Operation Analytics and Investigating Metric Spike

Case Study 1: Job Data Analysis

Kiruba Shankar S
Data Analytics Trainee

Job Data Analysis

Overview:

This project aims to leverage Operational Analytics to derive insights from job data. By analyzing various metrics, we aim to improve the company's operations and identify sudden changes in key metrics.

Data:

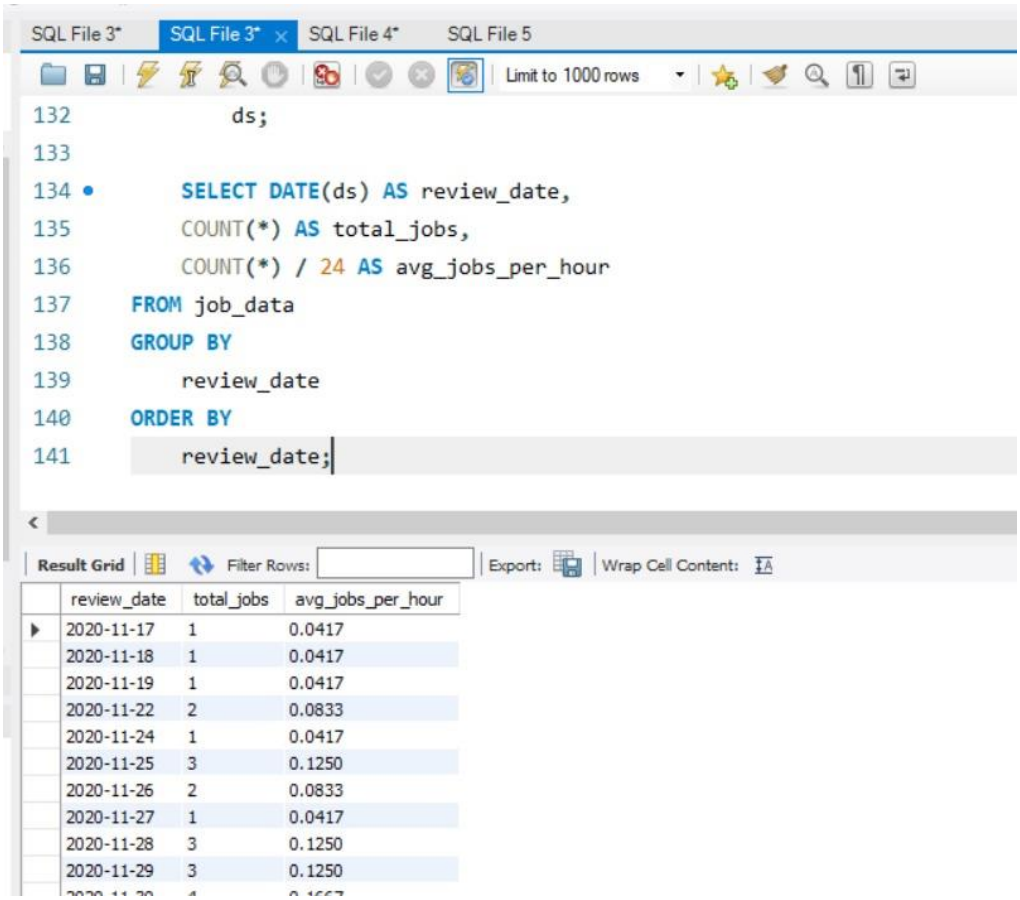
job_id	actor_id	event	language	time_spent	org	ds
21	1001	skip	English	15	A	11/30/2020
22	1006	transfer	Arabic	25	B	11/30/2020
23	1003	decision	Persian	20	C	11/29/2020
23	1005	transfer	Persian	22	D	11/28/2020
25	1002	decision	Hindi	11	B	11/28/2020
11	1007	decision	French	104	D	11/27/2020
23	1004	skip	Persian	56	A	11/26/2020
20	1003	transfer	Italian	45	C	11/25/2020
21	1006	transfer	Hindi	20	E	11/26/2020
29	1008	skip	English	16	A	11/28/2020
29	1002	decision	English	24	A	11/29/2020
23	1004	skip	English	12	A	11/30/2020
21	1004	decision	Hindi	25	A	11/25/2020
22	1007	decision	Hindi	29	C	11/22/2020
22	1010	decision	Arabic	21	C	11/24/2020
29	1009	skip	Spanish	24	D	11/18/2020
22	1003	decision	Persian	29	D	11/22/2020
22	1010	decision	Spanish	20	C	11/29/2020
28	1008	skip	Arabic	25	A	11/19/2020
22	1008	skip	Spanish	11	E	11/17/2020
26	1003	transfer	Spanish	26	E	11/25/2020
23	1008	decision	Hindi	29	C	11/30/2020

From the given data I have added some more data. The given SQL tasks will be performed with the above updated table.

SQL Tasks

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:

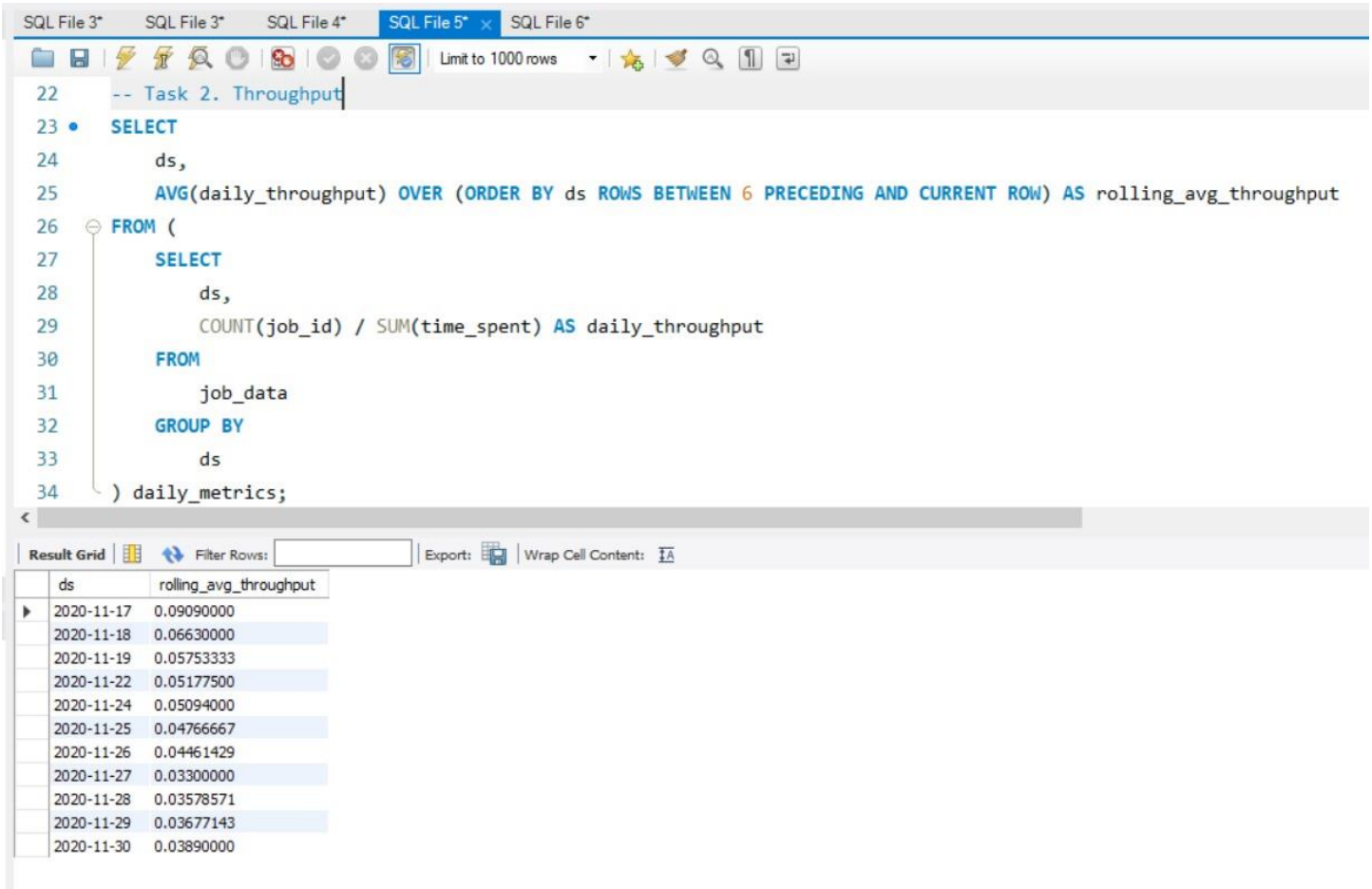
```
SELECT DATE(ds) AS review_date,
COUNT(*) AS total_jobs,
COUNT(*) / 24 AS avg_jobs_per_hour
FROM job_data
GROUP BY
review_date
ORDER BY
review_date;
```

Insights:

- This analysis helps to identify the distribution of job reviews across different hours of the day in November 2020.
- Peak hours and specific days with higher job reviews can be identified, aiding in resource allocation and planning.

B. 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.



ds	rolling_avg_throughput
2020-11-17	0.09090000
2020-11-18	0.06630000
2020-11-19	0.05753333
2020-11-22	0.05177500
2020-11-24	0.05094000
2020-11-25	0.04766667
2020-11-26	0.04461429
2020-11-27	0.03300000
2020-11-28	0.03578571
2020-11-29	0.03677143
2020-11-30	0.03890000

Query:

```
SELECT
   ds,
   AVG(daily_throughput) OVER (ORDER BY ds ROWS BETWEEN 6 PRECEDING AND CURRENT
ROW) AS rolling_avg_throughput
FROM (
   SELECT
      ds,
      COUNT(job_id) / SUM(time_spent) AS daily_throughput
   FROM
      job_data
   GROUP BY
      ds
) daily_metrics;
```

Explanation:

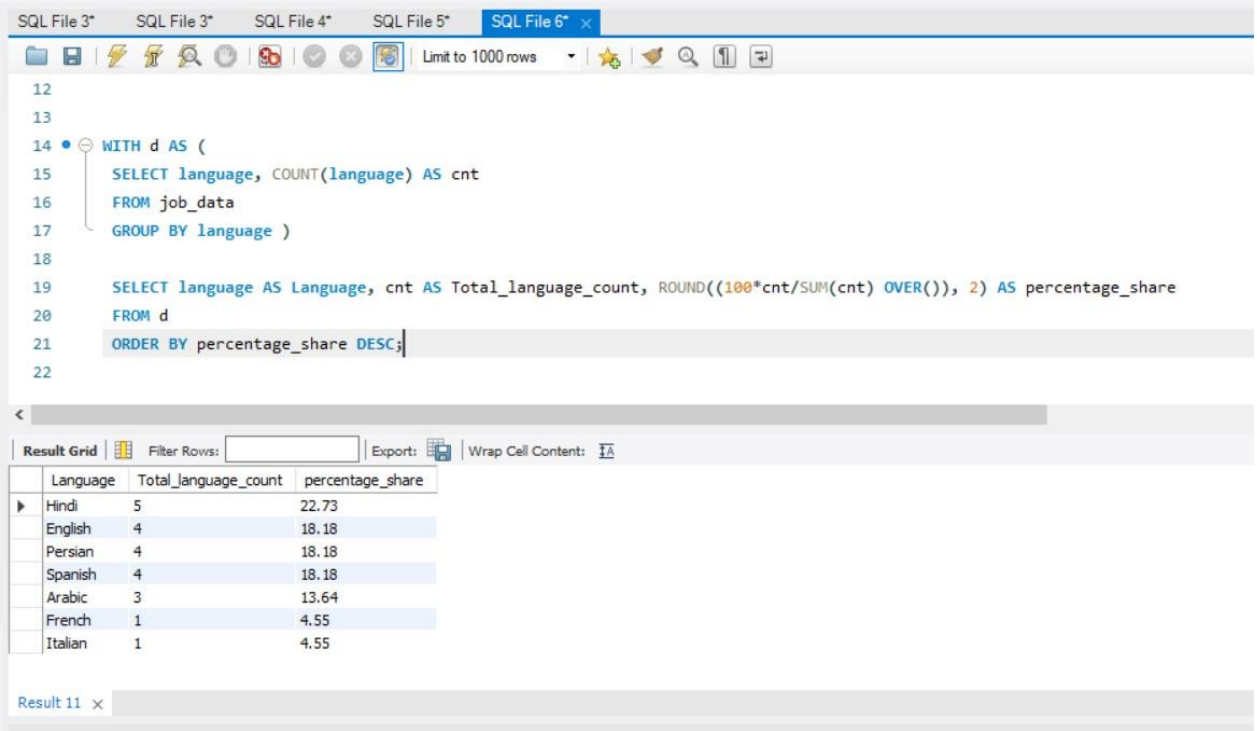
Using the 7-day rolling average provides a smoother trend line that can help identify underlying patterns, removing daily volatility.

Insights:

- Calculated the 7-day rolling average of throughput, which provides a smoother trend compared to daily metrics, revealing consistent performance trends.
- This approach helps in identifying underlying patterns by removing daily volatility, making it easier to spot long-term trends and anomalies.

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.



Query:

```
WITH d AS (  SELECT language, COUNT(language) AS cnt  FROM job_data
 GROUP BY language )
      SELECT language AS Language, cnt AS Total_language_count, ROUND((100*cnt/SUM(cnt)
OVER()), 2) AS percentage_share  FROM d  ORDER BY percentage_share DESC;
```

Insights:

- From the table and the plot, we can observe that Hindi language is the most used language with percentage share of 22.73% followed by English, Persian and Spanish with 18.18% and Arabic with 13.64%.

- Although Arabic is the most used language, other languages also have significant usage percentages.

D. Duplicate Rows Detection:

Objective: Identify duplicate rows in the data.

For the 'Duplicate Rows Detection' task there are no duplicate rows so I have inserted duplicate rows using the below query.

```
-- inserting duplicate data...
INSERT INTO job_data (job_id, actor_id, event, language, time_spent, org, ds)
SELECT job_id, actor_id, event, language, time_spent, org, ds
FROM job_data;
```

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

SQL File 3* | SQL File 3* | SQL File 4* | SQL File 5* | SQL File 6* | SQL File 7*

Limit to 1000 rows

```
3  Select count(*) from job_data;
4
5  select
6      job_id,
7      actor_id,
8      event,
9      language,
10     time_spent,
11     org,
12     ds,
13     row_number() over (partition by job_id, actor_id, event, language, time_spent, org, ds order by job_id) as 'duplicate count'
14 from job_data ;
15
```

Result Grid | Filter Rows: | Export: | Wrap Cell Content:

job_id	actor_id	event	language	time_spent	org	ds	duplicate count
11	1007	decision	French	104	D	2020-11-27	1
11	1007	decision	French	104	D	2020-11-27	2
20	1003	transfer	Italian	45	C	2020-11-25	1
20	1003	transfer	Italian	45	C	2020-11-25	2
21	1001	skip	English	15	A	2020-11-30	1
21	1001	skip	English	15	A	2020-11-30	2
21	1004	decision	Hindi	25	A	2020-11-25	1
21	1004	decision	Hindi	25	A	2020-11-25	2
21	1006	transfer	Hindi	20	E	2020-11-26	1
21	1006	transfer	Hindi	20	E	2020-11-26	2
22	1003	decision	Persian	29	D	2020-11-22	1
22	1003	decision	Persian	29	D	2020-11-22	2
22	1006	transfer	Arabic	25	B	2020-11-30	1
22	1006	transfer	Arabic	25	B	2020-11-30	2
22	1007	decision	Hindi	29	C	2020-11-22	1
22	1007	decision	Hindi	29	C	2020-11-22	2
22	1008	skip	Spanish	11	E	2020-11-17	1

Result 16

Query:

```
select
        job_id,
   actor_id,
   event,
   language,
   time_spent,
   org,
   ds,
   row_number() over (partition by job_id, actor_id, event, language, time_spent, org, ds

order by job_id)
as 'duplicate count'
from job_data ;
```

Insights:

- This query identifies duplicate rows in the dataset, ensuring data quality and accuracy in further analysis.
- After inserting duplicate data, the query can effectively display duplicate rows, helping maintain the integrity of the dataset.

Conclusion:

The analysis of job data provides valuable operational insights: identifying peak hours and days for job reviews facilitates better resource allocation, while the 7-day rolling average throughput offers a clearer view of performance trends by smoothing out daily volatility. The language share analysis highlights the multilingual nature of the data, indicating preferences and usage patterns. Additionally, detecting duplicate rows ensures data integrity, which is crucial for accurate analysis. These insights collectively support enhanced decision-making, efficient planning, and optimized operational strategies.