Operation Analytics and

Investigating Metric Spike

1. **Project Description:** In today's data-driven world, businesses need to analyze their operations and metrics to stay competitive. This project aims to provide insights into two different scenarios of operation and metric analytics. One of the primary goals of the project is to provide a dashboard that displays key performance indicators (KPIs). The dashboard will provide a real-time view of KPIs to help managers make data driven decisions. Both case studies will provide valuable insights into two different scenarios and will enable businesses to make data-driven decisions, improve operational efficiency, and enhance customer satisfaction.

2. **Approach:**

- First, the data prepared and analyzed using SQL. Various SQL functions like SELECT, WHERE, GROUP BY, JOIN, etc. are used to extract meaningful information from the dataset.
- Once the data is cleaned and transformed, the next step is to identify Key Performance Indicators (KPIs) that are relevant to the business. These KPIs include throughput, user engagement, user retention etc.
- The final step is to create interactive charts and dashboards using Tableau. Tableau helps to create interactive visualizations that help in identifying trends and patterns.

3. **Tech-Stack Used:** SQL Prepare, Process and analyze data.

4. Insights:

Case Study 1:

Job Data Analysis

A. Jobs Reviewed Over Time: Write an SQL query to calculate the number of jobs reviewed per hour for each day in November 2020**.**

Code:
```
SELECT DISTINCT ds AS days,
    Count(job_id) / (Sum(time_spent) / 3600) AS no_of_jobs_reviewed
    FROM job_data
   GROUP BY days;
```

CONCLUSION:

	days	no_of_jobs_reviewed
▶	11/30/2020	180.0000
	11/29/2020	180.0000
	11/28/2020	218.1818
	11/27/2020	34.6154
	11/26/2020	64.2857

B. **Throughput Analysis**: 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.

Code: SELECT a.ds AS day,

a.throughput,

avg(a.throughput) OVER (ORDER BY ds rows BETWEEN 6 PRECEDING AND CURRENT row) AS

7_day_avg_of_throughput

FROM

(SELECT ds, count(job_id) / sum(time_spent) AS throughput FROM job_data GROUP BY ds) AS a

GROUP BY ds;

CONCLUSION:

	day	throughput	7_day_avg_of_throughput
▶	11/25/2020	0.0222	0.02220000
	11/26/2020	0.0179	0.02005000
	11/27/2020	0.0096	0.01656667
	11/28/2020	0.0606	0.02757500
	11/29/2020	0.0500	0.03206000
	11/30/2020	0.0500	0.03505000

c. **Language Share Analysis:** Write an SQL query to calculate the percentage share of each language over the last 30 days.

Code: SELECT language,

count(job_id) as no_of_jobs,

count(job_id)*100 / sum(count(*)) OVER() as percentage_share

FROM job_data

WHERE ds between '2020-11-01' and '2020-11-30'

GROUP by language;

CONCLUSION:

language	no_of_jobs	percentage_share
English	1	12.5000
Arabic	1	12.5000
Persian	3	37.5000
Hindi	1	12.5000
French	1	12.5000
Italian	1	12.5000

D. Duplicate Rows Detection: Write an SQL query to display duplicate row from the job_data table.

Code:
```
SELECT a.ds,
 a.job_id,
 a.actor_id,
 a.event,
a.language,
a.time_spent,
a.org,
CASE when a.duplicates = 1 then "No Duplicate" else "Duplicate"
 end as Duplicate FROM
( SELECT *, row_number() OVER (partition by ds, job_id,
 actor_id, event, language, time_spent, org)
as duplicates FROM job_data ) as a ;
```

CONCLUSION:

ds	job_id	actor_id	event	language	time_spent	org	Duplicate
11/25/2020	20	1003	transfer	Italian	45	C	No Duplicate
11/26/2020	23	1004	skip	Persian	56	A	No Duplicate
11/27/2020	11	1007	decision	French	104	D	No Duplicate
11/28/2020	23	1005	transfer	Persian	22	D	No Duplicate
11/28/2020	25	1002	decision	Hindi	11	B	No Duplicate
11/29/2020	23	1003	decision	Persian	20	C	No Duplicate
11/30/2020	21	1001	skip	English	15	A	No Duplicate
11/30/2020	22	1006	transfer	Arabic	25	B	No Duplicate

Case Study 2

Investigating Metric Spike

A. Weekly User Engagement: Write an SQL query to calculate the weekly user engagement.

Code:

```
SELECT week(occurred_at) as Week,
count(DISTINCT user_id)as Weekly_User_engagement
FROM events
GROUP BY week(occurred_at)
ORDER BY week(occurred_at);
```

CONCLUSION:

Week	Weekly_User_engagement
17	740
18	1260
19	1287
20	1351
21	1299
22	1381
23	1446
24	1471
25	1459
26	1509
27	1573
28	1577
29	1607
30	1706
31	1514
32	1454
33	1438
34	1443
35	118

B. User Growth Analysis: Write an SQL query to calculate the user growth for the product.

Code: 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;

CONCLUSION:

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: Write an SQL query to calculate the weekly retention of users based on their sign-up cohort.

- Since the events dataset started from May 2014, I used only users who signed up on or after May 2014.

Code: SELECT user_id,

activated_at

FROM users

WHERE activated_at > '2014-05-01'

ORDER BY user_id;

CONCLUSION:

user_id	activated_at
11768.0	2014-05-01 08:03:12
11770.0	2014-05-01 06:08:50
11775.0	2014-05-01 16:38:06
11778.0	2014-05-01 18:49:49
11779.0	2014-05-01 18:24:54
11780.0	2014-05-01 10:34:04
11785.0	2014-05-01 07:20:37
11787.0	2014-05-01 18:23:18
11791.0	2014-05-01 15:50:33
11793.0	2014-05-01 09:29:48
11795.0	2014-05-01 03:44:18
11798.0	2014-05-01 23:12:34
11799.0	2014-05-01 12:07:10
11801.0	2014-05-01 10:15:50
11804.0	2014-05-01 08:09:58

- Then the user activity was extracted from events column for user who signed up after May 2014

Code:
```
SELECT DISTINCT u.user_id,
e.occurred_at
FROM users u join events e on u.user_id = e.user_id
WHERE u.activated_at > '2014-05-01' and e.event_name = 'login'
GROUP BY week( e.occurred_at)
ORDER BY e.occurred_at;
```

CONCLUSION:

user_id	occurred_at
11768.0	2014-05-01 08:03:12
11775.0	2014-05-09 12:21:24
11787.0	2014-05-13 15:59:11
11799.0	2014-05-20 07:56:03
11778.0	2014-05-28 14:10:56
11901.0	2014-06-03 15:12:16
11778.0	2014-06-09 07:06:11
12741.0	2014-06-20 14:31:18
12882.0	2014-06-22 18:46:04
13441.0	2014-07-04 10:33:32
13942.0	2014-07-12 08:53:20
13757.0	2014-07-18 20:39:28
13317.0	2014-07-20 12:18:05
13743.0	2014-07-30 06:10:40
14237.0	2014-08-04 23:54:39

D. Weekly Engagement Per Device: Write an SQL query to calculate the weekly engagement per device.

Code: 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);

CONCLUSION:

Weeks	device	User_engagement
17	acer aspire desktop	12
17	acer aspire notebook	23
17	amazon fire phone	4
17	asus chromebook	23
17	dell inspiron desktop	20
17	dell inspiron notebook	48
17	hp pavilion desktop	17
17	htc one	19
17	ipad air	29
17	ipad mini	19
17	iphone 4s	27
17	iphone 5	69
17	iphone 5s	47
17	kindle fire	6
17	lenovo thinkpad	94
17	mac mini	7
17	macbook air	61
17	macbook pro	154

E. Email Engagement Analysis: Write an SQL query to calculate the email engagement metrics.

Code: SELECT week(occurred_at) as Week,

count(DISTINCT (CASE WHEN action = "sent_weekly_digest"

THEN user_id end)) as weekly_digest,

```
 count( distinct ( CASE WHEN action = "sent_reengagement_email"
 THEN user_id end )) as reengagement_mail,
 count( distinct ( CASE WHEN action = "email_open"
THEN user_id end )) as opened_email,
count( distinct ( CASE WHEN action = "email_clickthrough"
THEN user_id end )) as email_clickthrough
FROM emails
GROUP BY week(occurred_at)
ORDER BY week(occurred_at);
```

CONCLUSION:

Week	weekly_digest	reengagement_mail	opened_email	email_clickthrough
17	908	73	310	166
18	2602	157	900	425
19	2665	173	961	476
20	2733	191	989	501
21	2822	164	996	436
22	2911	192	965	478
23	3003	197	1057	529
24	3105	226	1136	549
25	3207	196	1084	524
26	3302	219	1149	550
27	3399	213	1207	613
28	3499	213	1228	594
29	3592	213	1201	583
30	3706	231	1363	625
31	3793	222	1338	444