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:

```
14
15 •  select count(job_id)/(30*24) as NumberOfJobs from jobs_data;
16
```

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

NumberOfJobs
0.0208

```
16
17 •  select count(distinct job_id)/(30*24) as NumberOfJobsDistict from jobs_data;
18
```

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

NumberOfJobsDistict
0.0083

2) Throughput Analysis:

```
19 •  select ds as DateOfReview, job_review, AVG(job_review)
20    over(order by ds rows between 6 preceding and current row) as throughput7rolling
21    from (select ds, count(job_id) as job_review from jobs_data group by ds order by ds)a;
22
```

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

DateOfReview	job_review	throughput7rolling
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	9	2.5000

```sql
select ds as DateOfReview, job_review, AVG(job_review)
over(order by ds rows between 6 preceding and current row) as throughput7rollingDistinct
from (select ds, count(job_id) as job_review from jobs_data group by ds order by ds)a;
```

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

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	1.2000
2020-11-30	9	2.5000

3) Language Share Analysis:

```sql
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;
```

```sql
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;
```

4) Duplicate Rows Detection:

```
39 • select * from (select *,row_number()over(partition by job_id) as RowNum
40    from jobs_data)a where RowNum>1;
41
42
```

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

ds	job_id	actor_id	event	language1	time_spent	org	RowNum
2020-11-27	11	1007	decision	French	104	D	2
2020-11-25	20	1003	transfer	Italian	45	C	2
2020-11-30	22	1006	transfer	Arabic	25	B	2
2020-11-30	23	1005	transfer	Persion	22	D	2
2020-11-30	23	1004	skip	Persian	56	A	3
2020-11-29	23	1003	decision	Persian	20	C	4
2020-11-28	23	1005	transfer	Persion	22	D	5
2020-11-26	23	1004	skip	Persian	56	A	6
2020-11-28	25	1002	decision	Hindi	11	B	2

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:

```
69 •  select extract(week from occured_at) as weekNumber,
70    count(distinct user_id) as NumberOfEvents
71    from metrics
72    group by weekNumber;
```

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

weekNumber	NumberOfEvents
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

2) User Growth Analysis:

```
74 •  select yearNum,weekNum,NumActiveUsers,
75    SUM(NumActiveUsers)OVER(order by yearNum,weekNum ROWS BETWEEN UNBOUNDED PRECEDING AND CURRENT ROW)
76     as ActiveUsers
77    from (select extract(YEAR from a.activated_at) as yearNum,
78    extract(WEEK from a.activated_at) as WeekNum,
79    count(distinct user_id) as NumActiveUsers
80    from Operation_metric a
81    where state="active"
82    group by yearNum,weekNum order by yearNum,weekNum) a;
83
84
```

```
83
84 •    select count(*) from Operation_metric where state='active';
85
86
87
88
```

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:

```
1 •  select extract(year from occured_at) as yearNum,
2    extract(week from occured_at) as weekNum,
3    device,
4    count(distinct user_id)as No_of_users from metrics
5    where event_type='engage'
6    group by 1,2,3 order by 1,2,3;
7
```

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;
```