Operation Analytics and Investigating Metric Spike

Project Description:

Operation Analytics is the analysis done for the complete end to end operations of a company. With the help of this, the company then finds the areas on which it must improve upon.

Being one of the most important parts of a company, this kind of analysis is further used to understanding between cross-functional teams, and more effective workflows.

Investigating metric spike is also an important part of operation analytics as being a Data Analyst we must be able to understand or make other teams understand questions like- Why is there a dip in daily engagement? Why have sales taken a dip? Etc. Questions like these must be answered daily and for that it's very important to investigate metric spike.

I am working for a company like Microsoft designated as Data Analyst Lead and is provided with different data sets, tables from which I must derive certain insights out of it and answer the questions asked by different departments.

The things that we are going to find out through the projects are:

- Number of jobs reviewed
- Throughput
- Percentage share of each language
- Duplicate rows
- User Engagement
- User Growth
- Weekly Retention
- Weekly Engagement
- Email Engagement

Approach:

Firstly, I spent some time on understanding the data/table given. I cleared the questions which was in my mind like what does the job_id, actor_id, event means and what are the things to consider while reviewing the data. I use SQL to derive different insights from the dataset provided by the management team. I first created a database "operation_analytics" and then the tables using the structure and links provided by the team. Then, we performed analysis to generate valuable insights for the company.

Tech Stack Used:

The tech stack used include of MySQL Workbench which is an excellent tool for querying the database.

Execution:

Case Study 1: Job Data Analysis

A) 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.

Query:

```
select
        avg(t) as "avg jobs reviewed per day per hour",
        avg(p) as "avg jobs reviewed per day per second"
from (
        select
                ds,
                ((count(job_id) * 3600) / sum(time_spent)) AS t,
                (count(job_id) / sum(time_spent)) AS p
        From
                job_data
        Where
                month(ds) = 11
                group by ds ) a;
```

Result:

avg jobs reviewed per day per hour	avg jobs reviewed per day per second
126.18048333	0.03505000

B) 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.

Query:

```
select
        count(event)/sum(time_spent) as "7-day rolling average of throughput"
from
        job_data;


select
        ds,
   count(event)/sum(time_spent) as "7-day rolling average of throughput"
from
        job_data
group by
        ds
order by
        ds;
```

Result:

ds	7-day rolling average of throughput
2020-11-25	0.0222
2020-11-26	0.0179
2020-11-27	0.0096
2020-11-28	0.0606
2020-11-29	0.0500
2020-11-30	0.0500

C) 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.

Query:

```
select
        language as Language,
   round(100 * count(*) / total , 2) as Percentage, sub.total
from
        job_data
cross join (
        select
                count(*) as Total
        from
                job_data ) as sub
        group by
                language, sub.total;
```

Result:

Language	Percentage	total
English	12.50	8
Arabic	12.50	8
Persian	37.50	8
Hindi	12.50	8
French	12.50	8
Italian	12.50	8

D) 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.

Query:

```
select
        actor_id,
        count(*) as duplicate
from
        job_data
group by
        actor_id
having
        count(*) > 1 ;
```

Result:

actor_id	duplicate
1003	2

Case Study 2: Investigating Metric Spike

A) 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.

Query:

```
select
        extract(week from occurred_at) as week_number,
        count(distinct user_id) as active_user
from
        events
where
        event_type = 'engagement'
group by
        week_number
order by
        week_number ;
```

Result:

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

B) 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.

Query:

```
set @g := 0;
select
        a.no_of_users,
        a.date,
        ( @g := @g + a.no_of_users ) as user_growth
from (
        select
                count(user_id) as no_of_users,
                date(created_at) as date
        from
                users
        where
                state = "active"
                group by date(created_at) ) a;
```

Result:

no_of_users	date	user_growth
7	2013-01-01	7
7	2013-01-02	14
6	2013-01-03	20
1	2013-01-04	21
2	2013-01-05	23
3	2013-01-06	26
4	2013-01-07	30
2	2013-01-08	32
6	2013-01-09	38
6	2013-01-10	44
6	2013-01-11	50
3	2013-01-12	53

C) 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.

Query:

```
select
        user_id,
        created_at
from
        users
where
        created_at > '2014-05-01'
order by
        user_id;
```

Result:

user_id	created_at
11768	2014-05-01 08:01:00
11770	2014-05-01 06:07:00
11775	2014-05-01 16:36:00
11778	2014-05-01 18:48:00
11779	2014-05-01 18:23:00
11780	2014-05-01 10:32:00
11785	2014-05-01 07:19:00
11787	2014-05-01 18:21:00
11791	2014-05-01 15:49:00
11793	2014-05-01 09:27:00
11795	2014-05-01 03:42:00
11798	2014-05-01 23:11:00
11799	2014-05-01 12:05:00
11801	2014-05-01 10:14:00

D) 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.

Query:

```
select
        week(occurred_at) as Weeks,
        device,
        count(distinct user_id)as User_engagement
from
        events
group by
        device,
        week(occurred_at)
order by
        week(occurred_at);
```

Result:

Weeks	device	User_engagement
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

E) 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.

Query:

```
select
        action,
        extract(month from occurred_at)  as month,
        count(action) as number_of_emails
from
        email_events
group by
        action, month
order by
        action, month;
```

Result:

action	month	number_of_emails
email_clickthrough	5	2023
email_clickthrough	6	2274
email_clickthrough	7	2721
email_clickthrough	8	1992
email_open	5	4212
email_open	6	4658
email_open	7	5611
email_open	8	5978
sent_reengagement_email	5	758
sent_reengagement_email	6	889
sent_reengagement_email	7	933
sent_reengagement_email	8	1073
sent_weekly_digest	5	11730
sent_weekly_digest	6	13155

Insights:

Case Study 1 (Job Data):

• The number of distinct jobs reviewed per hour per day for November 2020 is 83%.

• We used the 7-day rolling average of throughput as it gives the average for all the days right from day 1 to day 7 whereas, daily metric gives the average for only that particular day itself.

• The percentage share of Persian language is the most (37.5%).

• There are two duplicate rows if we partition the data by job_id. But if we look the overall columns, all the rows are unique.

Case Study 2 (Investigating metric spike):

• The weekly user engagement increased from week 18th to week 31st and then started declining from then onwards. This means that some of the users do not find much quality in the product/service in the last of the weeks.

• There are in total 9381 active users from 1st week of 2013 to the 35th week of 2014.

• The overall count of weekly engagement per device used is the most for MacBook users and iPhone users.

• The email opening rate is around 34% and email clicking rate is around 15%. The users are engaging with the email service which is good for the company to expand.

Result:

In this project, I learned how to apply advanced SQL concepts like Windows Functions, etc. I understood how the real-world industry works. It helped me in mastering my SQL concepts. I learned how to ask the right questions given the circumstances. From the given data and questions, which columns to consider and how to find the valuable insights which help the business to grow. I learned how the company find different areas related to the company to improve it further. I got to know about investigating metri c spike (why there is a boom and why there is a dip).