Operation Analytics and investigating metric spike

Project Description:

Operation Analysis is the process of analysing companies end to end process. In this project as a data analyst, I'll show you how the data are fetched by writing the Sql queries. Here we are using advanced sql queries to extract the input from database.

Step by Step Approach:

- 1. Create a database with relavent table.
- 2. Use Mysql workbench to create a table.
- 3. Import csv file from Mysql to excel worksheet.
- 4. Analyse the process which you need to give for the user.

Tech-Stack used:

Here I am using MYSQL database. It is much better than other database. We can export the database to microsoft excel as a csv file. It is quick easier for analyst to understand the structure and is flexible to work with it to fetch the data for the output.

Insights:

CASE STUDY 1: JOB DATA ANALYSIS

You will be working with a table named **job_data** with the following columns:

job_id: Unique identifier of jobs

actor_id: Unique identifier of actor

event: The type of event (decision/skip/transfer).

language: The Language of the content

time_spent: Time spent to review the job in seconds.

org: The Organization of the actor

ds: The date in the format yyyy/mm/dd (stored as text).

Tasks:

1) 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.

2) 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.

3) 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.

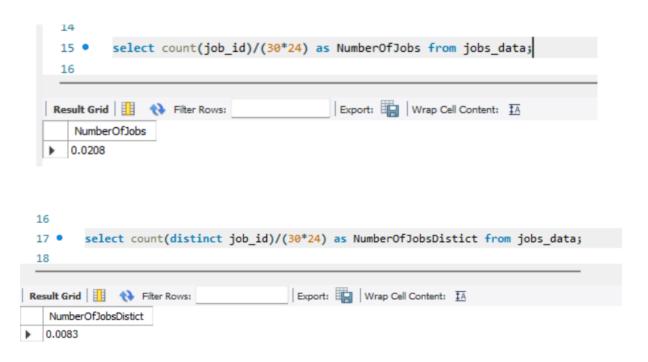
4) **Duplicate Rows Detection:**

Objective: Identify duplicate rows in the data.

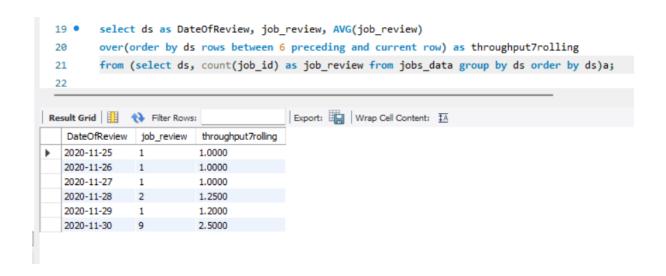
Your Task: Write an SQL query to display duplicate rows from the job_data table.

ANSWERS:

1) Jobs Reviewed Over Time:



2) Throughput Analysis:



```
select ds as DateOfReview, job_review, AVG(job_review)
23 •
        over(order by ds rows between 6 preceding and current row) as throughput7rollingDistinct
24
25
        from (select ds, count(job_id) as job_review from jobs_data group by ds order by ds)a;
26
                                      Export: Wrap Cell Content: IA
DateOfReview job_review throughput7rollingDistinct
 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
                       1.2000
             1
  2020-11-30 9
                       2.5000
```

3) Language Share Analysis:

```
select jobs_data.job_id,jobs_data.language1,
count(jobs_data.language1) as totalOfEachlanguage,
((count(jobs_data.language1)/(select count(*) from jobs_data))*100) as PercentageShareOfEachLanguage
from jobs_data
group by jobs_data.language1;

group by jobs_data.language1;
```

```
select jobs_data.job_id,jobs_data.language1,

count(jobs_data.language1) as totalOfEachlanguage,

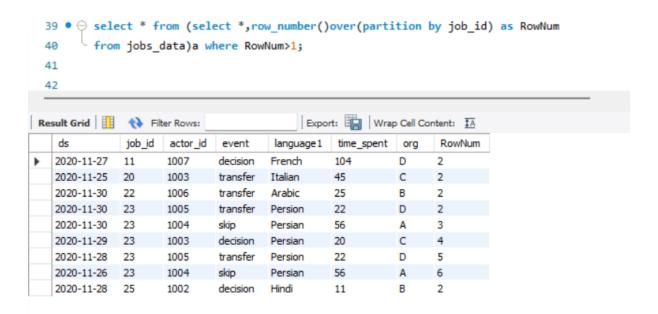
((count(jobs_data.language1)/(select count(*) from jobs_data))*100) as PercentageShareOfEachLanguageDistinct

from jobs_data

group by jobs_data.language1;

group by jobs_data.language1;
```

4) Duplicate Rows Detection:



CASE STUDY 2: INVESTIGATING METRIC SPIKE

You will be working with three tables:

users: Contains one row per user, with descriptive information about that user's account.

events: Contains one row per event, where an event is an action that a user has taken (e.g., login, messaging, search).

email_events: Contains events specific to the sending of emails.

Tasks:

1) 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.

2) 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.

3) 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.

4) 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.

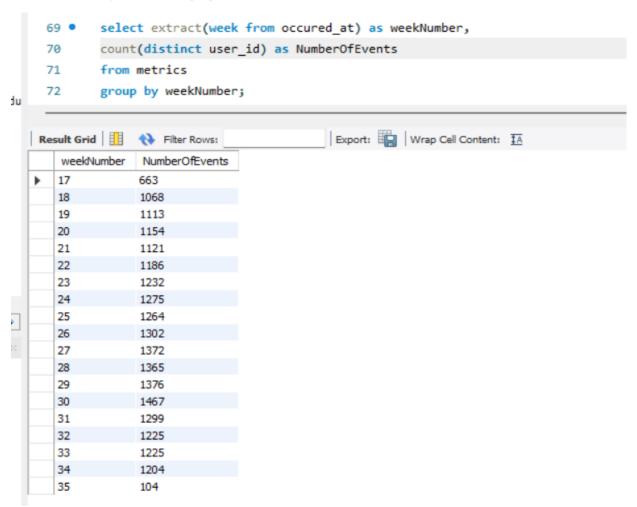
5) **Email Engagement Analysis:**

Objective: Analyze how users are engaging with the email service.

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

ANSWERS:

1) Weekly User Engagement:



2) User Growth Analysis:

```
select yearNum, weekNum, NumActiveUsers,
      SUM(NumActiveUsers)OVER(order by yearNum, weekNum ROWS BETWEEN UNBOUNDED PRECEDING AND CURRENT ROW)
75
76
      as ActiveUsers
77
   extract(WEEK from a.activated_at) as WeekNum,
78
79
       count(distinct user_id) as NumActiveUsers
30
       from Operation_metric a
      where state="active"
31
32
       group by yearNum, weekNum order by yearNum, weekNum) a;
33
34
```

```
83
84 • select count(*) from Operation_metric where state='active';
85
86
87
```

3) Weekly Retention Analysis:

```
select distinct user_id,

count(user_id),sum(case when RetentionWeek=1 then 1 else 0 end) as PerWeekRetention

from(select a.user_id,a.signupWeek,b.engagementWeek,b.engagementWeek-a.signWeek as RetentionWeek

from (select distinct user_id,extract(week from occured_at)as signupWeek from metrics

where event_type='signupFlow'

and event_name='completeSignup'

)a

left join

(select distinct user_id,extract(week from occured_at)as engagementWeek from metrics

where event_type='engagement'

)b

on a.user_id=b.user_id

)

group by user_id, order by user_id;
```

4) Weekly Engagement Per Device:

```
select extract(year from occured_at) as yearNum,
extract(week from occured_at) as weekNum,
device,
count(distinct user_id)as No_of_users from metrics
where event_type='engage'
group by 1,2,3 order by 1,2,3;
```

5) Email Engagement Analysis:

```
select 100.0 * sum(case when email='emailOpened' then 1 else 0 end)/sum(case when email='emailSent'
then 1 else 0 end) as emailOpening,

100.0 * sum(case when email='emailClicked' then 1 else 0 end)/sum(case when email='emailSent'
then 1 else 0 end) as emailClicked
from (select * , case when action in('SentWeekly','SentReengagement') then 'emailSent'
when action in ('emailOpened')
then 'emailOpening'
when action in ('emailClicked')
then 'emailClicked'
end as email
from metrics
)a;
```