Operation Analytics and Investigating Metric Spike

Project Description

This project consist of 2 Case Studies, One is for Operation Analytics and other one is for Investigating Metric Spike. In the Operation Analytics I have to find the ares on which company must improve on and I have to derive insights out of the data company collects.

In the Investigating Metric Spike, metric spike is the sudden increase in a particular metric. In this I have used different KPI's and parameter to analyse product's user engagement, user growth, weekly retention, weekly engagement and email engagement.

Approach

Approach for making this project is first undersatanding all the datasets. After that creating a database and insert all the tables or data in the database. I have used advance SQl queries like window functions, subqueries etc. to get the answers and gather insights to make good decisions for the company.

Tech-Stack Used

- I have used MySQL Version 8.0.33, MySql Workbench 8.0 CE for writing the queries.

Insights

- Through this project I have got the insights like number of jobs reviewed per hour per day, throughput, Percentage share of each language.
- Highest no. of active users each week.
- Maximum user engagement per week
- Maximum no. of email sent per month.
- Daily throughput is always better than weekly as we can see daily ups and downs.

Result

In this project of Operation Analytics and Investigating Metric Spike, I have achieved various Analytics and logical skills as well as technical skills to efficiently use MySQl. I have come to know the use of the window function, nested queries or sub-queries etc. It helped me to get more understanding of when and where to use appropriate functions. It will add to my portfolio by which this project help me to get good opportunities in the future.

Commands for Queries

Case Study 1 (Job Data)

1. Number of jobs reviewed: Amount of jobs reviewed over time.

```
SELECT
    ds,
    (COUNT(job_id) / SUM(time_spent) * 3600) AS jobs_per_hour_per_day
FROM
    job_data
WHERE
    ds BETWEEN '2020-11-01' AND '2020-11-30'
GROUP BY ds
ORDER BY ds;
```

ds	jobs_per_hour_per_day
2020-11-25	80.0000
2020-11-26	64.2857
2020-11-27	34.6154
2020-11-28	218.1818
2020-11-29	180.0000
2020-11-30	180.0000

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

-- Weekly Throghput (7 Day Rolling)

```
SELECT
   num_of_events / total_time_spent AS weekly_throughput
FROM
   (SELECT
      COUNT(event) AS num_of_events,
         SUM(time_spent) AS total_time_spent
   FROM
      job_data) j;
```

	weekly_throughput
▶	0.0268

-- Daily Throghput

```
SELECT
   ds, num_of_events / total_time_spent AS daily_throughput
FROM
   (SELECT
      ds,
         COUNT(event) AS num_of_events,
         SUM(time_spent) AS total_time_spent
   FROM
      job_data
   GROUP BY ds) j
GROUP BY ds;
```

	ds	daily_throughput
▶	2020-11-30	0.0500
	2020-11-29	0.0500
	2020-11-28	0.0606
	2020-11-27	0.0096
	2020-11-26	0.0179
	2020-11-25	0.0222

3. Percentage share of each language: Share of each language for different contents.

```
SELECT
   language,
   ROUND((COUNT(language) * 100 / (SELECT
               COUNT(*)
            FROM
               job_data)),
         2) AS percentage_share
FROM
   job_data
GROUP BY language;
```

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

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

```
SELECT *
FROM (
    SELECT *, ROW_NUMBER() OVER (PARTITION BY job_id) AS row_no
    FROM job_data
) AS a
WHERE row_no > 1;
```

job_id	actor_id	event	language	time_spent	org	ds	row_no
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)

1. User Engagement: To measure the activeness of a user. Measuring if the user finds quality in a product/service.

```
SELECT
    COUNT(DISTINCT user_id) AS active_user,
    WEEK(occurred_at) AS week_num
FROM
    events
WHERE
    event_type = 'engagement'
GROUP BY week_num;
```

active_user	week_num
663	17
1068	18
1113	19
1154	20
1121	21
1186	22
1232	23
1275	24
1264	25
1302	26
1372	27
1365	28
1376	29
1467	30
1299	31
1225	32
1225	33
1204	34
104	35

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

```
select
        year, week, num_users,
        sum(num_users) over ( rows between unbounded preceding and current row) as active_users
from
        (select year(activated_at) as year, week(activated_at) as week, count(user_id) as num_users from users
where
        state='active'
group by year, week
order by year, week) as a;
```

year	week	num_users	active_users
2013	0	23	23
2013	1	30	53
2013	2	48	101
2013	3	36	137
2013	4	30	167
2013	5	48	215
2013	6	38	253
2013	7	42	295
2013	8	34	329
2013	9	43	372
2013	10	32	404
2013	11	31	435
2013	12	33	468
2013	13	39	507
2013	14	35	542
2013	15	43	585
2013	16	46	631
2013	17	49	680
2013	18	44	724
2013	19	57	781
2013	20	39	820
2013	21	49	869
2013	22	54	923

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

```
SELECT
   YEAR(OCCURRED_AT) AS YEAR,
   WEEK(OCCURRED_AT) AS WEEK,
   DEVICE,
   COUNT(DISTINCT user_id) AS users_retention
FROM
   events
WHERE
   EVENT_TYPE = 'ENGAGEMENT'
GROUP BY year , week , device
ORDER BY year , week , device;
```

YEAR	WEEK	DEVICE	users_retention
2014	17	acer aspire desktop	9
2014	17	acer aspire notebook	20
2014	17	amazon fire phone	4
2014	17	asus chromebook	21
2014	17	dell inspiron desktop	18
2014	17	dell inspiron notebook	46
2014	17	hp pavilion desktop	14
2014	17	htc one	16
2014	17	ipad air	27
2014	17	ipad mini	19
2014	17	iphone 4s	21
2014	17	iphone 5	65
2014	17	iphone 5s	42
2014	17	kindle fire	6
2014	17	lenovo thinkpad	86
2014	17	mac mini	6
2014	17	macbook air	54
2014	17	macbook pro	143
2014	17	nexus 10	16
2014	17	nexus 5	40
2014	17	nexus 7	18
2014	17	nokia lumia 635	17
2014	17	samsumg galaxy tablet	8
2014	17	samsung galaxy note	7
2014	17	samsung galaxy s4	52

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

```
SELECT DISTINCT
    device,
    WEEK(occurred_at) AS week,
    COUNT(DISTINCT user_id) AS users_engagement
FROM
    events
WHERE
    event_type = 'engagement'
GROUP BY week , device
ORDER BY week , device;
```

device	week	users_engagement
acer aspire desktop	17	9
acer aspire notebook	17	20
amazon fire phone	17	4
asus chromebook	17	21
dell inspiron desktop	17	18
dell inspiron notebook	17	46
hp pavilion desktop	17	14
htc one	17	16
ipad air	17	27
ipad mini	17	19
iphone 4s	17	21
iphone 5	17	65
iphone 5s	17	42
kindle fire	17	6
lenovo thinkpad	17	86
mac mini	17	6
macbook air	17	54
macbook pro	17	143
nexus 10	17	16
nexus 5	17	40
nexus 7	17	18
nokia lumia 635	17	17
samsumg galaxy tablet	17	8
samsung galaxy note	17	7
samsung galaxy s4	17	52

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

```
SELECT
    YEAR(occurred_at) AS year,
    MONTH(occurred_at) AS month,
    action,
    COUNT(action) AS email_engagement
FROM
    email_events
GROUP BY year , month , action
ORDER BY year , month , action;
```

year	month	action	email_engagement
2014	5	email_clickthrough	2025
2014	5	email_open	4215
2014	5	engagement	5
2014	5	sent_reengagement_email	758
2014	5	sent_weekly_digest	11737
2014	5	signup_flow	1
2014	6	email_clickthrough	2275
2014	6	email_open	4661
2014	6	sent_reengagement_email	889
2014	6	sent_weekly_digest	13162
2014	7	email_clickthrough	2721
2014	7	email_open	5613
2014	7	sent_reengagement_email	933
2014	7	sent_weekly_digest	15907
2014	8	email_clickthrough	1992
2014	8	email_open	5979
2014	8	sent_reengagement_email	1073
2014	8	sent_weekly_digest	16485