Operation Analytics and Investigating Metric Spike

**A. Jobs Reviewed Over Time:**

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

-- From the below query we could see that to review one job is taking 1 day on an avg in november month

SELECT

DATE\_FORMAT(ds, '%y%-%m-%d') AS date,

COUNT(\*) AS jobs\_reviewed

FROM job\_data

WHERE month(ds)='11'

GROUP BY 1;

SELECT DISTINCT ds AS days,

(Count(job\_id) \*3600) / (Sum(time\_spent)) AS no\_of\_jobs\_reviewed

FROM asj\_job\_data

GROUP BY days;

**Throughput Analysis:**

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

select a.ds as day, a.throughput,

avg(a.throughput) over( order by DS ROWS BETWEEN 6 preceding AND CURRENT ROW) AS 7\_DAY\_THROUGHPUT\_AVG

FROM

(select ds, count(\*)/sum(time\_spent) as throughput from asj\_job\_data group by ds)a

GROUP BY DS;

**Daily Metric:** Using the daily metric (raw throughput) gives you a snapshot of throughput for each individual day. This is useful for analyzing short-term fluctuations and pinpointing specific days with exceptionally high or low throughput. It's suitable for daily operational insights.

**7-Day Rolling Average:** The 7-day rolling average provides a smoothed, longer-term view of throughput trends. It helps in identifying trends and patterns over a week, which can be useful for understanding weekly cycles, identifying trends, and making longer-term capacity planning and resource allocation decisions.

**C.Language Share Analysis:**

* + Objective: Calculate the percentage share of each language in the last 30 days.
  + Your Task: Write an SQL query to calculate the percentage share of each language over the last 30 days.

SELECT LANGUAGE,

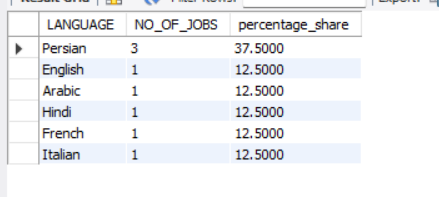
COUNT(JOB\_ID) AS NO\_OF\_JOBS,

COUNT(JOB\_ID)\*100/sum(count(\*)) OVER() as percentage\_share FROM asj\_job\_data

WHERE DS BETWEEN '2020-11-01'AND '2020-11-30'

GROUP BY language

order by percentage\_share desc;



From the above query we could see persian language is double compared with rest all.

**D. Duplicate Rows Detection:**

* + Objective: Identify duplicate rows in the data.
  + Your Task: Write an SQL query to display duplicate rows from the job\_data table.

SELECT DS,JOB\_ID,LANGUAGE,EVENT,ORG,

CASE WHEN COUNT(\*) OVER(partition by DS,JOB\_ID,LANGUAGE,EVENT,ORG) >1 THEN 'DUPLICATE'

ELSE 'NOT DUPLICATE'

END AS DUP\_STATUS

FROM asj\_job\_data;



Hence all are records are distinct

***# Case Study 2: Investigating Metric Spike***

**A. Weekly User Engagement:**

* + Objective: Measure the activeness of users on a weekly basis.
  + Your Task: Write an SQL query to calculate the weekly user engagement.

select extract(week from occurred\_at) as weeks,

COUNT(DISTINCT USER\_ID) AS NO\_OF\_USERS FROM ASJ\_EVENTS

WHERE EVENT\_TYPE='ENGAGEMENT'

GROUP BY 1 ORDER BY 1;

**B.User Growth Analysis:**

* + Objective: Analyze the growth of users over time for a product.
  + Your Task: Write an SQL query to calculate the user growth for the product.

select years, a.weeks as week\_num, a.new\_users as active\_user,

sum(new\_users) over (order by years,weeks rows between unbounded preceding and current row) as cum\_active\_user

from

(

select extract(week from activated\_at) as weeks,

EXTRACT(YEAR FROM activated\_at) AS YEARS,

count(distinct user\_id) as new\_users from ASJ\_USERS

where state like '%active%'

group by 1,2)a;

1. **Weekly Retention Analysis:**
   * Objective: Analyze the retention of users on a weekly basis after signing up for a product.
   * Your Task: Write an SQL query to calculate the weekly retention of users based on their sign-up cohort.

SELECT

DATE\_FORMAT(cohort\_start\_date, '%Y-%m-%d') AS cohort\_week\_start,

DATE\_FORMAT(DATE\_ADD(cohort\_start\_date, INTERVAL 6 DAY), '%Y-%m-%d') AS cohort\_week\_end,

COUNT(DISTINCT u.user\_id) AS cohort\_size,

COUNT(DISTINCT e.user\_id) AS weekly\_retention

FROM (

SELECT

user\_id,

created\_at AS cohort\_start\_date

FROM ASJ\_USERS

) u

LEFT JOIN (

SELECT

user\_id,

occurred\_at AS event\_date

FROM ASJ\_EVENTS1

) e ON u.user\_id = e.user\_id AND event\_date BETWEEN cohort\_start\_date AND DATE\_ADD(cohort\_start\_date, INTERVAL 6 DAY)

GROUP BY cohort\_start\_date

ORDER BY cohort\_start\_date;

**Weekly Engagement Per Device:**

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

SELECT

DATE\_FORMAT(occurred\_week, '%Y-%m-%d') AS week\_start,

DATE\_FORMAT(DATE\_ADD(occurred\_week, INTERVAL 6 DAY), '%Y-%m-%d') AS week\_end,

device,

COUNT(\*) AS engagement\_count

FROM (

SELECT

device,

DATE(occurred\_at) AS occurred\_week

FROM ASJ\_EVENTS

) combined\_events

GROUP BY 1,2,3

ORDER BY week\_start, device;

**Email Engagement Analysis:**

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

SELECT

COUNT(\*) AS total\_emails\_sent,

COUNT(DISTINCT USER\_ID) AS unique\_users\_sent,

SUM(CASE WHEN action = 'sent\_weekly\_digest' THEN 1 ELSE 0 END) AS weekly\_digest\_count,

SUM(CASE WHEN action = 'email\_open' THEN 1 ELSE 0 END) AS emails\_open\_count,

SUM(CASE WHEN action = 'email\_clickthrough' THEN 1 ELSE 0 END) AS email\_clickthrough\_count,

SUM(CASE WHEN action = 'sent\_reengagement\_email' THEN 1 ELSE 0 END) AS sent\_reengagement\_email\_count

FROM ASJ\_EMAIL\_EVENTS;