

# Operation Analytics and Investigating Metric Spike

## Description:

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.

In this project, you'll take on the role of a Lead Data Analyst at a company like Microsoft. You'll be provided with various datasets and tables, and your task will be to derive insights from this data to answer questions posed by different departments within the company. Your goal is to use your advanced SQL skills to analyze the data and provide valuable insights that can help improve the company's operations and understand sudden changes in key metrics.

## Tasks:

### 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.

## Query:

The screenshot shows a SQL IDE interface with two tabs: 'SQL File 4\*' and 'SQL File 3\*'. The 'SQL File 3\*' tab is active, displaying a SQL query. The query includes several INSERT statements for job data, followed by a DROP TABLE statement, a CREATE TABLE statement for 'job\_data', a TRUNCATE TABLE statement, and a SELECT statement to calculate the number of jobs reviewed per day in November 2020.

```
42 ('2020-11-08', 3, 1024, 'transfer', 'Gujarati', 37, 'A'),
43 ('2020-11-07', 2, 1025, 'decision', 'Malayalam', 48, 'B'),
44 ('2020-11-06', 1, 1026, 'skip', 'Kannada', 59, 'C'),
45 ('2020-11-05', 0, 1027, 'transfer', 'Odia', 70, 'D'),
46 ('2020-11-04', 29, 1028, 'decision', 'Assamese', 81, 'A'),
47 ('2020-11-03', 28, 1029, 'skip', 'Maithili', 92, 'B');
48
49 • drop table job_data;
50 • create table job_data(ds date,job_id int,actor_id int,event varchar(10),language varchar(10),t
51 • truncate table job_data;
52
53 • select ds,count(job_id) as jobs_reviewed from job_data group by ds;
```

Below the query editor, the 'Result Grid' is visible, showing the results of the query. The grid has two columns: 'ds' and 'jobs\_reviewed'. The results are as follows:

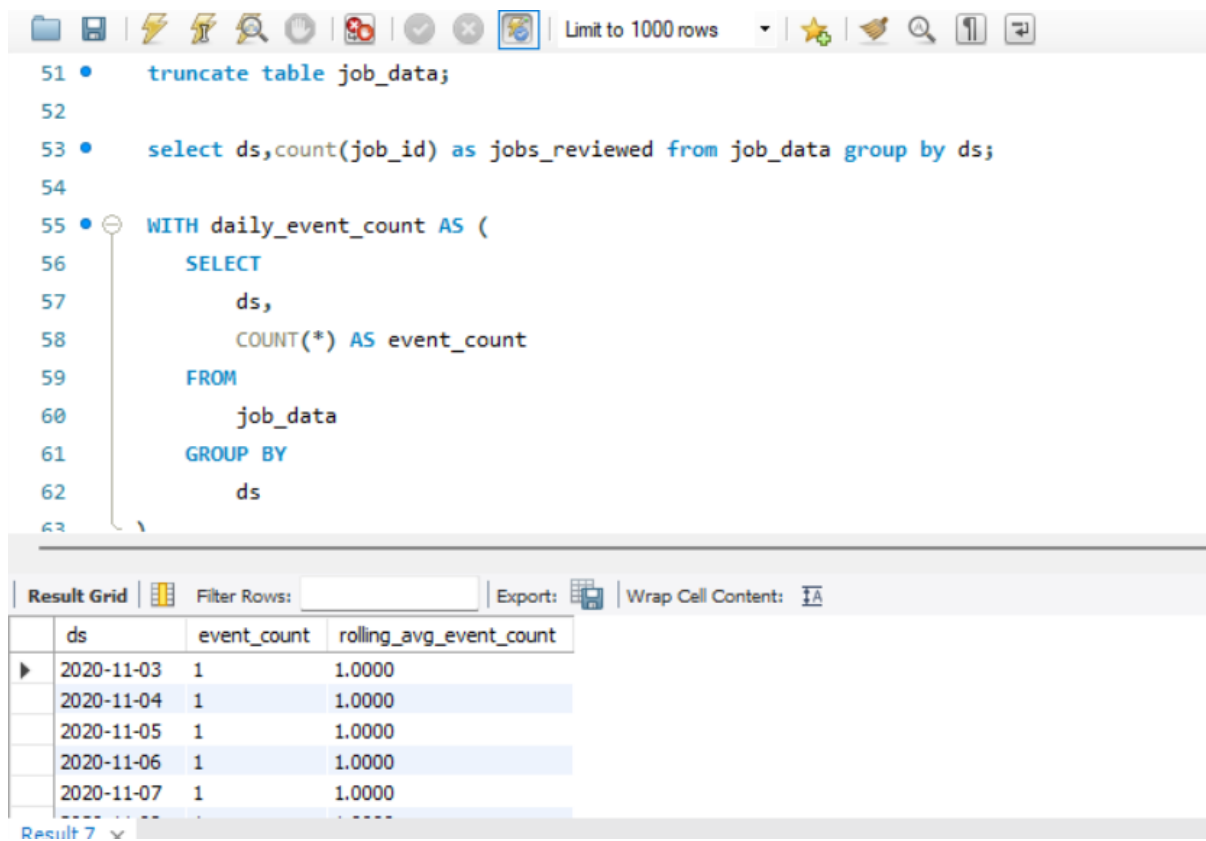
ds	jobs_reviewed
2020-11-30	2
2020-11-29	1
2020-11-28	2
2020-11-27	1
2020-11-26	1

The interface also includes a 'Filter Rows' section, an 'Export' button, and a 'Wrap Cell Content' option. The 'Result Grid' is currently in 'Read Only' mode.

## 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.

## Query:



The screenshot shows a SQL IDE interface. The query editor contains the following SQL code:

```
51 • truncate table job_data;
52
53 • select ds,count(job_id) as jobs_reviewed from job_data group by ds;
54
55 • WITH daily_event_count AS (
56     SELECT
57         ds,
58         COUNT(*) AS event_count
59     FROM
60         job_data
61     GROUP BY
62         ds
63 )
```

The results pane shows a table with the following data:

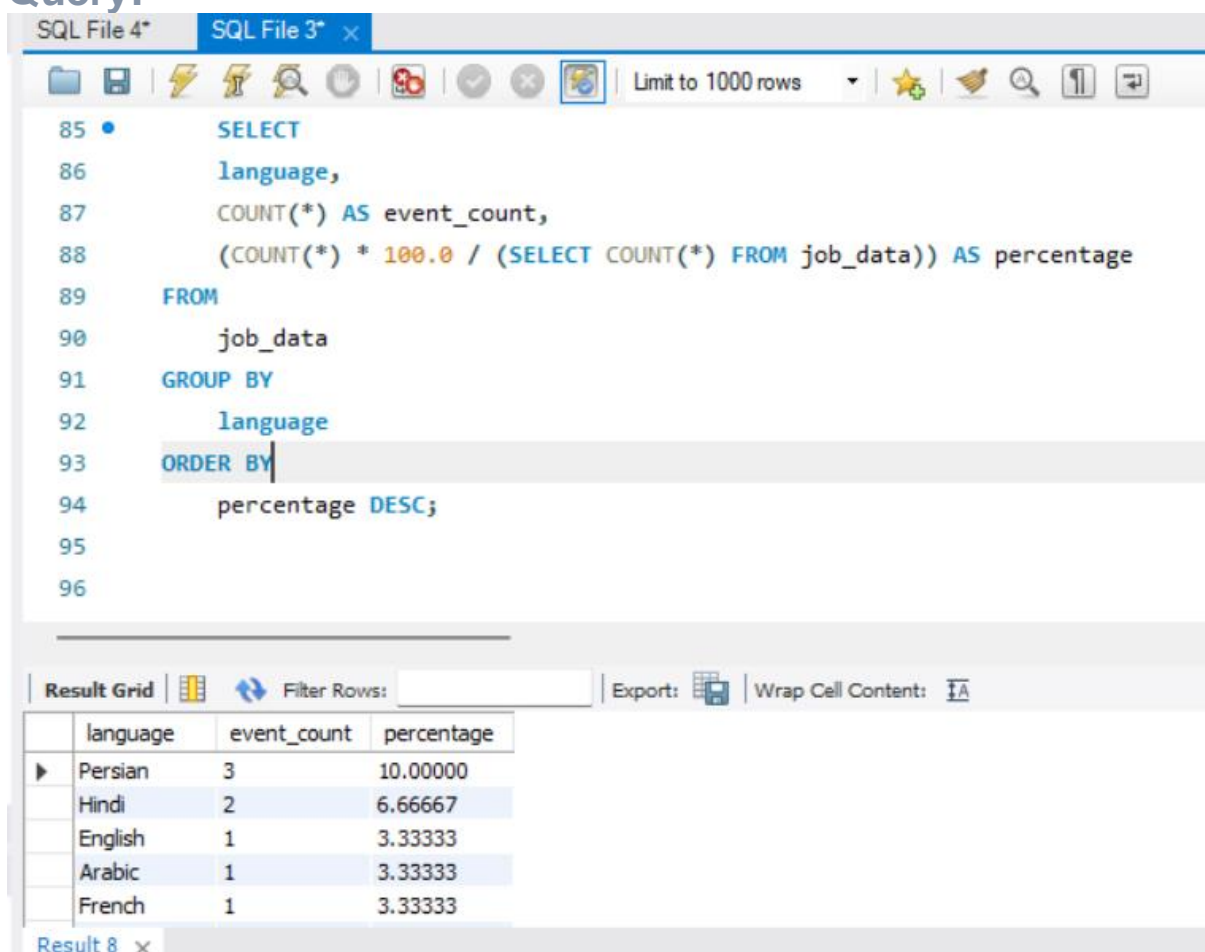
ds	event_count	rolling_avg_event_count
2020-11-03	1	1.0000
2020-11-04	1	1.0000
2020-11-05	1	1.0000
2020-11-06	1	1.0000
2020-11-07	1	1.0000

The interface includes a toolbar at the top with various icons for file operations, execution, and formatting. The results pane also has options for filtering rows, exporting, and wrapping cell content.

## 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:



The screenshot shows a SQL IDE interface with a query editor and a results grid. The query editor contains the following SQL code:

```
85 • SELECT
86     language,
87     COUNT(*) AS event_count,
88     (COUNT(*) * 100.0 / (SELECT COUNT(*) FROM job_data)) AS percentage
89 FROM
90     job_data
91 GROUP BY
92     language
93 ORDER BY
94     percentage DESC;
95
96
```

The results grid displays the following data:

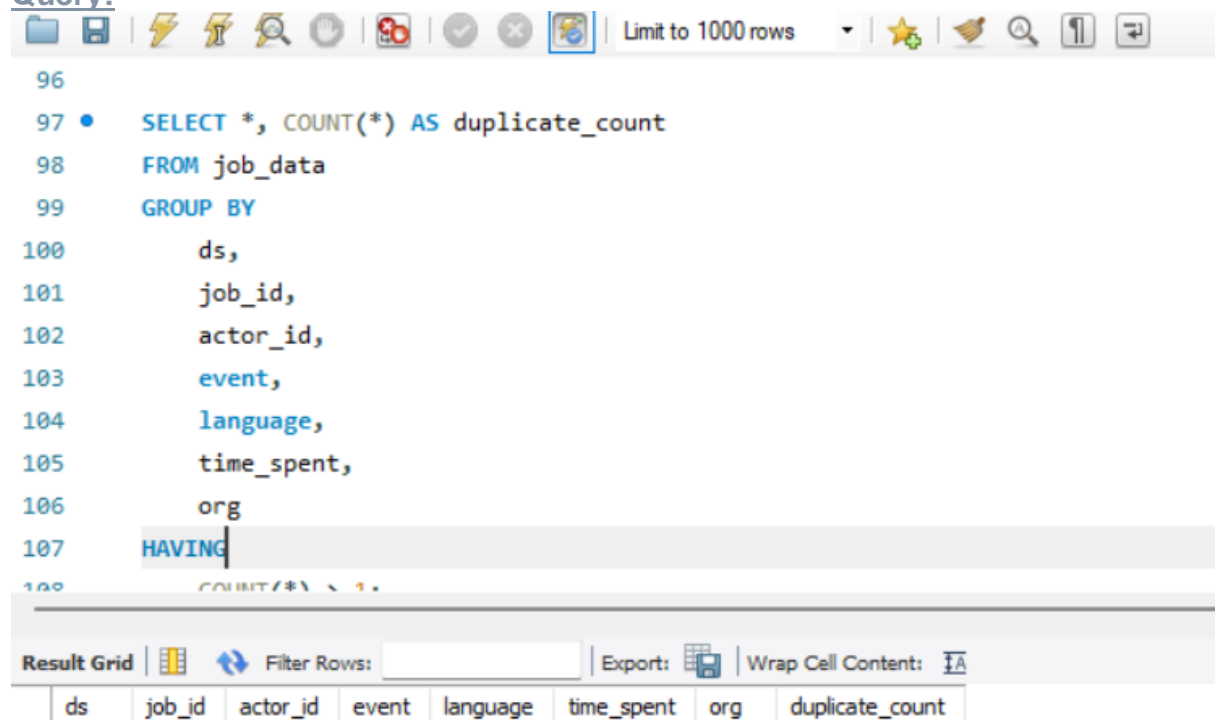
language	event_count	percentage
Persian	3	10.00000
Hindi	2	6.66667
English	1	3.33333
Arabic	1	3.33333
French	1	3.33333

The interface also includes a toolbar with various icons and a status bar at the bottom showing "Result 8 x".

## 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.

Query:



The screenshot shows a SQL query editor interface. At the top, there is a toolbar with various icons for file operations, execution, and navigation. Below the toolbar, the query text is displayed with line numbers on the left. The query is as follows:

```
96
97 • SELECT *, COUNT(*) AS duplicate_count
98 FROM job_data
99 GROUP BY
100     ds,
101     job_id,
102     actor_id,
103     event,
104     language,
105     time_spent,
106     org
107 HAVING
108     COUNT(*) > 1.
```

Below the query editor, there is a section for the results. It includes a toolbar with options like 'Result Grid', 'Filter Rows', 'Export', and 'Wrap Cell Content'. Below this toolbar, a table header is visible with the following columns:

ds	job_id	actor_id	event	language	time_spent	org	duplicate_count
----	--------	----------	-------	----------	------------	-----	-----------------

