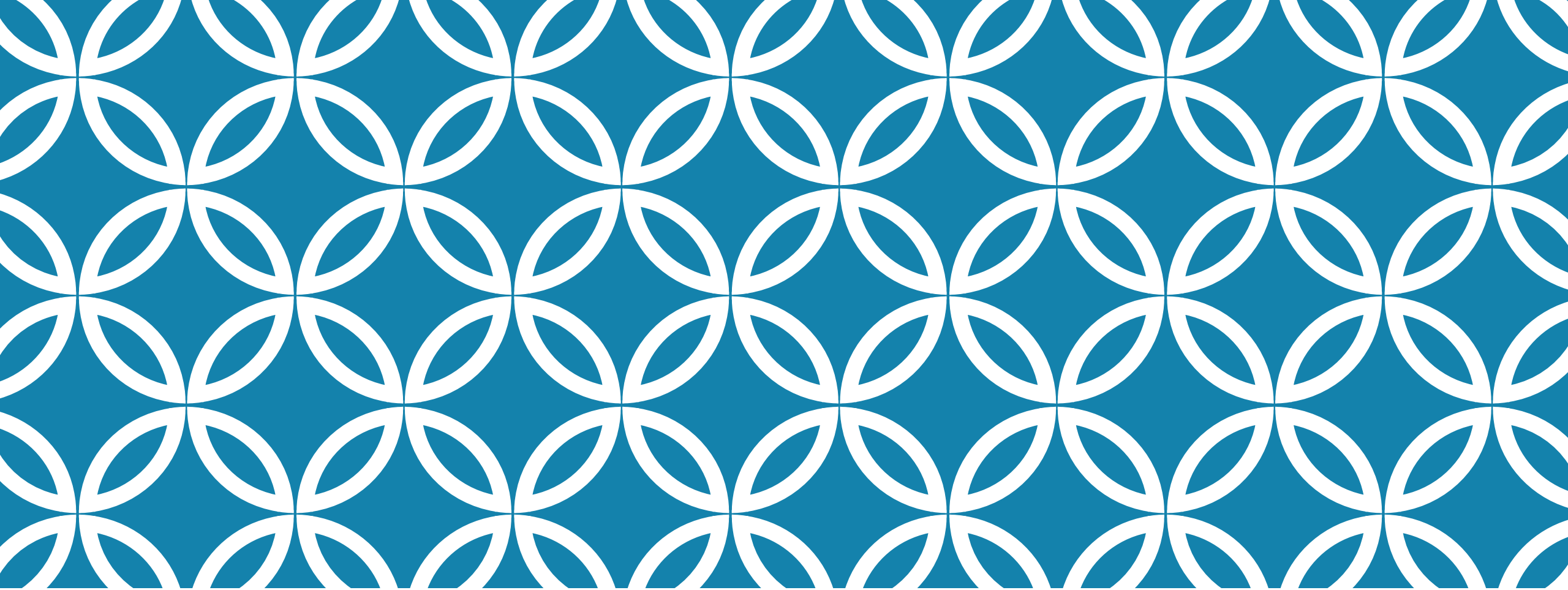

OPERATION ANALYTICS AND INVESTIGATING METRIC SPIKE

By-
Rahul M Ramchandani

ABOUT THE PROJECT

Operational Analytics is a crucial process that involves analyzing a company's end-to-end operations. This analysis helps identify areas for improvement within the company. As a Data Analyst, you'll work closely with various teams, such as operations, support, and marketing, helping them derive valuable insights from the data they collect.

One of the key aspects of Operational Analytics is investigating metric spikes. This involves understanding and explaining sudden changes in key metrics, such as a dip in daily user engagement or a drop in sales. As a Data Analyst, you'll need to answer these questions daily, making it crucial to understand how to investigate these metric spikes.

ABOUT THE PROJECT

SQL TASKS :-

Case Study 1: Job Data Analysis

A. Jobs Reviewed Over Time
B. Throughput Analysis
C. Language Share Analysis
D. Duplicate Rows Detection

ABOUT THE PROJECT

Case Study 2: Investigating Metric Spike

A. Weekly User Engagement
B. User Growth Analysis
C. Weekly Retention Analysis
D. Weekly Engagement Per Device
E. Email Engagement Analysis

Software used :-
MySQL Workbench 8.0 CE

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.

Task: Write an SQL query to calculate the number of jobs reviewed per hour for each day in November 2020.

Query:-

```
select count(distinct job_id)/(30*24) as no_of_jobs_reviewed

from job_data;
```

A. Jobs Reviewed Over Time

Output :-

	no_of_jobs_reviewed_perday_perhour
▶	0.0083

CASE STUDY 1: JOB DATA ANALYSIS

B. Throughput Analysis

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

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 ds,
count(distinct job_id) as jobs_reviewed,
avg(count(distinct job_id)) OVER (ORDER BY ds ROWS BETWEEN 6 PRECEDING AND CURRENT
ROW) as throughput_7_rolling_avg
from job_data
group by ds
order by ds;
```

B. Throughput Analysis

Output :-

	date	jobs_reviewed	throughput_7_rolling_avg
▶	11/25/2020	1	1.0000
	11/26/2020	1	1.0000
	11/27/2020	1	1.0000
	11/28/2020	2	1.2500
	11/29/2020	1	1.2000
	11/30/2020	2	1.3333

CASE STUDY 1: JOB DATA ANALYSIS

C. Language Share Analysis

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

Task: Write an SQL query to calculate the percentage share of each language over the last 30 days.

Query:-

```
SELECT      language, (COUNT(language) * 100.0 / SUM(COUNT(*)) OVER ()) AS
percentage_share

from job_data

group by language;
```

C. Language Share Analysis

Output :-

	language	percentage_share
▶	English	12.50000
	Arabic	12.50000
	Persian	37.50000
	Hindi	12.50000
	French	12.50000
	Italian	12.50000

CASE STUDY 1: JOB DATA ANALYSIS

D. Duplicate Rows Detection

Objective: Identify duplicate rows in the data.

Task: Write an SQL query to display duplicate rows from the job_data table.

Query:-

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

D. Duplicate Rows Detection

Output :-

ds	job_id	actor_id	event	language	time_spent	org	no_of_rows
11/28/2020	23	1005	transfer	Persian	22	D	2
11/26/2020	23	1004	skip	Persian	56	A	3

CASE STUDY 2: INVESTIGATING METRIC SPIKE

A. Weekly User Engagement

Objective: Measure the activeness of users on a weekly basis.

Task: Write an SQL query to calculate the weekly user engagement.

Query:-

```
select extract(week from occured_at) as week_numbers, count(distinct user_id) as
active_users
from events
where event_type='engagement'
group by week_numbersorder by week_numbers;
```

A. Weekly User Engagement

Output :-

week_numbers	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

CASE STUDY 2: INVESTIGATING METRIC SPIKE

B. User Growth Analysis

Objective: Analyze the growth of users over time for a product

Task: Write an SQL query to calculate the user growth for the product.

Query:-

```
SELECT YEAR(u.created_at) AS year,
    MONTH(u.created_at) AS month,
    COUNT(DISTINCT u.user_id) AS new_users
FROM users u
GROUP BY YEAR(u.created_at), MONTH(u.created_at)
ORDER BY YEAR(u.created_at), MONTH(u.created_at);
```

B. User Growth Analysis

Output :-

year	month	new_users
2013	1	160
2013	2	160
2013	3	150
2013	4	181
2013	5	214
2013	6	213
2013	7	284
2013	8	316
2013	9	330
2013	10	390
2013	11	399
2013	12	486
2014	1	552
2014	2	525
2014	3	615
2014	4	726
2014	5	779
2014	6	873
2014	7	997
2014	8	1031

CASE STUDY 2: INVESTIGATING METRIC SPIKE

C. Weekly Retention Analysis:

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

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

Query:-

Query:-

```sql
SELECT
    cohort_week,
    retention_week,
    COUNT(DISTINCT user_id) AS retained_users,
    COUNT(DISTINCT CASE WHEN retention_week = 1 THEN user_id END) AS cohort_size,
    ROUND((COUNT(DISTINCT user_id) / COUNT(DISTINCT CASE WHEN retention_week = 1 THEN user_id END)) * 100, 2) AS retention_rate
FROM (
    SELECT
        users.user_id,
        WEEK(users.created_at) AS cohort_week,
        DATEDIFF(events.occured_at, users.created_at) DIV 7 AS retention_week
    FROM users
    JOIN events ON users.user_id = events.user_id
) AS user_retention
GROUP BY
    cohort_week,
    retention_week
ORDER BY
    cohort_week,
    retention_week;
```

C. Weekly Retention Analysis:

Output :- output was huge as there were 1000 rows. Attaching some of output screenshots-

cohort_week	retention_week	retained_users	cohort_size	retention_rate
0	16	1	0	NULL
0	17	5	0	NULL
0	18	6	0	NULL
0	19	10	0	NULL
0	20	8	0	NULL
0	21	8	0	NULL
0	22	10	0	NULL
0	23	7	0	NULL
0	24	11	0	NULL
0	25	10	0	NULL
0	26	7	0	NULL
0	27	6	0	NULL
0	28	8	0	NULL
0	29	5	0	NULL
0	30	6	0	NULL
0	31	2	0	NULL
0	32	5	0	NULL
0	33	2	0	NULL
0	34	3	0	NULL
0	69	3	0	NULL
0	70	3	0	NULL
0	71	3	0	NULL

cohort_week	retention_week	retained_users	cohort_size	retention_rate
10	14	15	0	NULL
10	15	12	0	NULL
10	16	10	0	NULL
10	17	19	0	NULL
10	18	16	0	NULL
10	19	15	0	NULL
10	20	12	0	NULL
10	21	8	0	NULL
10	22	8	0	NULL
10	23	4	0	NULL
10	24	10	0	NULL
10	59	3	0	NULL
10	60	2	0	NULL
10	61	6	0	NULL
10	62	5	0	NULL
10	63	4	0	NULL
10	64	4	0	NULL
10	65	4	0	NULL
10	66	4	0	NULL
10	67	5	0	NULL
10	68	7	0	NULL
10	69	8	0	NULL

cohort_week	retention_week	retained_users	cohort_size	retention_rate
26	50	7	0	NULL
26	51	4	0	NULL
26	52	5	0	NULL
26	53	6	0	NULL
26	54	4	0	NULL
26	55	5	0	NULL
26	56	5	0	NULL
26	57	5	0	NULL
26	58	4	0	NULL
26	59	3	0	NULL
26	60	4	0	NULL
27	0	222	0	NULL
27	1	113	113	100.00
27	2	86	0	NULL
27	3	67	0	NULL
27	4	44	0	NULL
27	5	32	0	NULL
27	6	24	0	NULL
27	7	20	0	NULL
27	42	3	0	NULL
27	43	7	0	NULL
27	44	6	0	NULL

CASE STUDY 2: INVESTIGATING METRIC SPIKE

D. Weekly Engagement Per Device:

Objective: Measure the activeness of users on a weekly basis per device.
Task: Write an SQL query to calculate the weekly engagement per device.

Query:-

```
SELECT YEAR(occured_at) AS year, WEEK(occured_at) AS week_number, device,
COUNT(*) AS engagement_count
FROM events
GROUP BY YEAR(occured_at),WEEK(occured_at),device
ORDER BY year, week_number, device;
```

D. Weekly Engagement Per Device:

Output :- output was huge. Attaching some of output screenshots-

year	week_number	device	engagement_count
2014	17	acer aspire desktop	69
2014	17	acer aspire notebook	207
2014	17	amazon fire phone	84
2014	17	asus chromebook	254
2014	17	dell inspiron desktop	188
2014	17	dell inspiron notebook	506
2014	17	hp pavilion desktop	134
2014	17	htc one	192
2014	17	ipad air	331
2014	17	ipad mini	208
2014	17	iphone 4s	219
2014	17	iphone 5	715
2014	17	iphone 5s	476
2014	17	kindle fire	57
2014	17	lenovo thinkpad	801
2014	17	mac mini	60
2014	17	macbook air	493
2014	17	macbook pro	1527
2014	17	nexus 10	145
2014	17	nexus 5	385
2014	17	nexus 7	181
2014	17	nokia lumia 635	130

year	week_number	device	engagement_count
2014	24	nexus 5	1160
2014	24	nexus 7	425
2014	24	nokia lumia 635	490
2014	24	samsumg galaxy tablet	101
2014	24	samsung galaxy note	246
2014	24	samsung galaxy s4	957
2014	24	windows surface	215
2014	25	acer aspire desktop	264
2014	25	acer aspire notebook	612
2014	25	amazon fire phone	132
2014	25	asus chromebook	438
2014	25	dell inspiron desktop	654
2014	25	dell inspiron notebook	1229
2014	25	hp pavilion desktop	593
2014	25	htc one	288
2014	25	ipad air	651
2014	25	ipad mini	237
2014	25	iphone 4s	444
2014	25	iphone 5	1661
2014	25	iphone 5s	969
2014	25	kindle fire	211
2014	25	lenovo thinkpad	2118

year	week_number	device	engagement_count
2014	35	asus chromebook	38
2014	35	dell inspiron desktop	5
2014	35	dell inspiron notebook	69
2014	35	hp pavilion desktop	10
2014	35	htc one	19
2014	35	ipad mini	22
2014	35	iphone 4s	58
2014	35	iphone 5	9
2014	35	iphone 5s	22
2014	35	kindle fire	32
2014	35	lenovo thinkpad	126
2014	35	mac mini	25
2014	35	macbook air	66
2014	35	macbook pro	124
2014	35	nexus 10	15
2014	35	nexus 5	35
2014	35	nexus 7	17
2014	35	nokia lumia 635	8
2014	35	samsung galaxy note	6
2014	35	samsung galaxy s4	29
2014	35	windows surface	31

CASE STUDY 2: INVESTIGATING METRIC SPIKE

E. Email Engagement Analysis

Objective: Analyze how users are engaging with the email service.
Task: Write an SQL query to calculate the email engagement metrics.

Query:-

```
SELECT action, COUNT(*) AS total_actions, COUNT(DISTINCT user_id) AS unique_users, COUNT(*) / COUNT(DISTINCT user_id) AS average_actions_per_user
FROM email_events
GROUP BY action;
```

E. Email Engagement Analysis

Output :-

action	total_actions	unique_users	average_actions_per_user
email_clickthrough	9010	5277	1.7074
email_open	20459	5927	3.4518
sent_reengagement_email	3653	3653	1.0000
sent_weekly_digest	57267	4111	13.9302