Operation Analytics and Investigating Metric Spike

Advanced SQL

Project Description :-

Operation Analytics is the process of using data analysis and business intelligence to improve efficiency and streamline everyday operations in real time. A subset of business analytics, operational analytics is supported by data mining, artificial intelligence, and machine learning. It requires a robust team of business and data analysts. In this assignment I worked closely with the operational team, support team, and the marketing team, which helped me derive insights out of the data provided by the team.

In this assignment I have analysed job reviews, no of events happening in the company, how many languages do people speak in the company and any duplication of records. 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 regarding engagement, growth, weekly retention, email etc.

Approach:-

I Firstly evaluated the objectives and considered the actual data that the team needed and after that I imported the data into www.DB Fiddle.com and www.Mode.com and ran numerous queries to acknowledge the data and track down the insights that the team desired for maximum marketing,support and operational growth.

Tech Stack Used:

Db Fiddle (SQL Database Playground) – My SQL v 8.0 and Mode.com (A cloud based collaborative analytics platform).

Insights:

I ran various sql commands to acquire insights and understanding of how to handle real time sql queries through this assignment.I analysed the dataset furnished by the team and derived several insights about weekly user engagement, user growth, weekly retention, email engagement, number of jobs, and duplicate data .

RESULTS

Case Study 1 (JOB DATA):

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

a) TASK: Calculate the number of jobs reviewed per hour per day for November 2020?

QUERY

Query SQL

```
SELECT
  COUNT(DISTINCT job_id)/(30*24)
FROM
  job_data
WHERE
  ds between '2020-11-01' AND '2020-11-30'
```

OUTPUT

Results

Copy as Markdown

Query #1 Execution time: 0ms

COUNT(DISTINCT job_id)/(30*24)
0.0111

Throughput : It's the numbers of event happening per second.

b) <u>TASK:</u> Let's say the above metric is called throughput. Calculate seven day rolling average of throughput? For throughput, do you prefer daily metric or seven day rolling and why ?

QUERY

Database: MySQL v8.0 ▼ ▷ Run Save Load Example Collaborate

Query SQL

```sql
SELECT
  ds,
  jobs_reviewed,
  AVG(jobs_reviewed)OVER (ORDER BY ds ROWS BETWEEN 6 PRECEDING AND CURRENT ROW)
  AS throughput_7
  FROM
  (
    SELECT
    ds,
    COUNT(DISTINCT job_id) AS jobs_reviewed
    FROM
    job_data
    WHERE
    ds BETWEEN '2020-11-01' AND '2020-11-30'
    GROUP BY
    ds
    ORDER BY ds
    ) a
```

OUTPUT

Database: MySQL v8.0 ▼ ▷ Run Save Load Example Collaborate Sign in Have any feedback?

Results

Copy as Markdown

Query #1 Execution time: 1ms

ds	jobs_reviewed	throughput_7
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	2	1.3333

Percentage share of each language : share of each language for different content.

c)<u>TASK:</u> Calculate the percentage share of each language in the last 30 days ?

QUERY

Database: MySQL v8.0 ▼ ▷ Run Save Lo

Query SQL

```sql
SELECT
    language,
    num_jobs,
    100.0 * num_jobs/total_jobs AS pct_jobs
FROM
(
SELECT
      language,
      COUNT(DISTINCT job_id) AS num_jobs
  FROM
      job_data
GROUP BY
      language
) a
CROSS JOIN
(SELECT COUNT(DISTINCT job_id) AS total_jobs
 FROM
 job_data) b

```

OUTPUT

Database: MySQL v8.0 ▼ ▷ Run Save Load Example Collaborate Sign in Have any feedback?

Results Copy as Markdown

Query #1 Execution time: 2ms

language	num_jobs	pct_jobs
Arabic	1	12.50000
English	1	12.50000
French	1	12.50000
Hindi	1	12.50000
Italian	1	12.50000
Persian	3	37.50000

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

d)<u>TASK:</u> Let's say you see some duplicate rows in the data.How will you display duplicates from the table?

QUERY

Sign in

Query SQL

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

```

OUTPUT

Results

Copy as Markdown

Query #1 Execution time: 1ms

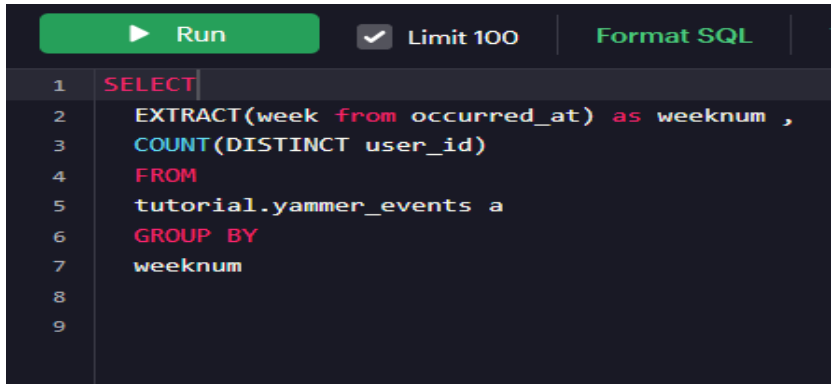
job_id	actors_id	event	language	time_spent	org	ds	row_num
28	1008	transfer	Italian	45	C	2020-11-25	2

CASE STUDY 2 (Investigating Metric Spike)

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

a)TASK: Calculate the weekly user engagement ?

QUERY



OUTPUT

Data | Fields | Source

	weeknum	count
1	18	791
2	19	1244
3	20	1270
4	21	1341
5	22	1293
6	23	1366
7	24	1434
8	25	1462
9	26	1443
10	27	1477

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

b) TASK: Calculate the user growth for product ?

QUERY

Run | Limit 100 | Format SQL | View History...

```
SELECT
  year,
  weeknum,
  num_active_user,
  SUM(num_active_user) OVER(ORDER BY year, weeknum ROWS BETWEEN UNBOUNDED PRECEDING AND CURRENT ROW) as cum_active_users
  FROM
  (
  SELECT
  EXTRACT(year from a.activated_at) as year,
  EXTRACT(week from a.activated_at) as weeknum,
  COUNT(DISTINCT user_id) as num_active_user
  FROM
  tutorial.yammer_users a
  WHERE
  state='active'
  GROUP BY
  year,weeknum
  ORDER BY
  year,weeknum ) a

```

OUTPUT

Data | Fields | Source

	year	weeknum	num_active_user	cum_active_users
1	2013	1	67	67
2	2013	2	29	96
3	2013	3	47	143
4	2013	4	36	179
5	2013	5	30	209
6	2013	6	48	257
7	2013	7	41	298
8	2013	8	39	337
9	2013	9	33	370
10	2013	10	43	413

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

c)<u>TASK:</u> Calculate the weekly retention of users-sign up (cohort) ?

QUERY

Run | Limit 100 | Format SQL | View History...

```
SELECT
COUNT(user_id),
SUM(CASE WHEN retention_week = 1 THEN 1 ELSE 0 END) as week_1
FROM
(
SELECT a.user_id, a.signup_week, b.engagement_week, b.engagement_week -a.signup_week AS retention_week
FROM
(
(SELECT DISTINCT user_id,
EXTRACT(week FROM occurred_at) AS signup_week
FROM tutorial.yammer_events
WHERE event_type = 'signup_flow'
AND event_name = 'complete_signup' AND EXTRACT(week from occurred_at) = 18 ) a
LEFT JOIN (
SELECT DISTINCT user_id,
EXTRACT(week FROM occurred_at) AS engagement_week
FROM
tutorial.yammer_events
WHERE event_type = 'engagement' ) b
ON a.user_id = b.user_id )
ORDER BY
a.user_id ) a
```

OUTPUT

Data | Fields | Source

	count	week_1
1	317	64

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

d)<u>TASK:</u> Calculate the weekly engagement per device ?

QUERY

Run | Limit 100 | Format SQL | View History...

```sql
SELECT
EXTRACT(year FROM occurred_at) AS year,
EXTRACT(week FROM occurred_at) AS week,
device,
COUNT(distinct user_id)
FROM
 tutorial.yammer_events
 WHERE
 event_type='engagement'
 GROUP BY
 1,2,3
 ORDER BY 1,2,3

```

OUTPUT

Data | Fields | Source

	year	week	device	count
1	2014	18	acer aspire desktop	10
2	2014	18	acer aspire notebook	21
3	2014	18	amazon fire phone	4
4	2014	18	asus chromebook	23
5	2014	18	dell inspiron desktop	21
6	2014	18	dell inspiron notebook	49
7	2014	18	hp pavilion desktop	15
8	2014	18	htc one	16
9	2014	18	ipad air	30
10	2014	18	ipad mini	21

Email Engagement: Users engaging with the email service.

e)<u>TASK:</u> Calculate the email engagement metrics ?

QUERY

Run | Limit 100 | Format SQL | View History...

```sql
SELECT
100.0 *SUM(CASE WHEN email_cat = 'email_open' THEN 1 ELSE 0 END)/SUM(CASE WHEN email_cat = 'email_sent' THEN 1 ELSE 0 END) AS email_open_rate,
100.0 *SUM(CASE WHEN email_cat = 'email_clicked' THEN 1 ELSE 0 END)/SUM(CASE WHEN email_cat = 'email_sent' THEN 1 ELSE 0 END) AS email_clicked_rate
FROM
(
SELECT
*,
CASE
WHEN action IN ('sent_weekly_digest', 'sent_reengagement_email')
  THEN 'email_sent'
WHEN action IN ('email_open')
  THEN 'email_open'
WHEN action in ('email_clickthrough')
  THEN 'email_clicked'
END AS email_cat
  FROM
tutorial.yammer_emails ) a
```

OUTPUT

Data | Fields | Source

	email_open_rate	email_clicked_rate
1	33.5834	14.7899

Thank You