Operation Analytics and Investigating Metric Spike

VICTOR SHAH

Project Description

- As a Data Analyst collaborating with the Microsoft Product team, my primary responsibility is about identifying areas for improvement within the company.
- The aspect of my role is to extract meaningful insights from the user data. These insights will offer actionable information for various teams within the business. •
- The Operational Analytics is a crucial process that involves analysing a company's end-to-end operations. One of the key aspects being Investigating metric spikes.
- The goal is to empower the product manager and the entire team with actionable insights that will shape the future development and user experience in the application.

Approach

- Importing the Dataset: The first step is maintaining a file path to export data (.csv) into the SQL workbench.
- Understanding the Schema: The next step is to examine the structure of the table holding the data (job_data, events, etc).
- Identifying the Key tables: Identification of the primary key from each of the tables of job_data, email_events, events, users etc.
- Checking for null values: Before the analysis, it is necessary to check for null values in the given tables
- Visually Appealing: The SQL Queries need to be properly formatted so that they can be understood by any user.

CASE STUDY 1:
Job Data Analysis

A. Jobs Reviewed Over Time:

Calculate the number of jobs reviewed per hour for each day in November 2020.

```sql
-- TASK 01  JOBS REVIEWED OVER TIME  (VICTOR SHAH)
SELECT
    COUNT(job_id) AS number_of_jobs,
    ROUND(SUM(time_spent) / 3600, 3) AS hours_per_day
FROM
    Job_Data
WHERE
    ds BETWEEN '2020-11-01' AND '2020-11-30'
GROUP BY ds;
```

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

number_of_jobs	hours_per_day
2	0.011
1	0.006
2	0.009
1	0.029
1	0.016
1	0.013

B. Throughput Analysis:

Calculate the 7-day rolling average of throughput (number of events per second).

```sql
-- TASK 2: THROUGHPUT ANALYSIS (VICTOR SHAH)
SELECT
AVG(number_of_events) OVER(ROWS BETWEEN 6 PRECEDING AND CURRENT ROW) AS 7_day_rolling_avg
FROM
(SELECT
    COUNT(DISTINCT event) AS number_of_events
FROM
    Job_Data
GROUP BY ds) AS sub;
```

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

7_day_rolling_avg
1.0000
1.0000
1.0000
1.2500
1.2000
1.3333

C. Language Share Analysis:

Calculate the percentage share of each language in the last 30 days.

The language Persian has the most share out of the other languages in the past 30 days.

```sql
-- TASK 03 LANGUAGE SHARE ANALYSIS  (VICTOR SHAH)
SELECT language,
    ROUND((COUNT(language) / (SELECT COUNT(*) FROM job_data)) * 100,2)
            AS language_share
FROM    job_data
WHERE    ds > (SELECT
            MAX(ds) - INTERVAL 30 DAY
        FROM            job_data)
GROUP BY language;
```

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

language	language_share
English	12.50
Arabic	12.50
Persian	37.50
Hindi	12.50
French	12.50
Italian	12.50

D. Duplicate Rows Detection:

Identify duplicate rows in the data.

job_id	actor_id	event	language	time_spent	org	ds
21	1001	skip	English	15	A	2020-11-30 00:00:00
22	1006	transfer	Arabic	25	B	2020-11-30 00:00:00
23	1003	decision	Persian	20	C	2020-11-29 00:00:00
23	1005	transfer	Persian	22	D	2020-11-28 00:00:00
25	1002	decision	Hindi	11	B	2020-11-28 00:00:00
11	1007	decision	French	104	D	2020-11-27 00:00:00
23	1004	skip	Persian	56	A	2020-11-26 00:00:00
20	1003	transfer	Italian	45	C	2020-11-25 00:00:00

```sql
-- TASK 04 DUPLICATE ROWS DETECTION  (VICTOR SHAH)
SELECT
    job_id, COUNT(*)
FROM
    job_data
GROUP BY job_id , actor_id , event , language , time_spent , org , ds
HAVING COUNT(*) > 1;
```

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

job_id	COUNT(*)

The given table job_data has no duplicate rows.

Hence the query returns an empty output.

CASE STUDY 2: Investigating Metric Spike

A. Weekly User Engagement:

Measure the activeness of users on a weekly basis.

We can see that the user's engagement is measured upon events like login, home_page, like_message, view_inbox.

```sql
-- TASK 01 WEEKLY USER ENGAGEMENT (VICTOR SHAH)
SELECT EXTRACT(WEEK FROM occurred_at) AS week_num,
    COUNT(DISTINCT user_id) AS active_users
FROM    events
WHERE
    event_type = 'engagement'
GROUP BY week_num;
```

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

week_num	active_users
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

B. User Growth Analysis:

Analyze the growth of users over time for a product.

The cumulative frequency is used to determine the number of observations that lie above a particular value in a dataset.

```sql
-- TASK 02 USER GROWTH ANALYSIS (VICTOR SHAH)
with tab1 as (select extract(year from created_at) as years,
extract(month from created_at) as months,count(*) as freq from users
group by years,months)
select years, months, sum(freq) over(order by years,months) as cum_freq,
 freq as user_growth from tab1;
```

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

years	months	cum_freq	user_growth
2013	1	160	160
2013	2	320	160
2013	3	470	150
2013	4	651	181
2013	5	865	214
2013	6	1078	213
2013	7	1362	284
2013	8	1678	316
2013	9	2008	330
2013	10	2398	390
2013	11	2797	399
2013	12	3283	486
2014	1	3835	552
2014	2	4360	525
2014	3	4975	615
2014	4	5701	726
2014	5	6480	779
2014	6	7353	873
2014	7	8350	997
2014	8	9381	1031

C. Weekly Retention Analysis:

Analyze the retention of users on a weekly basis after signing up for a product.

Week by week we can see the number of active users start to deteriorate.

```sql
-- TASK 03 WEEKLY RETENTION ANALYSIS (VICTOR SHAH)
with retention as (
select e.user_id,extract(week from created_at) as create_week_no,
min(case when event_type = 'engagement' then extract(week from occurred_at) end) as login_week
 from users u join events e on u.user_id = e.user_id
 group by e.user_id,create_week_no),
 week_retention as (
 select *, login_week - create_week_no as weeks_retained from retention order by weeks_retained DESC)
 select weeks_retained, count(user_id) as no_of_users from week_retention group by weeks_retained order by weeks_retained;
```

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

weeks_retained	no_of_users
0	3766
1	98
2	109
3	80
4	79
5	65
6	60
7	51
8	63
9	66
10	64
11	61
12	76
13	69
14	72
15	59
16	58
17	67
18	61
19	50

D. Weekly Engagement Per Device:

Measure the activeness of users on a weekly basis per device.

The users are differentiated weekly by the devices they use concurrently. The count of these devices are measured.

```
-- TASK 04 WEEKLY ENGAGEMENT PER DEVICE  (VICTOR SHAH)
select extract(week from occurred_at) as weeks, device,
count(distinct user_id) as occurrence from events where event_type = 'engagement'
group by weeks, device order by weeks, device;

-- TASK 05 EMAIL ENGAGEMENT ANALYSIS  (VICTOR SHAH)

SELECT
    EXTRACT(WEEK FROM occurred_at) AS weeks,
```

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

weeks	device	occurrence
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
17	lenovo thinkpad	86
17	mac mini	6
17	macbook air	54
17	macbook pro	143
17	nexus 10	16
17	nexus 5	40

E. Email Engagement Analysis:

Analyze how users are engaging with the email service.

We are able to figure out the events occurred in the email like email_clickthrough, email_open, sent_weekly_digest and many more.

```sql
-- TASK 05 EMAIL ENGAGEMENT ANALYSIS  (VICTOR SHAH)
SELECT
    EXTRACT(WEEK FROM occurred_at) AS weeks,
    action,
    COUNT(*) AS email_actions
FROM
    email_events
GROUP BY weeks , action
ORDER BY weeks;
```

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

weeks	action	email_actions
17	email_clickthrough	166
17	email_open	310
17	sent_reengagement_email	73
17	sent_weekly_digest	908
18	email_clickthrough	430
18	email_open	912
18	sent_reengagement_email	157
18	sent_weekly_digest	2602
19	email_clickthrough	477
19	email_open	972
19	sent_reengagement_email	173
19	sent_weekly_digest	2665
20	email_clickthrough	507
20	email_open	1004
20	sent_reengagement_email	191
20	sent_weekly_digest	2733
21	email_clickthrough	443
21	email_open	1014
21	sent_reengagement_email	164
21	sent_weekly_digest	2822

Result 7

Tech-Stack Used

- MySQL Workbench(8.0.34): This is the primary interactive development environment for SQL queries. It enables efficient query building, execution and debugging for data analysis
- SQL commands:
  1. DDL commands: These commands were used for the creation of the database and the multiple tables such as users, job_data, email_events, events.
  2. DML commands: These commands were used for the insertion of the data into the records of the table.
  3. DQL commands: The select query with where, order by, group by clauses helped for the further analysis of the data from the table.
- Window Functions: Does calculationn across a set of rows that are related to the current row. These functions are used when we want to calculate Average Running Price, Running Total Orders, Running Sum Sales, Rank and Percentile.

Insights

- While analysing the tables we were able to figure the jobs reviewed per hour for each day in November 2020.
- The identification of duplicate rows from the table.
- We were able to understand how different users engage with email events in the application.
- The span of every language share in the last 30 days.
- To be able to analzye the retention of users on a weekly basis after signing up for a product..

Result

- Remembering to adapt these queries on specific database schemas.
- These learned insights helped me understand specific business questions which were addressed by SQL queries.
- Learning about the SQL clauses such as the join clauses and sub-queries. The importance of order by and group by and many more.
- We were able to import the dataset from a .CSV file into the SQL workbench for performing analysis.
- Achieving the ability to learn and write SQL queries to execute different business questions.
- Solving business related problems using Windows functions of SQL.

Thank you