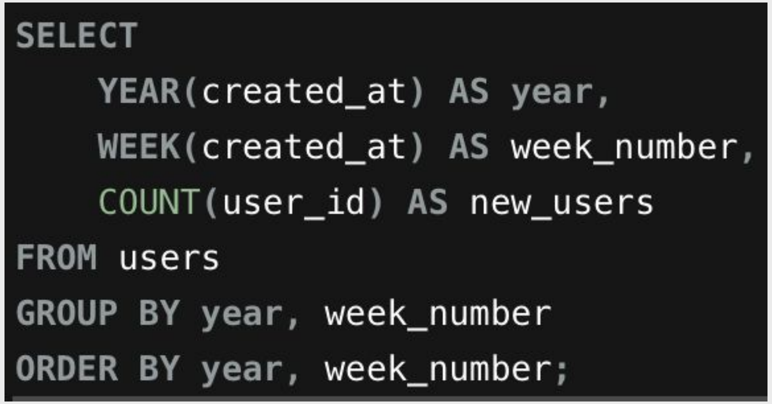


week_number	active_user
17	663
18	1068
19	1113
20	1154
21	1121
22	1186
23	1232
24	1275
25	1264
26	1302
27	1372
28	1365
29	1376
30	1467
31	1299
32	1225
33	1225
34	1204
35	104

B.User Growth Analysis: Analyze the growth of users over time for a product.

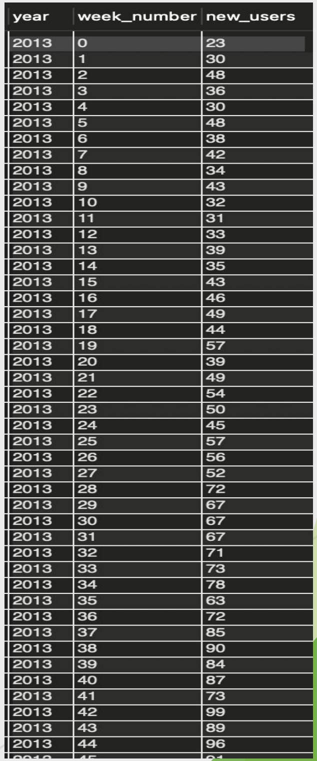
The task is to find the user growth for the product,so I have calculated weekly and yearly wise user growth.

QUERY:

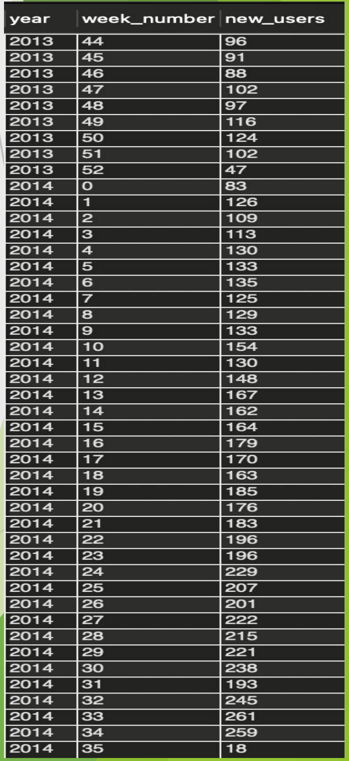


```
SELECT
    YEAR(created_at) AS year,
    WEEK(created_at) AS week_number,
    COUNT(user_id) AS new_users
FROM users
GROUP BY year, week_number
ORDER BY year, week_number;
```

OUTPUT:



year	week_number	new_users
2013	0	23
2013	1	30
2013	2	48
2013	3	36
2013	4	30
2013	5	48
2013	6	38
2013	7	42
2013	8	34
2013	9	43
2013	10	32
2013	11	31
2013	12	33
2013	13	39
2013	14	35
2013	15	43
2013	16	46
2013	17	49
2013	18	44
2013	19	57
2013	20	39
2013	21	49
2013	22	54
2013	23	50
2013	24	45
2013	25	57
2013	26	56
2013	27	52
2013	28	72
2013	29	67
2013	30	67
2013	31	67
2013	32	71
2013	33	73
2013	34	78
2013	35	63
2013	36	72
2013	37	85
2013	38	90
2013	39	84
2013	40	87
2013	41	73
2013	42	99
2013	43	89
2013	44	96

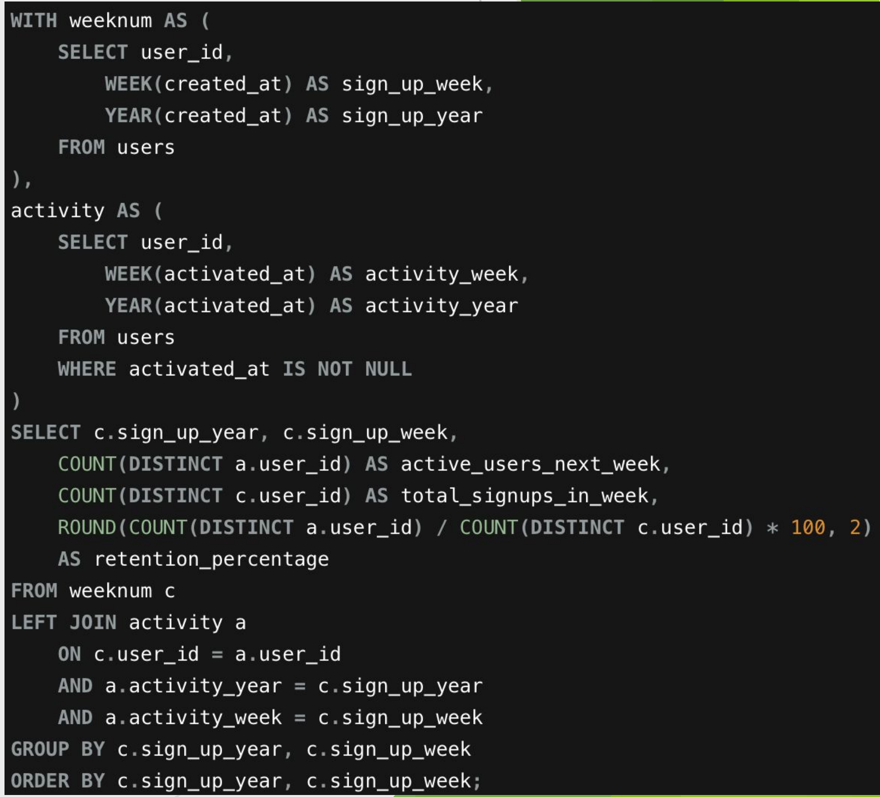


year	week_number	new_users
2013	44	96
2013	45	91
2013	46	88
2013	47	102
2013	48	97
2013	49	116
2013	50	124
2013	51	102
2013	52	47
2014	0	83
2014	1	126
2014	2	109
2014	3	113
2014	4	130
2014	5	133
2014	6	135
2014	7	125
2014	8	129
2014	9	133
2014	10	154
2014	11	130
2014	12	148
2014	13	167
2014	14	162
2014	15	164
2014	16	179
2014	17	170
2014	18	163
2014	19	185
2014	20	176
2014	21	183
2014	22	196
2014	23	196
2014	24	229
2014	25	207
2014	26	201
2014	27	222
2014	28	215
2014	29	221
2014	30	238
2014	31	193
2014	32	245
2014	33	261
2014	34	259
2014	35	18

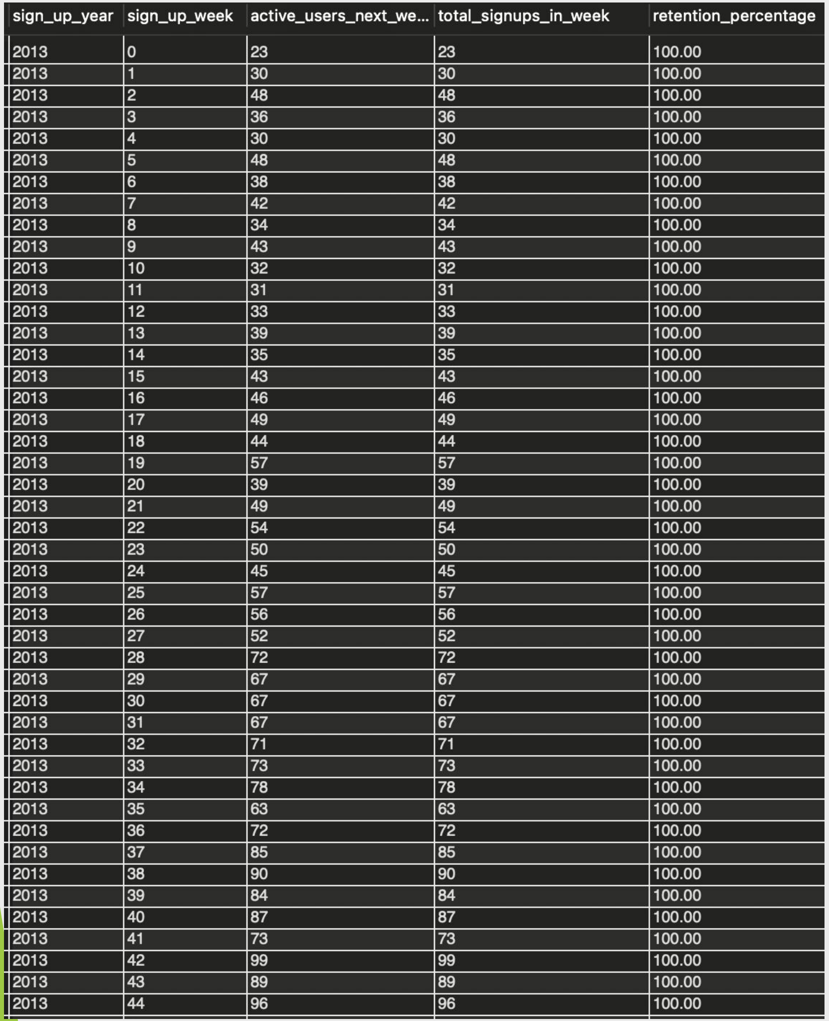
C.Weekly Retention Analysis: Analyze the retention of users on a weekly basis after signing up for a product.

The task is to find out how long users are active after signing up for the product.so I have calculated the week of signing then calculated the week of activated_at.then joined them to find how long users are active.

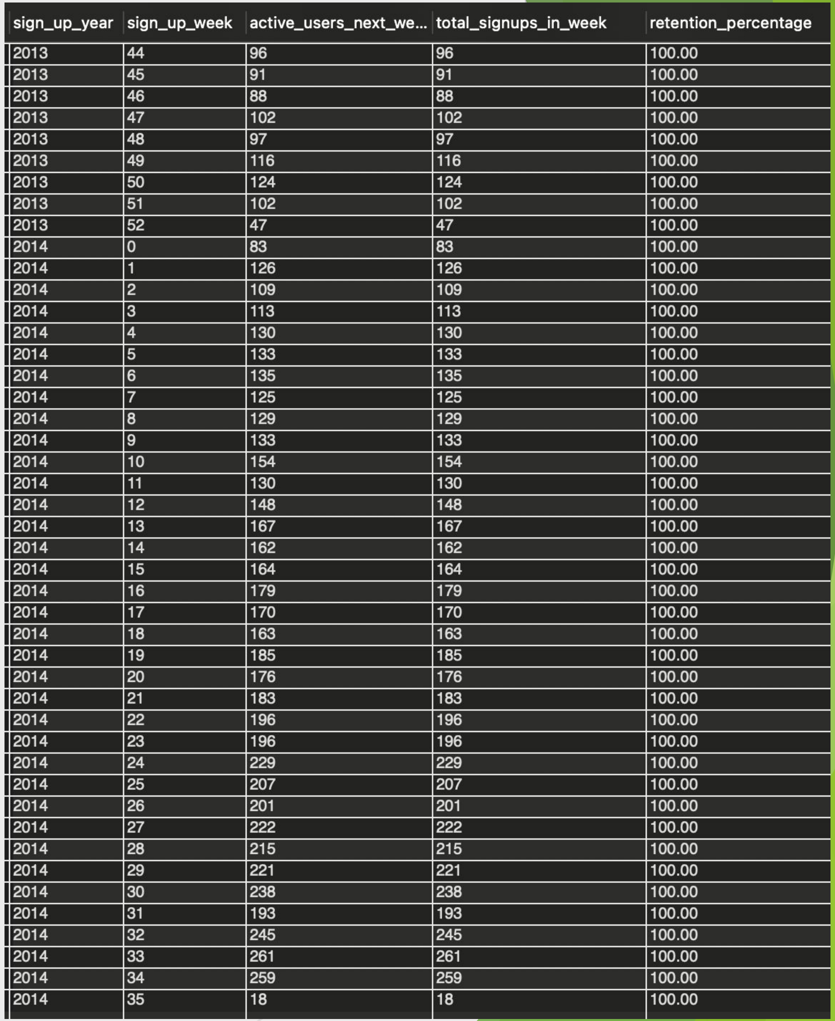
QUERY:



```
WITH weeknum AS (
    SELECT user_id,
        WEEK(created_at) AS sign_up_week,
        YEAR(created_at) AS sign_up_year
    FROM users
),
activity AS (
    SELECT user_id,
        WEEK(activated_at) AS activity_week,
        YEAR(activated_at) AS activity_year
    FROM users
    WHERE activated_at IS NOT NULL
)
SELECT c.sign_up_year, c.sign_up_week,
    COUNT(DISTINCT a.user_id) AS active_users_next_week,
    COUNT(DISTINCT c.user_id) AS total_signups_in_week,
    ROUND(COUNT(DISTINCT a.user_id) / COUNT(DISTINCT c.user_id) * 100, 2)
    AS retention_percentage
FROM weeknum c
LEFT JOIN activity a
    ON c.user_id = a.user_id
    AND a.activity_year = c.sign_up_year
    AND a.activity_week = c.sign_up_week
GROUP BY c.sign_up_year, c.sign_up_week
ORDER BY c.sign_up_year, c.sign_up_week;
```



sign_up_year	sign_up_week	active_users_next_we...	total_signups_in_week	retention_percentage
2013	0	23	23	100.00
2013	1	30	30	100.00
2013	2	48	48	100.00
2013	3	36	36	100.00
2013	4	30	30	100.00
2013	5	48	48	100.00
2013	6	38	38	100.00
2013	7	42	42	100.00
2013	8	34	34	100.00
2013	9	43	43	100.00
2013	10	32	32	100.00
2013	11	31	31	100.00
2013	12	33	33	100.00
2013	13	39	39	100.00
2013	14	35	35	100.00
2013	15	43	43	100.00
2013	16	46	46	100.00
2013	17	49	49	100.00
2013	18	44	44	100.00
2013	19	57	57	100.00
2013	20	39	39	100.00
2013	21	49	49	100.00
2013	22	54	54	100.00
2013	23	50	50	100.00
2013	24	45	45	100.00
2013	25	57	57	100.00
2013	26	56	56	100.00
2013	27	52	52	100.00
2013	28	72	72	100.00
2013	29	67	67	100.00
2013	30	67	67	100.00
2013	31	67	67	100.00
2013	32	71	71	100.00
2013	33	73	73	100.00
2013	34	78	78	100.00
2013	35	63	63	100.00
2013	36	72	72	100.00
2013	37	85	85	100.00
2013	38	90	90	100.00
2013	39	84	84	100.00
2013	40	87	87	100.00
2013	41	73	73	100.00
2013	42	99	99	100.00
2013	43	89	89	100.00
2013	44	96	96	100.00

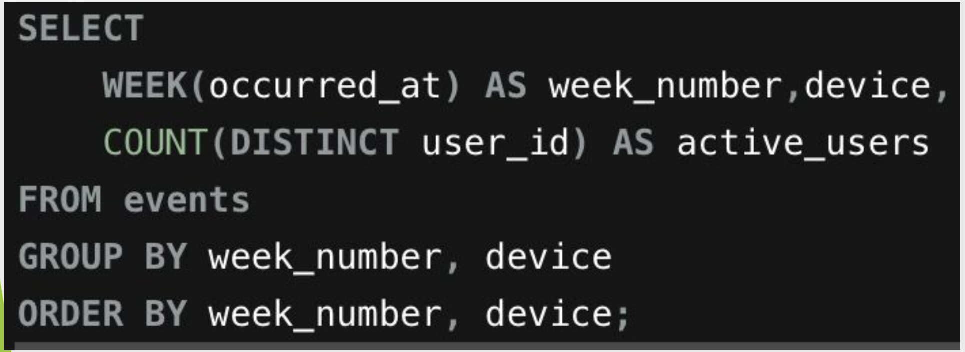


sign_up_year	sign_up_week	active_users_next_we...	total_signups_in_week	retention_percentage
2013	44	96	96	100.00
2013	45	91	91	100.00
2013	46	88	88	100.00
2013	47	102	102	100.00
2013	48	97	97	100.00
2013	49	116	116	100.00
2013	50	124	124	100.00
2013	51	102	102	100.00
2013	52	47	47	100.00
2014	0	83	83	100.00
2014	1	126	126	100.00
2014	2	109	109	100.00
2014	3	113	113	100.00
2014	4	130	130	100.00
2014	5	133	133	100.00
2014	6	135	135	100.00
2014	7	125	125	100.00
2014	8	129	129	100.00
2014	9	133	133	100.00
2014	10	154	154	100.00
2014	11	130	130	100.00
2014	12	148	148	100.00
2014	13	167	167	100.00
2014	14	162	162	100.00
2014	15	164	164	100.00
2014	16	179	179	100.00
2014	17	170	170	100.00
2014	18	163	163	100.00
2014	19	185	185	100.00
2014	20	176	176	100.00
2014	21	183	183	100.00
2014	22	196	196	100.00
2014	23	196	196	100.00
2014	24	229	229	100.00
2014	25	207	207	100.00
2014	26	201	201	100.00
2014	27	222	222	100.00
2014	28	215	215	100.00
2014	29	221	221	100.00
2014	30	238	238	100.00
2014	31	193	193	100.00
2014	32	245	245	100.00
2014	33	261	261	100.00
2014	34	259	259	100.00
2014	35	18	18	100.00

D.Weekly Engagement per Device: Measure the activeness of users on a weekly basis per device.

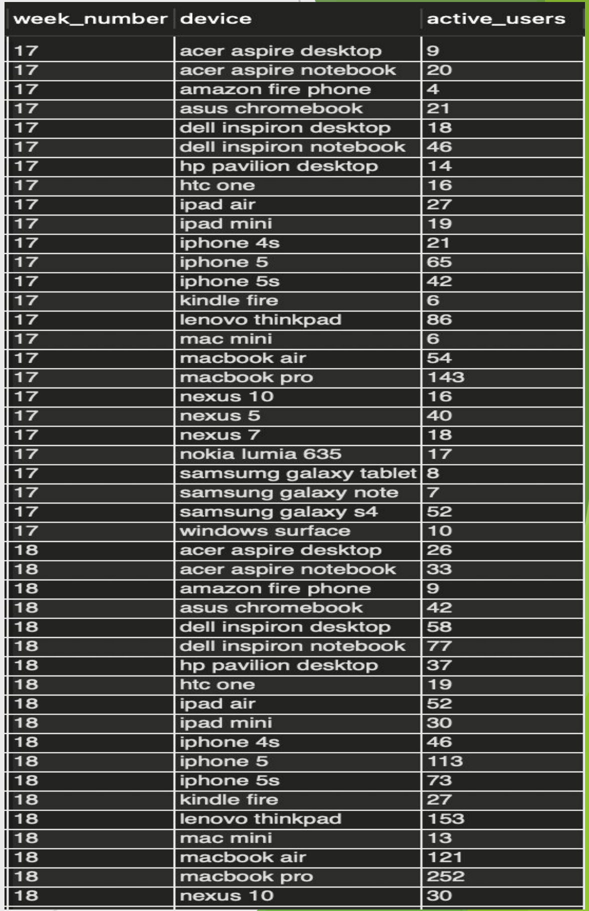
The given task is to find how users engaged with the product per device.like different laptops,mobiles.I have used the week() to generate week number and counted the users as per devices.

QUERY:

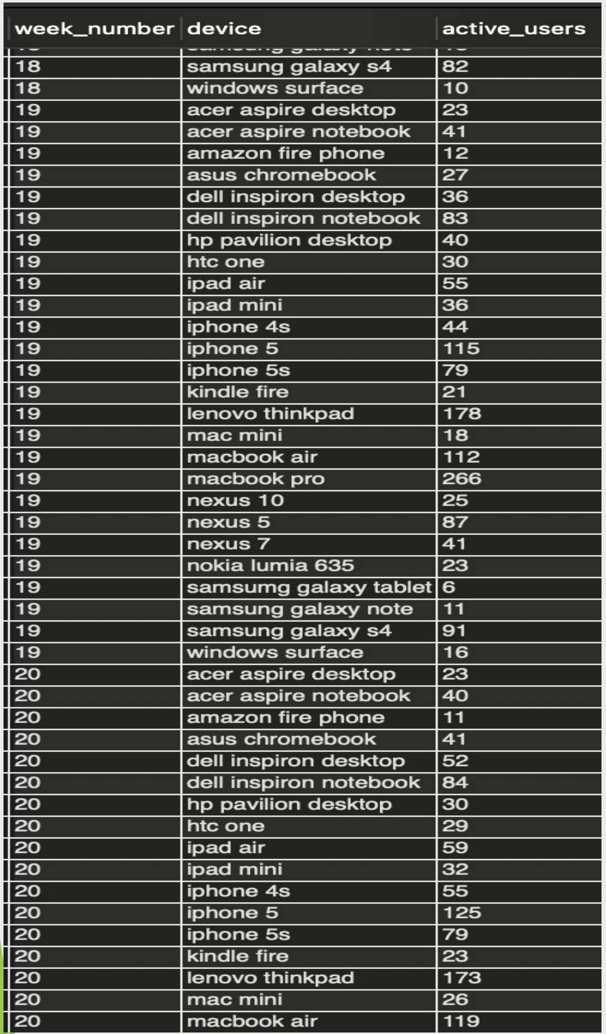


```
SELECT
    WEEK(occurred_at) AS week_number,device,
    COUNT(DISTINCT user_id) AS active_users
FROM events
GROUP BY week_number, device
ORDER BY week_number, device;
```

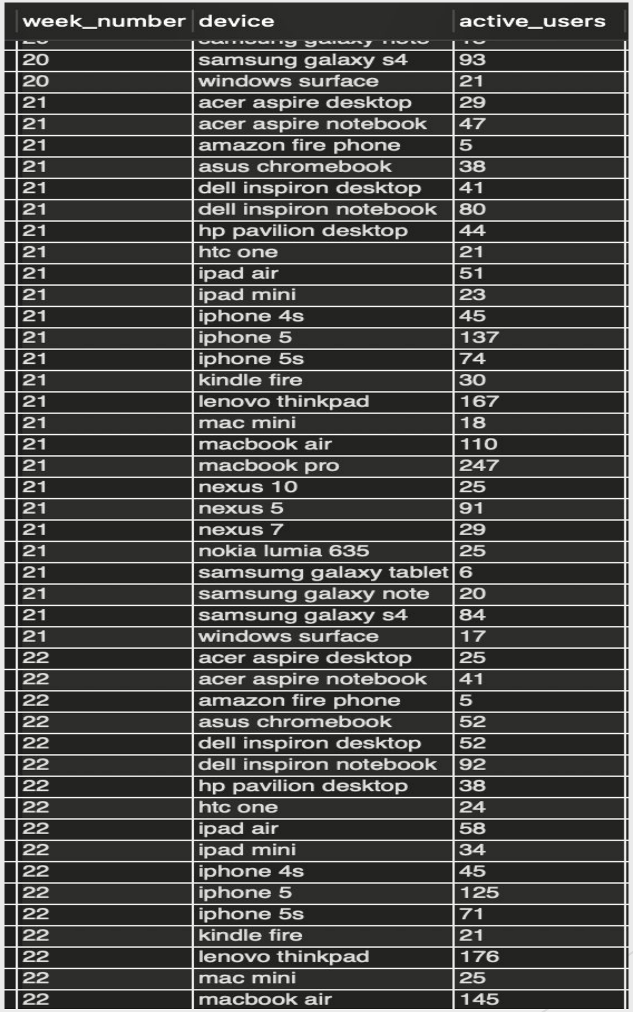
OUTPUT:



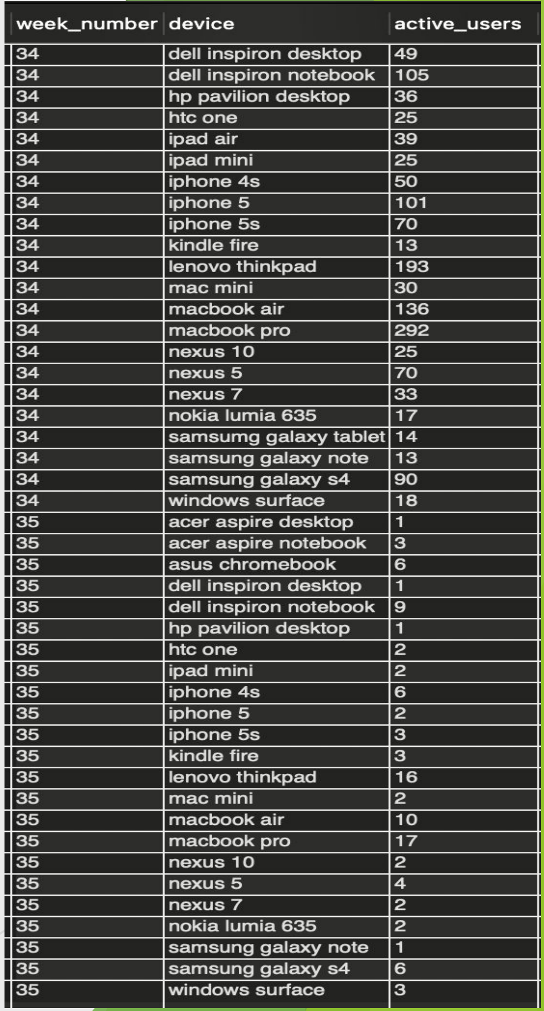
week_number	device	active_users
17	acer aspire desktop	9
17	acer aspire notebook	20
17	amazon fire phone	4
17	asus chromebook	21
17	dell inspiron desktop	18
17	dell inspiron notebook	46
17	hp pavilion desktop	14
17	htc one	16
17	ipad air	27
17	ipad mini	19
17	iphone 4s	21
17	iphone 5	65
17	iphone 5s	42
17	kindle fire	6
17	lenovo thinkpad	86
17	mac mini	6
17	macbook air	54
17	macbook pro	143
17	nexus 10	16
17	nexus 5	40
17	nexus 7	18
17	nokia lumia 635	17
17	samsumg galaxy tablet	8
17	samsung galaxy note	7
17	samsung galaxy s4	52
17	windows surface	10
18	acer aspire desktop	26
18	acer aspire notebook	33
18	amazon fire phone	9
18	asus chromebook	42
18	dell inspiron desktop	58
18	dell inspiron notebook	77
18	hp pavilion desktop	37
18	htc one	19
18	ipad air	52
18	ipad mini	30
18	iphone 4s	46
18	iphone 5	113
18	iphone 5s	73
18	kindle fire	27
18	lenovo thinkpad	153
18	mac mini	13
18	macbook air	121
18	macbook pro	252
18	nexus 10	30



week_number	device	active_users
18	samsung galaxy s4	82
18	windows surface	10
19	acer aspire desktop	23
19	acer aspire notebook	41
19	amazon fire phone	12
19	asus chromebook	27
19	dell inspiron desktop	36
19	dell inspiron notebook	83
19	hp pavilion desktop	40
19	htc one	30
19	ipad air	55
19	ipad mini	36
19	iphone 4s	44
19	iphone 5	115
19	iphone 5s	79
19	kindle fire	21
19	lenovo thinkpad	178
19	mac mini	18
19	macbook air	112
19	macbook pro	266
19	nexus 10	25
19	nexus 5	87
19	nexus 7	41
19	nokia lumia 635	23
19	samsumg galaxy tablet	6
19	samsung galaxy note	11
19	samsung galaxy s4	91
19	windows surface	16
20	acer aspire desktop	23
20	acer aspire notebook	40
20	amazon fire phone	11
20	asus chromebook	41
20	dell inspiron desktop	52
20	dell inspiron notebook	84
20	hp pavilion desktop	30
20	htc one	29
20	ipad air	59
20	ipad mini	32
20	iphone 4s	55
20	iphone 5	125
20	iphone 5s	79
20	kindle fire	23
20	lenovo thinkpad	173
20	mac mini	26
20	macbook air	119



week_number	device	active_users
20	samsung galaxy s4	93
20	windows surface	21
21	acer aspire desktop	29
21	acer aspire notebook	47
21	amazon fire phone	5
21	asus chromebook	38
21	dell inspiron desktop	41
21	dell inspiron notebook	80
21	hp pavilion desktop	44
21	htc one	21
21	ipad air	51
21	ipad mini	23
21	iphone 4s	45
21	iphone 5	137
21	iphone 5s	74
21	kindle fire	30
21	lenovo thinkpad	167
21	mac mini	18
21	macbook air	110
21	macbook pro	247
21	nexus 10	25
21	nexus 5	91
21	nexus 7	29
21	nokia lumia 635	25
21	samsumg galaxy tablet	6
21	samsung galaxy note	20
21	samsung galaxy s4	84
21	windows surface	17
22	acer aspire desktop	25
22	acer aspire notebook	41
22	amazon fire phone	5
22	asus chromebook	52
22	dell inspiron desktop	52
22	dell inspiron notebook	92
22	hp pavilion desktop	38
22	htc one	24
22	ipad air	58
22	ipad mini	34
22	iphone 4s	45
22	iphone 5	125
22	iphone 5s	71
22	kindle fire	21
22	lenovo thinkpad	176
22	mac mini	25
22	macbook air	145

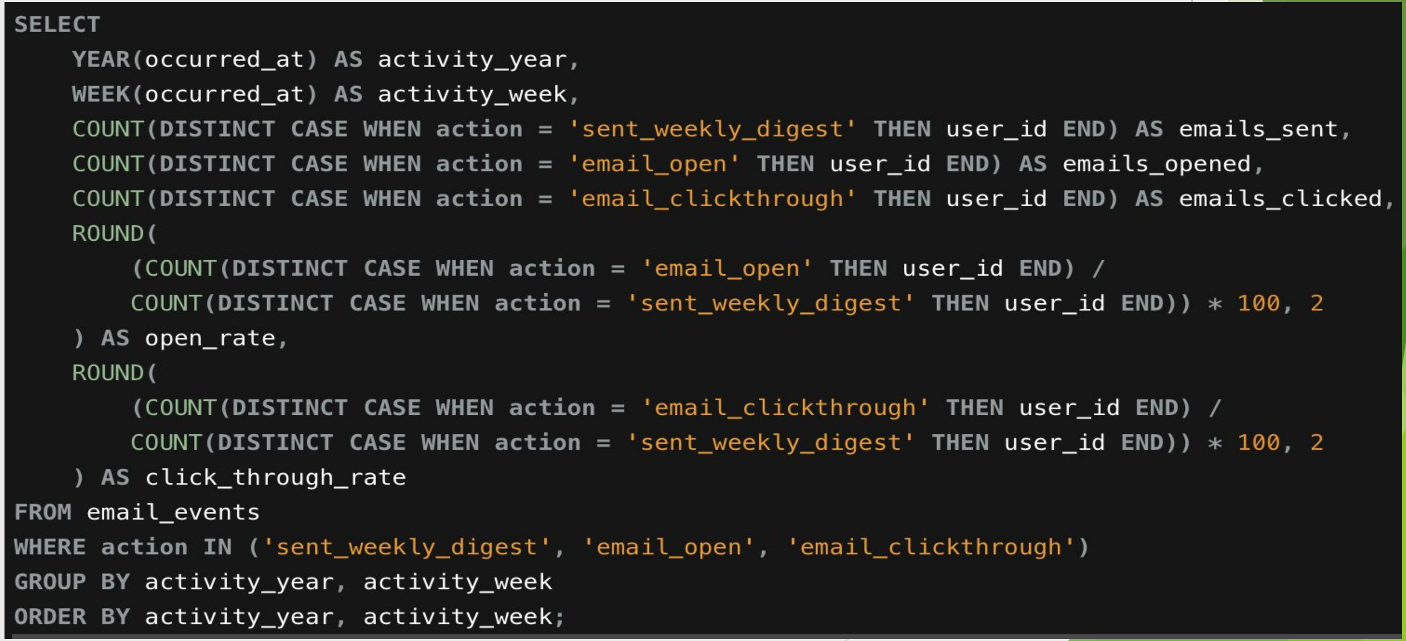


week_number	device	active_users
34	dell inspiron desktop	49
34	dell inspiron notebook	105
34	hp pavilion desktop	36
34	htc one	25
34	ipad air	39
34	ipad mini	25
34	iphone 4s	50
34	iphone 5	101
34	iphone 5s	70
34	kindle fire	13
34	lenovo thinkpad	193
34	mac mini	30
34	macbook air	136
34	macbook pro	292
34	nexus 10	25
34	nexus 5	70
34	nexus 7	33
34	nokia lumia 635	17
34	samsumg galaxy tablet	14
34	samsung galaxy note	13
34	samsung galaxy s4	90
34	windows surface	18
35	acer aspire desktop	1
35	acer aspire notebook	3
35	asus chromebook	6
35	dell inspiron desktop	1
35	dell inspiron notebook	9
35	hp pavilion desktop	1
35	htc one	2
35	ipad mini	2
35	iphone 4s	6
35	iphone 5	2
35	iphone 5s	3
35	kindle fire	3
35	lenovo thinkpad	16
35	mac mini	2
35	macbook air	10
35	macbook pro	17
35	nexus 10	2
35	nexus 5	4
35	nexus 7	2
35	nokia lumia 635	2
35	samsung galaxy note	1
35	samsung galaxy s4	6
35	windows surface	3

E.Email Engagement Analysis: Analyze how users are engaging with the email service.

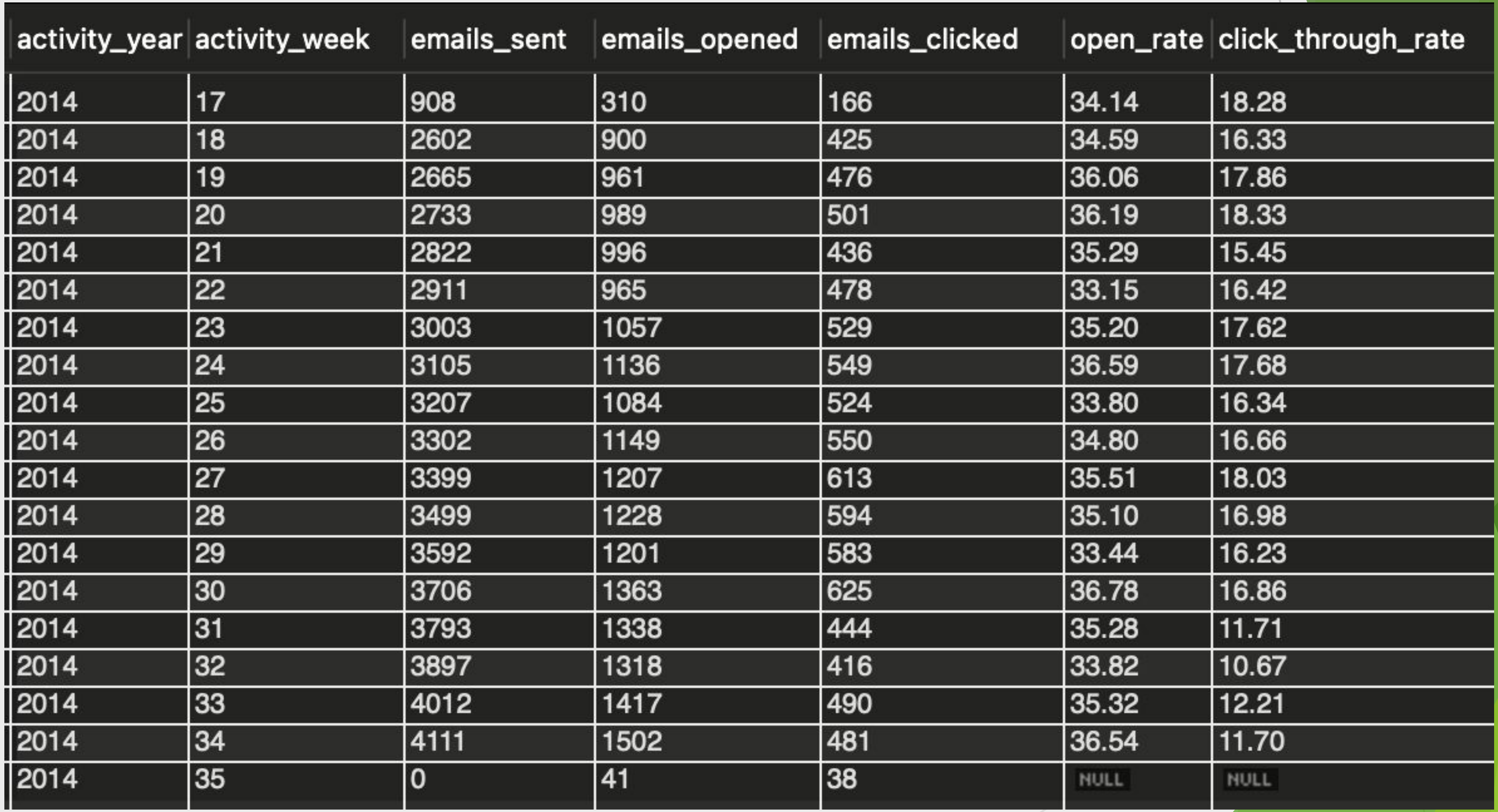
The task is to find how users are engaging with email services like email opening,email sending and email clicking.I found the count of email sent,opened and clicked and calculated the open rate and clickthrough rate.

QUERY:



```sql
SELECT
    YEAR(occurred_at) AS activity_year,
    WEEK(occurred_at) AS activity_week,
    COUNT(DISTINCT CASE WHEN action = 'sent_weekly_digest' THEN user_id END) AS emails_sent,
    COUNT(DISTINCT CASE WHEN action = 'email_open' THEN user_id END) AS emails_opened,
    COUNT(DISTINCT CASE WHEN action = 'email_clickthrough' THEN user_id END) AS emails_clicked,
    ROUND(
        (COUNT(DISTINCT CASE WHEN action = 'email_open' THEN user_id END) /
        COUNT(DISTINCT CASE WHEN action = 'sent_weekly_digest' THEN user_id END)) * 100, 2
    ) AS open_rate,
    ROUND(
        (COUNT(DISTINCT CASE WHEN action = 'email_clickthrough' THEN user_id END) /
        COUNT(DISTINCT CASE WHEN action = 'sent_weekly_digest' THEN user_id END)) * 100, 2
    ) AS click_through_rate
FROM email_events
WHERE action IN ('sent_weekly_digest', 'email_open', 'email_clickthrough')
GROUP BY activity_year, activity_week
ORDER BY activity_year, activity_week;
```

OUTPUT:



activity_year	activity_week	emails_sent	emails_opened	emails_clicked	open_rate	click_through_rate
2014	17	908	310	166	34.14	18.28
2014	18	2602	900	425	34.59	16.33
2014	19	2665	961	476	36.06	17.86
2014	20	2733	989	501	36.19	18.33
2014	21	2822	996	436	35.29	15.45
2014	22	2911	965	478	33.15	16.42
2014	23	3003	1057	529	35.20	17.62
2014	24	3105	1136	549	36.59	17.68
2014	25	3207	1084	524	33.80	16.34
2014	26	3302	1149	550	34.80	16.66
2014	27	3399	1207	613	35.51	18.03
2014	28	3499	1228	594	35.10	16.98
2014	29	3592	1201	583	33.44	16.23
2014	30	3706	1363	625	36.78	16.86
2014	31	3793	1338	444	35.28	11.71
2014	32	3897	1318	416	33.82	10.67
2014	33	4012	1417	490	35.32	12.21
2014	34	4111	1502	481	36.54	11.70
2014	35	0	41	38	NULL	NULL

INSIGHTS:

- **CASE STUDY-1:Job Data Analysis**

A. Jobs Reviewed Over Time: I identified the hours when the highest number of jobs were reviewed.

B. Throughput Analysis:I calculated the 7 day rolling average to understand the changes in job processing speed.

C. Language Share Analysis:I found the percentage share of all languages used in job reviewing.

D. Duplicate Rows Detection:I learned how to find the duplicate rows in the dataset to decrease the data redundancy.

INSIGHTS:

- **CASE STUDY-2:Investigating Metric Spikes**

A. Weekly User Engagement:I measured the activeness of user on weekly basis.

B. User Growth Analysis:I found out how many new users are registered per year and per week.

C. Weekly Retention Analysis:I calculated how long users stay active after signing up.

D. Weekly Engagement Per Device:I measured how users are engaged through different devices.

E. Email Engagement Analysis:I analyzed email engagement to determine how users are using the servies.

RESULT:

By the Project,I have analyzed Job data,User engagement metrics and extracted the actionable insights for given questions.

I Identified areas of operational improvement.and also improved my advanced sql skills, learned new functions.

This project had good impact on improving decision making for operations,marketing,and support teams.

THANK YOU