Name:- Yash Rajendra Gaikwad
Data Analytics Trainee
Project 3:- Operation Analytics and Investigating Metric Spike
Software Used:- My SQL Workbench 8.0 CE

❖ Case Study 1 : Job Data

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

Your 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.

Your 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?

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

Your 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.

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

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.

Your task: Calculate the weekly user engagement?

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

Your task: Calculate the user growth for product?

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

Your 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.

Your task: Calculate the weekly engagement per device?

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

Your task: Calculate the email engagement metrics?

Case Study 1 : Job Data

A. **Number of jobs reviewed:-** Amount of jobs reviewed over time.
   Your task:- Calculate the number of jobs reviewed per hour per day for November 2020?

- **Process:-**
- We will use the data from job_id columns of the job_data table.
- Then we will 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.

- **Query & Result:-**The number of jobs reviewed per hours per day for november 2020 is **0.01**.

```
21    # TASK-A NUMBER OF JOBS REVIEWED #
22
23    SELECT
24        COUNT(job_id) / (30 * 24) AS number_of_jobs_reviewed_per_day_non_distinct
25    FROM
26        job_data;
```

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

number_of_jobs_reviewed_per_day_non_distinct
0.0111

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

Your 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?

- **Process:-**
- For calculating the throughput we will be using the 7-day rolling because 7-day rolling gives us the average for all the days right from day 1 to day 7 Whereas daily metric gives us average for only that particular day itself.
- We will be first taking the count of job_id(distinct and non-distinct) and ordering them w.r.t ds (date of interview).
- Then by using the ROW function we will be considering the rows between 6 preceding rows and the current row.
- Then we will be taking the average of the jobs_reviewed.

- **Query & Result:-**

```
28      # TASK-B THROUGHPUT #
29
30  •   SELECT ds as date_of_review, jobs_reviewed, AVG(jobs_reviewed)
31      OVER(ORDER BY ds ROWS BETWEEN 6 PRECEDING AND CURRENT ROW) AS
32      throughput_7_rolling_average
33      FROM
34      (
35      SELECT ds, COUNT( DISTINCT job_id) AS jobs_reviewed
36      FROM job_data
37      GROUP BY ds ORDER BY ds
38      ) a;
```

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

date_of_review	jobs_reviewed	throughput_7_rolling_average
2020-11-25	1	1.0000
2020-11-26	1	1.0000
2020-11-27	1	1.0000
2020-11-28	2	1.2500
2020-11-29	1	1.2000
2020-11-30	2	1.3333

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

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

- **Process:-**
- First we select the job_id, language and we use Count function for language.
- We name it as total_of_each_language.
- Then divide the total number of languages (distinct/non-distinct) by the total number of rows presents in the table.
- Name it as percentage_share_of_each_language.
- Then we will do the grouping based on the languages.

- **Query & Result:-** The Percentage share of each language in 30 days is given in the table.

```
40    # Task-c PERCENTAGE SHARE OF EACH LANGUAGE #
41
42    SELECT
43        job_data.job_id,
44        job_data.language,
45        COUNT(job_data.language) AS total_of_each_language,
46        ((COUNT(job_data.language) / (SELECT
47                COUNT(*)
48            FROM
49                job_data)) * 100) AS percentage_share_of_each_language
50    FROM
51        job_data
52    GROUP BY job_data.language;
```

job_id	language	total_of_each_language	percentage_share_of_each_language
21	English	1	12.5
22	Arabic	1	12.5
23	Persian	3	37.5
25	Hindi	1	12.5
11	French	1	12.5
20	Italian	1	12.5

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

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

- **Process:-**
- First decide in which do we need to find the duplicate row values
- After deciding the column(parameter) we will use the ROW_NUMBER function to find the row numbers having the same value
- Then we will portioning the ROW_NUMBER function over the column (parameter) that we decided i.e. job_id
- Then using the WHERE function we will find the row_num having value greater than 1 i.e. row_num > 1 based on the occurrence of the job_id in the table.

- **Query & Result:-** There are **two duplicate rows** that have the same value present in them.

```
54    # TASK-D DUPLICATE ROWS #
55
56    SELECT *
57    FROM
58    (
59    SELECT *, ROW_NUMBER()OVER(PARTITION BY job_id) AS row_num
60    FROM job_data
61    ) a
62    WHERE row_num>1;
```

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

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

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.

Your task: Calculate the weekly user engagement?

- **Process:-**
- We will extract the week from the occurred_at column of the events table using the EXTRACT function and WEEK function.
- Then we will be counting the number of distinct user_id from the events table.
- Then we will use the GROUP BY function to group the output w.r.t week from occurred_at.

- **Query & Result:-** The weekly user engagement is given in the table.

```sql
# TASK-2 A USER ENGAGEMENT #

SELECT
    *,
    EXTRACT(WEEK FROM occurred_at) AS week_number,
    COUNT(DISTINCT user_id) AS number_of_users
FROM
    project3.The_events
GROUP BY week_number;
```

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

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

Your task: Calculate the user growth for product?

- **Process:-**
- First we will the extract the year and week for the occurred_at column of the users table using the extract, year and week functions.
- Then we will group the extracted week and year on the basis of year and week number.
- Then we ordered the result on the basis of year and week number.
- Then we will find the cumm_active_users using the SUM, OVER and ROW function between unbounded preceding and current row.

- **Query & Result:-** Hence,There are total 9381 active users. The table shows some active users.

```
# Task-2 B  USER GROWTH #

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 activated_at) as year_num,
extract (week from activated_at) as week_num,
count(distinct user_id) as num_active_users
from
users
WHERE
state = 'active'
group by year_num,week_num
order by year_num,week_num
) ;
```

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

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

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

- **Process:-**
- We can calculate the weekly retention of users-sign up cohort by two means i.e. either by specifying the week number or for the entire column of occurred_at of the events table.
- In this case we have not given specific weeks so we calculate for the entire column of occurred_at of event table.
- Firstly we will extract the week from occurred_at column using the extract,weekfunctions.
- Then, we will select out those rows in which event_type = 'signup_flow' and event_name = 'complete_signup'.
- If finding for a spectifc week we will spectify the week number using the extract function.
- Then using the left join we will join the two tables on the basis of user_id where event_type = 'engagement' .
- Then we will use the Group By function to group the output table on the basis of user_id.
- Then we will use the Order By function to order the result table on the basis of user_id.

- **Query & Result:-** There are total 4680 users data, I have only mention some of them in the table.

```sql
# TASK-2 C WEEKLY RETENTION #

SELECT
distinct user_id,
COUNT(user_id),
SUM(CASE WHEN retention_week = 1 Then 1 Else 0 END) as week_1
FROM
(
SELECT
user_id,signup_week,engagement_week,engagement_week-signup_week as week_1
FROM
(
(SELECT distinct user_id, extract(week from occurred_at) as signup_week from The_events
WHERE event_type = 'signup_flow'
and event_name = 'complete_signup'
and extract(week from occurred_at) = 18
)
LEFT JOIN
(SELECT distinct user_id, extract (week from occurred_at) as engagement_week FROM The_events
where event_type = 'engagement'
)
on user_id = user_id
)order by user_id;
```

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

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

Your task: Calculate the weekly engagement per device?

- **Process:-**
- Firstly we will extract the year_num and week_num from the occurred_at column of the events table using the extract, year and week function.
- Then we will select those rows where event_type = 'engagement' using the WHERE clause.
- Then by using the Group By and Order By function we will group and order the result on the basis of year_num, week_num and device.

- **Query & Result:-** There are total 468 users data, I have only mention some of them in the table.

```
124     # TASK-2 D WEEKLY ENGAGEMENT #
125
126 •   SELECT
127     extract(year from occurred_at) as year_num,
128     extract(week from occurred_at) as week_num,
129     device,
130     COUNT(distinct user_id) as no_of_users
131     FROM
132     The_events
133     where event_type = 'engagement'
134     GROUP by 1,2,3
135     order by 1,2,3;
```

year_num	week_num	device	no_of_users
2014	18	acer aspire desktop	10
2014	18	acer aspire notebook	21
2014	18	amazon fire phone	4
2014	18	asus chromebook	23
2014	18	dell inspiron desktop	21
2014	18	dell inspiron notebook	49
2014	18	hp pavilion desktop	15
2014	18	htc one	16
2014	18	ipad air	30
2014	18	ipad mini	21
2014	18	iphone 4s	21
2014	18	iphone 5	70
2014	18	iphone 5s	45
2014	18	kindle fire	6
2014	18	lenovo thinkpad	90
2014	18	macbook air	57
2014	18	macbook pro	154
2014	18	mac mini	8
2014	18	nexus 10	16
2014	18	nexus 5	43
2014	18	nexus 7	20
2014	18	nokia lumia 635	19
2014	18	samsumg galaxy tablet	8
2014	18	samsung galaxy note	7
2014	18	samsung galaxy s4	56
2014	18	windows surface	10

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

Your task: Calculate the email engagement metrics?

- **Process:-**
- We will first categorize the action on the basis of email_sent, email_openedand email_clicked using the CASE, WHEN, THEN functions.
- Then we select the sum of category of email_opened divide by the sum of the category of email_sent and multiply the result by 100.0 and name is as email_opening_rate.
- Then we select the sum of 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.
- Then we use comand email_sent = ('sent_weekly_digest','sent_reengagement_email').
- Then we use command email_opened = 'email_open'.
- Lastly we use command email_clicked = 'email_clickthrough'.

- **Query & Result:-** Hence, The total email opening rate is **33.58%** and total email clicking rate is **14.78%**.

```sql
# TASK-2 E EMAIL ENGAGEMENT #

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 email_events
) a;
```

Result Grid | Filter Rows:

email_opening_rate	email_clicking_rate
33.58339	14.78989

END