Name : DEBANJAN BHATTACHARJEE

Email id : debanjanbhattacharya2015@gmail.com

Data analytics

Project :Operation Analytics and Investigating Metric Spike

Analysis done on the following points:-

Case Study 1 : Job Data

A. Number of jobs reviewed: Amount of jobs reviewed over time.

Our task: Calculate the number of jobs reviewed per hour per day for November 2020?

B. Throughput: It is the no. of events happening per second.

**Our task: Let's say the above metric is called throughput. Calculate 7 day rolling average of throughput? For throughput, do we prefer daily metric or
7-day rolling and why?**

C. Percentage share of each language: Share of each language for different contents.

Our task: Calculate the percentage share of each language in the last 30 days?

D. Duplicate rows: Rows that have the same value present in them.

Our task: Let's say you see some duplicate rows in the data. How will we display duplicates from the table?

Case Study 2:
Investigating metric spike

A. User Engagement: To measure the activeness of a user. Measuring if the user finds quality in a product/service.

Our task: Calculate the weekly user engagement?

B. User Growth: Amount of users growing over time for a product.

Our task: Calculate the user growth for product?

C. Weekly Retention: Users getting retained weekly after signing-up for a product.

Our task: Calculate the weekly retention of users-sign up cohort?

D. Weekly Engagement: To measure the activeness of a user. Measuring if the user finds quality in a product/service weekly.

Our task: Calculate the weekly engagement per device?

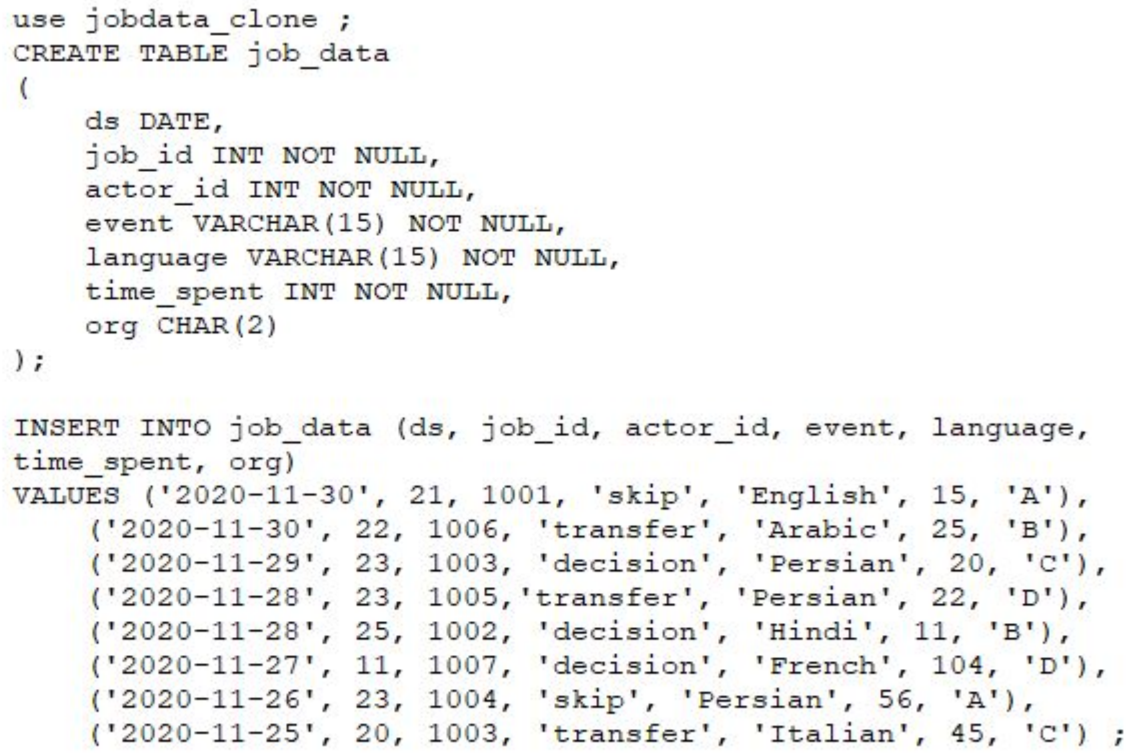
E. Email Engagement: Users engaging with the email service.

Our task: Calculate the email engagement metrics?

CASE STUDY : 1 (JOB DATA)

CREATE THE DATA BASE AND LOAD THE DATA

TO FULLFILL THIS STEP WE HAVE TO WRITE DOWN THE FOLLOWING QUERIES AND EXECUTE THIS.



```
use jobdata_clone ;
CREATE TABLE job_data
(
     ds DATE,
     job_id INT NOT NULL,
     actor_id INT NOT NULL,
     event VARCHAR(15) NOT NULL,
     language VARCHAR(15) NOT NULL,
     time_spent INT NOT NULL,
     org CHAR(2)
);

INSERT INTO job_data (ds, job_id, actor_id, event, language,
time_spent, org)
VALUES ('2020-11-30', 21, 1001, 'skip', 'English', 15, 'A'),
     ('2020-11-30', 22, 1006, 'transfer', 'Arabic', 25, 'B'),
     ('2020-11-29', 23, 1003, 'decision', 'Persian', 20, 'C'),
     ('2020-11-28', 23, 1005,'transfer', 'Persian', 22, 'D'),
     ('2020-11-28', 25, 1002, 'decision', 'Hindi', 11, 'B'),
     ('2020-11-27', 11, 1007, 'decision', 'French', 104, 'D'),
     ('2020-11-26', 23, 1004, 'skip', 'Persian', 56, 'A'),
     ('2020-11-25', 20, 1003, 'transfer', 'Italian', 45, 'C') ;
```

As the database has created now, We have to execute the following tasks:

TASK 1 : Number of jobs reviewed

Amount of jobs reviewed over time. Calculate the number of jobs reviewed per hour per day for November 2020?

STEPS THAT HAVE TO FOLLOWED

To find the number of jobs reviewed per hour per day of November 2020:

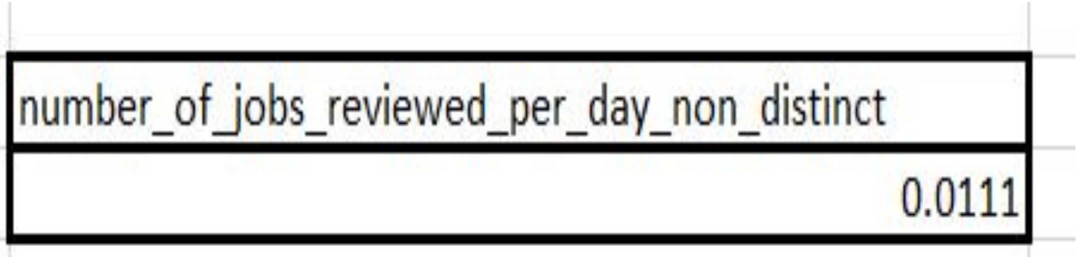
1. We have to use the data from **job_id** columns of the **job_data** table.

2. Then we have to divide the total count of **job_id** (distinct and non-distinct) by **(30 days * 24 hours)**for finding the number of jobs reviewed per day.

QUERIES (Non_distinct _job_id)

```
SELECT
count(job_id)/(30*24) AS
number_of_jobs_reviewed_per_day_non_distinct FROM job_data;
```

RESULTS



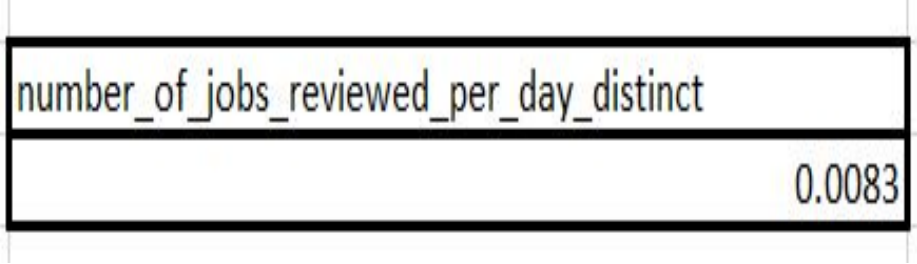
number_of_jobs_reviewed_per_day_non_distinct
0.0111

SO, the number of jobs reviewed per hour per day for November 2020 is 0.0111 for non_distinct_job_id.

QUERIES
Distinct_job_id

```
SELECT
 count(DISTINCT job_id)/(30*24) AS
number_of_jobs_reviewed_per_day_distinct
FROM job_data;
```

RESULTS



number_of_jobs_reviewed_per_day_distinct
0.0083

SO, the number of jobs reviewed per hour per day for November 2020 is 0.0083 for Distinct_job_id.

TASK : 2 THROUGHPUT

It is the no. of events happening per second.

OUR TASK :

Let’s say the above metric is called throughput.
Calculate 7 day rolling average of throughput?
For throughput, do you prefer daily metric or 7-day rolling and why?

STEPS THAT HAVE TO FOLLOWED

1. Firstly ,We have to **SELECT** the count of job_id(distinct and non-distinct) and ordering them as respect of ds (date of interview).

2. Then by using the **ROW** function we will be considering the rows .

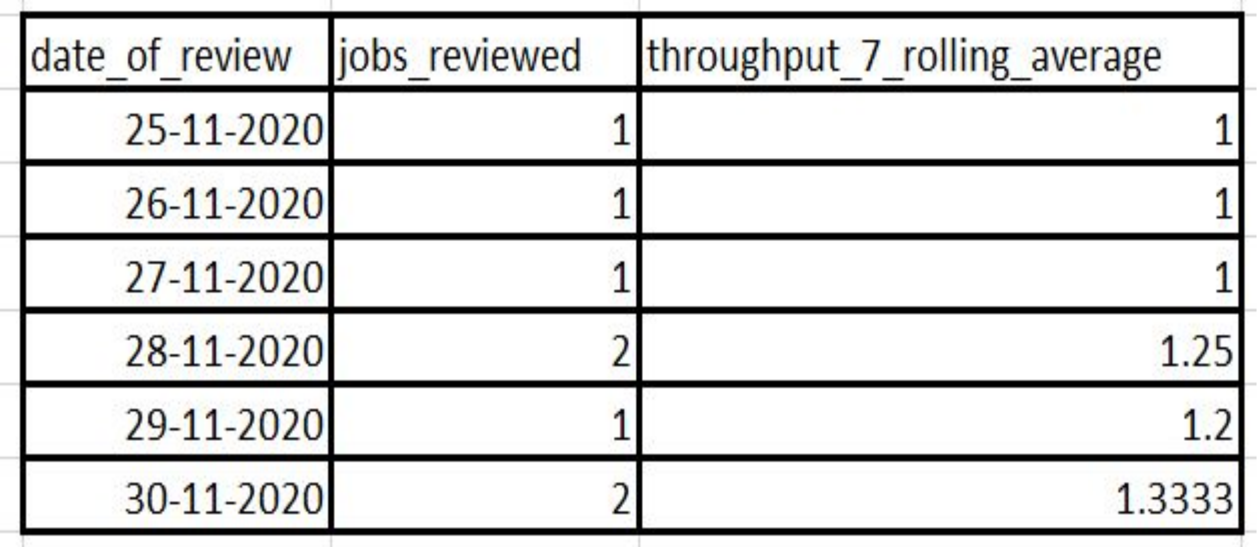
3. Then we will be taking the average of the jobs_reviewed .

Query
(distinct_job_id)

```
SELECT ds AS  date_of_review,
jobs_reviewed,  AVG(jobs_reviewed)

 OVER(ORDER BY ds ROWS BETWEEN 6
PRECEDING AND CURRENT ROW) AS
throughput_7_rolling_average
FROM (
SELECT ds, COUNT( DISTINCT job_id) AS
jobs_reviewed
 FROM  job_data
 GROUP BY ds
ORDER BY ds ) a;
```

RESULTS

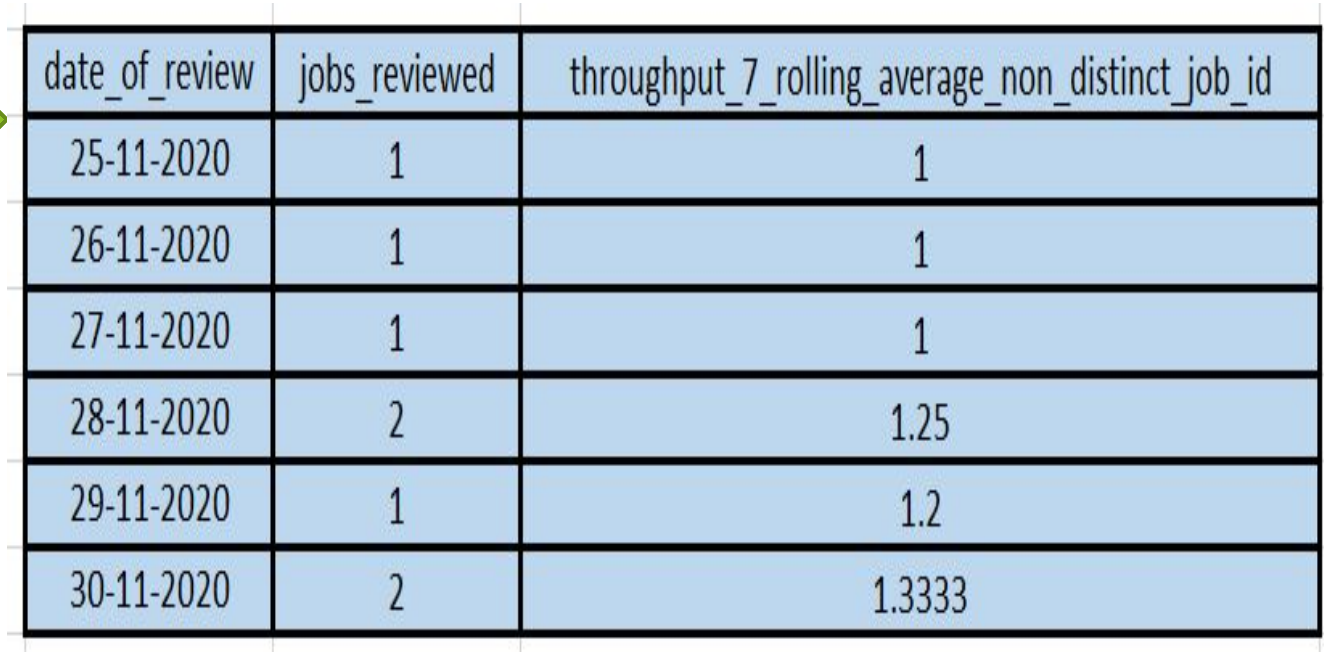


date_of_review	jobs_reviewed	throughput_7_rolling_average
25-11-2020	1	1
26-11-2020	1	1
27-11-2020	1	1
28-11-2020	2	1.25
29-11-2020	1	1.2
30-11-2020	2	1.3333

QUERIES
non_distinct_job_id

```
SELECT ds as date_of_review, jobs_reviewed,
AVG(jobs_reviewed)
OVER(ORDER BY ds ROWS BETWEEN 6 PRECEDING AND CURRENT
ROW) AS
throughput_7_rolling_average_non_distinct_job_id
FROM
(
SELECT ds, count(job_id) AS jobs_reviewed
FROM job_data
GROUP BY ds ORDER BY ds
) a;
```

RESULTS



date_of_review	jobs_reviewed	throughput_7_rolling_average_non_distinct_job_id
25-11-2020	1	1
26-11-2020	1	1
27-11-2020	1	1
28-11-2020	2	1.25
29-11-2020	1	1.2
30-11-2020	2	1.3333

SO,IN THE ABOVE TWO TABLES IT SHOWS ABOUT THE THROUGHPUT IN THE BOTH DISTINCT_JOB_ID AND NON_DISTINCT_JOB_ID.

TASK : 3 Percentage share of each language

Share Percentage of each language for different contents.

STEPS THAT HAVE TO FOLLOWED

We have to find the percentage of each language.
To avail the values for both distinct and non-distinct,

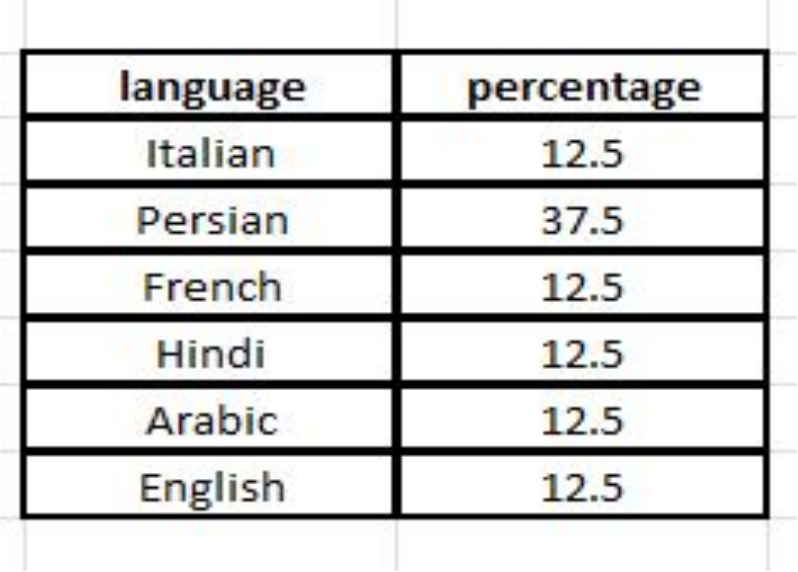
1.We have to divide the total number of languages(distinct and non-distinct)by the total number of rows in the tables

2. Then we have to use GROUP BY function on the languages.

QUERIES

```
with a as
(select max(ds)  as m from job_data)
select distinct
language,
(count(event) over(partition by language
rows between unbounded preceding and
unbounded following) /count(*)
over(order by ds rows between
unbounded preceding and unbounded
following) ) * 100 as percentage
 from
(select *
From
job_data cross join a
Where
datediff(m,date(ds)) between 0 and
30)a1;
```

RESULTS



language	percentage
Italian	12.5
Persian	37.5
French	12.5
Hindi	12.5
Arabic	12.5
English	12.5

TASK - 4

Duplicate rows

Rows that have the same value present in them.

STEPS THAT HAVE TO FOLLOW

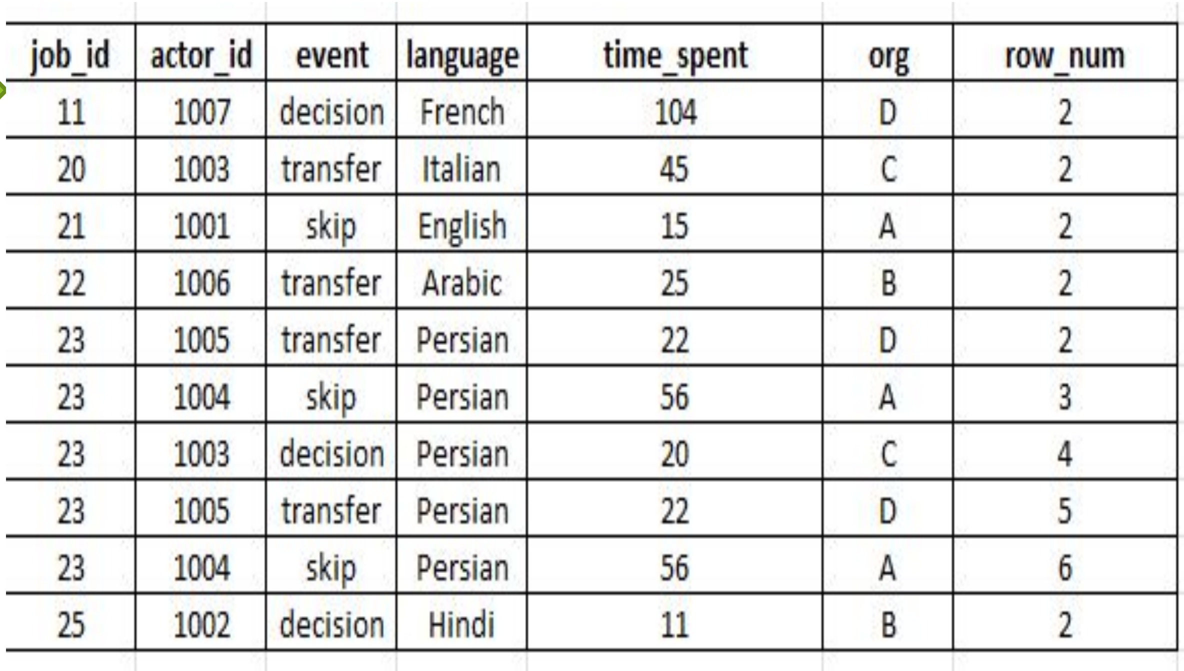
To view the duplicate rows having the same value we will:-

1. First decide in which we need to find the duplicate row values.
2. After deciding the column(parameter) we will use the ROW_NUMBER function to find the row numbers having the same value
3. Then we will portion the ROW_NUMBER function over the column (parameter) that we decided i.e. job_id
4. Then using the WHERE function we will find the row_num having a value greater than 1 that is row_num > 1 based on the occurrence of the job_id in the table

QUERIES

```
SELECT *
FROM
(
SELECT *,
ROW_NUMBER()OVER(PARTITION
BY job_id) AS row_num
 FROM job_data
 ) a
WHERE row_num>1;
```

RESULTS



job_id	actor_id	event	language	time_spent	org	row_num
11	1007	decision	French	104	D	2
20	1003	transfer	Italian	45	C	2
21	1001	skip	English	15	A	2
22	1006	transfer	Arabic	25	B	2
23	1005	transfer	Persian	22	D	2
23	1004	skip	Persian	56	A	3
23	1003	decision	Persian	20	C	4
23	1005	transfer	Persian	22	D	5
23	1004	skip	Persian	56	A	6
25	1002	decision	Hindi	11	B	2

CASE - 2

Investigating Metric Spike

(TASK – 1) User Engagement: To measure the activeness of a user. Measuring if the user finds quality in a product/service.

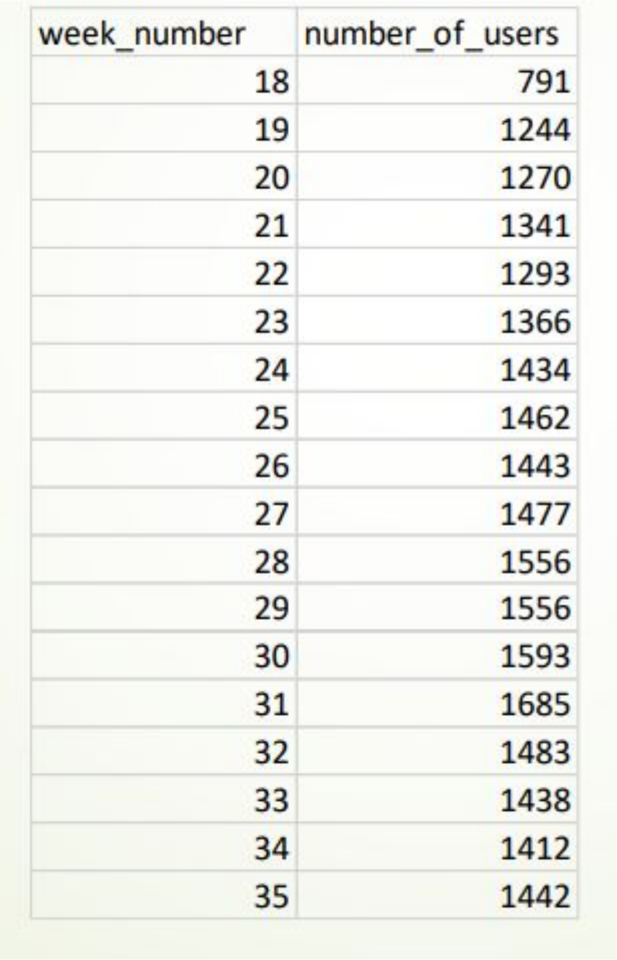
My task: Calculate the weekly user engagement.

STEPS THAT I HAVE TO FOLLOW: 1. We will extract the week's data from the occurred_at column of the events table using the EXTRACT function and WEEK function
2. Then we will be counting the number of distinct user_id from the events table
3. Then we will use the GROUP BY function to group the output that means a week from occurred_at Program/Query.

QUERY

```
SELECT
 extract (week from occurred_at)
as week_number,
 count(distinct user_id) as
number_of_users
FROM
tutorial.yammer_events
 group by
week_number;
```

RESULTS



week_number	number_of_users
18	791
19	1244
20	1270
21	1341
22	1293
23	1366
24	1434
25	1462
26	1443
27	1477
28	1556
29	1556
30	1593
31	1685
32	1483
33	1438
34	1412
35	1442

(TASK: 2)User Growth: Amount of users growing over time for a product.

My task: Calculate the user growth for product. User Growth = Number of active users per week

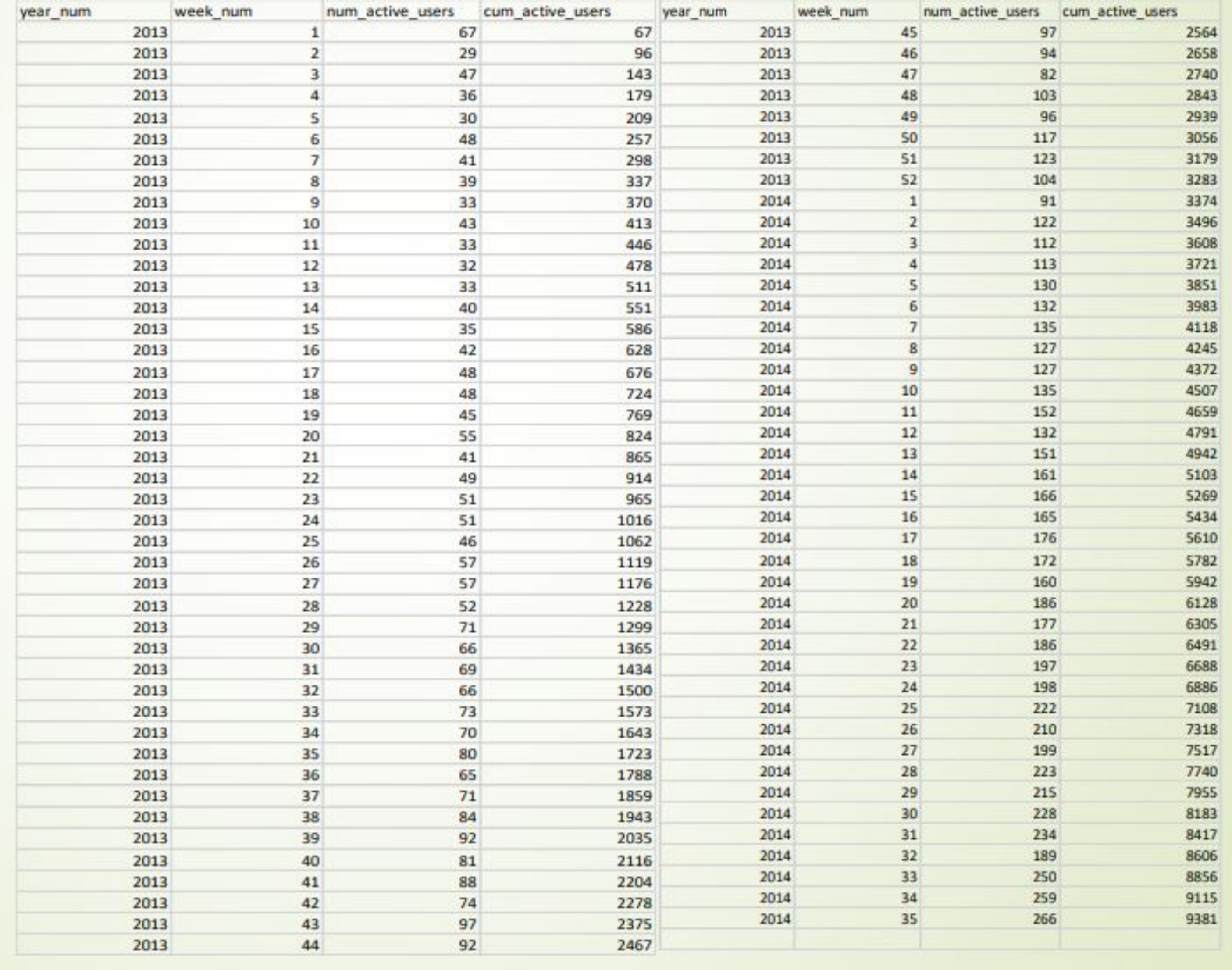
STEPS THAT I HAVE TO FOLLOW :

1. First I have to extract the year and week for the **occurred_at** column of the users table using the **extract, year and week** functions.
2. Then I have to group the extracted week and year on the basis of year and week number.
3. Then I have to order the result on the basis of year and week numbers.
4. Then I have to find the cumm_active_users using the **SUM**, **OVER** and **ROW** function **between unbounded preceding and current row.**

QUERY

```
Select
 year_num,
 week_num,
num_active_users,
SUM(num_active_users)OVER(ORDER BY
year_num, week_num ROWS BETWEEN
UNBOUNDED PRECEDING AND CURRENT ROW) AS
cum_active_users
 from
 (
 select
 extract (year from a.activated_at) as year_num,
extract (week from a.activated_at) as
week_num,
 count(distinct user_id) as num_active_users
from
 tutorial.yammer_users a WHERE
state = 'active’
 group by year_num,week_num
order by year_num,week_num
) a;
```

RESULTS



year_num	week_num	num_active_users	cum_active_users
2013	1	67	67
2013	2	29	96
2013	3	47	143
2013	4	36	179
2013	5	30	209
2013	6	48	257
2013	7	41	298
2013	8	39	337
2013	9	33	370
2013	10	43	413
2013	11	33	446
2013	12	32	478
2013	13	33	511
2013	14	40	551
2013	15	35	586
2013	16	42	628
2013	17	48	676
2013	18	48	724
2013	19	45	769
2013	20	55	824
2013	21	41	865
2013	22	49	914
2013	23	51	965
2013	24	51	1016
2013	25	46	1062
2013	26	57	1119
2013	27	57	1176
2013	28	52	1228
2013	29	71	1299
2013	30	66	1365
2013	31	69	1434
2013	32	66	1500
2013	33	73	1573
2013	34	70	1643
2013	35	80	1723
2013	36	65	1788
2013	37	71	1859
2013	38	84	1943
2013	39	92	2035
2013	40	81	2116
2013	41	88	2204
2013	42	74	2278
2013	43	97	2375
2013	44	92	2467
2013	45	97	2564
2013	46	94	2658
2013	47	82	2740
2013	48	103	2843
2013	49	96	2939
2013	50	117	3056
2013	51	123	3179
2013	52	104	3283
2014	1	91	3374
2014	2	122	3496
2014	3	112	3608
2014	4	113	3721
2014	5	130	3851
2014	6	132	3983
2014	7	135	4118
2014	8	127	4245
2014	9	127	4372
2014	10	135	4507
2014	11	152	4659
2014	12	132	4791
2014	13	151	4942
2014	14	161	5103
2014	15	166	5269
2014	16	165	5434
2014	17	176	5610
2014	18	172	5782
2014	19	160	5942
2014	20	186	6128
2014	21	177	6305
2014	22	186	6491
2014	23	197	6688
2014	24	198	6886
2014	25	222	7108
2014	26	210	7318
2014	27	199	7517
2014	28	223	7740
2014	29	215	7955
2014	30	228	8183
2014	31	234	8417
2014	32	189	8606
2014	33	250	8856
2014	34	259	9115
2014	35	266	9381

TASK : 3 Weekly Retention: Users get retained weekly after signing up for a product.
My task: Calculate the weekly retention of users-sign up cohort.

STEPS THAT I HAVE TO FOLLOW:

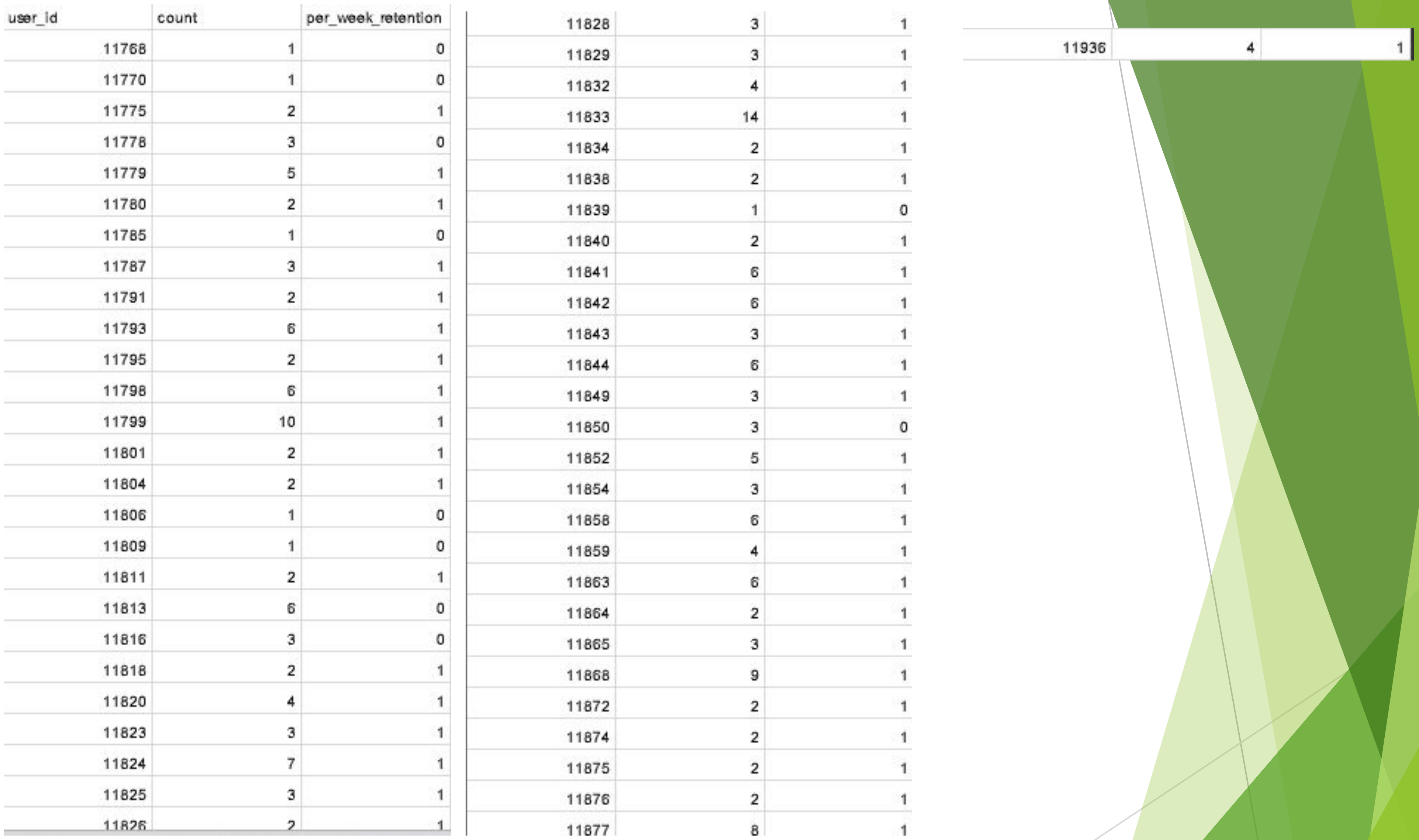
The weekly retention of the users-sign-up cohort can be calculated by two means that means either by specifying the week number (18 to 35) or for the entire column of occurred_at of the events table.

1. Firstly I have to extract the week from the occurred_at column using the extract, week functions.
2. Then, I have to select those rows in which event_type = 'signup_flow’ and event_name = 'complete_signup’.
3. If finding for a specific week I have to specify the week number using the extract function.
4. Then using the left join I have to join the two tables on the basis of user_id where event_type = 'engagement’.
5. Then I have to use the Group By function to group the output table on the basis of user_id.
6. Lastly I have to use the Order By function to order the result table on the basis of user_id.

QUERY

```
SELECT
 distinct user_id,
 COUNT(user_id),
 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 tutorial.yammer_events
WHERE event_type = 'signup_flow’
 and event_name = 'complete_signup’
 and extract(week from occurred_at) = 18
)a
 LEFT JOIN
 (SELECT distinct user_id, extract (week from occurred_at) as
engagement_week FROM tutorial.yammer_events where
event_type = 'engagement’ )
b
on a.user_id = b.user_id
 )
 )d
 group by user_id
 order by user_id;
```

RESULTS



user_id	count	per_week_retention
11768	1	0
11770	1	0
11775	2	1
11778	3	0
11779	5	1
11780	2	1
11785	1	0
11787	3	1
11791	2	1
11793	6	1
11795	2	1
11798	6	1
11799	10	1
11801	2	1
11804	2	1
11806	1	0
11809	1	0
11811	2	1
11813	6	0
11816	3	0
11818	2	1
11820	4	1
11823	3	1
11824	7	1
11825	3	1
11826	2	1
11828	3	1
11829	3	1
11832	4	1
11833	14	1
11834	2	1
11838	2	1
11839	1	0
11840	2	1
11841	6	1
11842	6	1
11843	3	1
11844	6	1
11849	3	1
11850	3	0
11852	5	1
11854	3	1
11858	6	1
11859	4	1
11863	6	1
11864	2	1
11865	3	1
11868	9	1
11872	2	1
11874	2	1
11875	2	1
11876	2	1
11877	8	1
11936	4	1

TASK : 4 Weekly Engagement: To measure the activeness of a user. Measuring if the user finds quality in a product/service weekly.
My task: Calculate the weekly engagement per device?

STEPS THAT I HAVE TO FOLLOW:

To find the weekly user engagement per device:-

1. First, I have to extract the year_num and week_num from the occurred_at column of the events table using the extract, year and week function.
2. Then I have to select those rows where event_type = 'engagement’ using the WHERE clause.
3. Lastly by using the Group By and Order By function I have to group and order the result on the basis of year_num, week_num and device.

QUERY

```
SELECT
extract(year from occurred_at) as year_num,
extract(week from occurred_at) as week_num,
device,
 COUNT(distinct user_id) as no_of_users
FROM
tutorial.yammer_events
where event_type = 'engagement’
GROUP by 1,2,3
order by 1,2,3;
```

RESULTS

TO FIND THE RESULT PLEASE CLICK ON THE LINK: ..\TASK 4 RESULT.csv

TASK: 5 Email Engagement: Users engaging with the email service.

My task: Calculate the email engagement metrics.

STEPS THAT I HAVE TO FOLLOW:

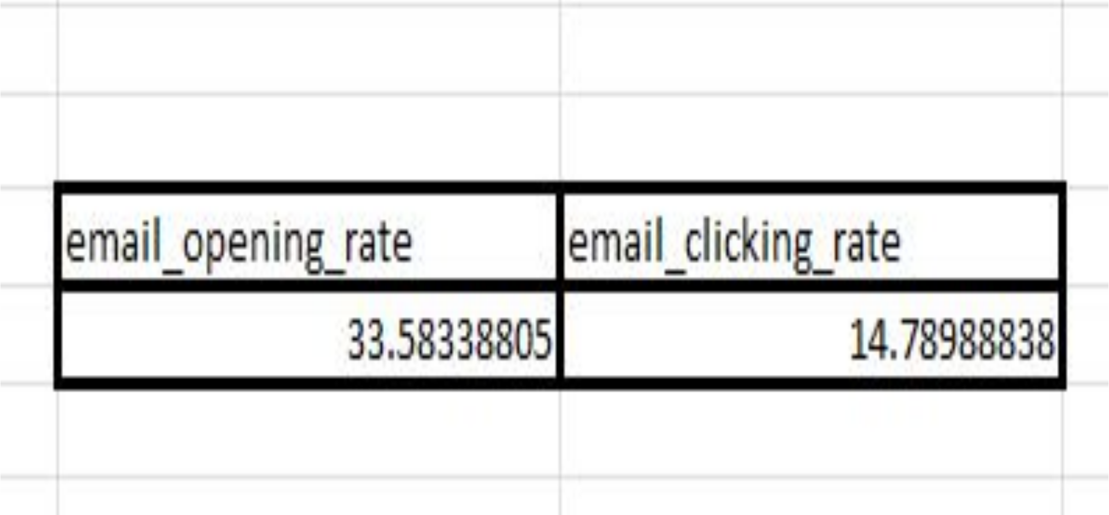
To find the email engagement metrics of users:-

1. I have to first categorize the action on the basis of email_sent, email_opened and email_clicked using the **CASE**, **WHEN**, **THEN** functions.
2. Then I have to select the sum of the category of email_opened divide by the sum of the category of email_sent and multiply the result by 100.0 and name is email_opening_rate.
3. Then I have to select the sum of the category of email_clicked divide by the sum of the category of email_sent and multiply the result by 100.0 and name is as email_clicking_rate
4. email_sent = ('sent_weekly_digest','sent_reengagement_email')
5. email_opened = 'email_open'
6. email_clicked = 'email_clickthrough'.

QUERY

```
SELECT
 100.0*SUM(CASE when email_cat = 'email_opened'
then 1 else 0 end)/SUM(CASE when
email_cat = 'email_sent' then 1 else 0 end) as
email_opening_rate, 100.0*SUM(CASE when
email_cat = 'email_clicked' then 1 else 0
end)/SUM(CASE when email_cat = '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_cat
from tutorial.yammer_emails
) a;
```

RESULTS



email_opening_rate	email_clicking_rate
33.58338805	14.78988838

Software used: MySQL Workbench 8.0 CE

THANK YOU

THE PRESENTATION IS MADE BY

DEBANJAN BHATTACHARJEE