**Writing Advanced Queries for Real-Life Use Cases**

**Objective:** To write complex PromQL queries for detecting performance issues, monitoring resource utilization, and setting up alerts

**Tools required:** Linux operating system

**Prerequisites:** Refer to Demo 02 of Lesson 02 for understanding basic PromQL queries

Steps to be followed:

1. Set alerts for high memory usage
2. Analyze disk I/O patterns for performance issues
3. Track node uptime for maintenance alerts
4. Monitor network traffic patterns to identify bottlenecks

**Step 1: Set alerts for high memory usage**

1. Execute the following query in the expression browser to identify nodes with high memory utilization:

**100 \* (1 - ((node\_memory\_MemFree\_bytes + node\_memory\_Cached\_bytes + node\_memory\_Buffers\_bytes) / node\_memory\_MemTotal\_bytes))**

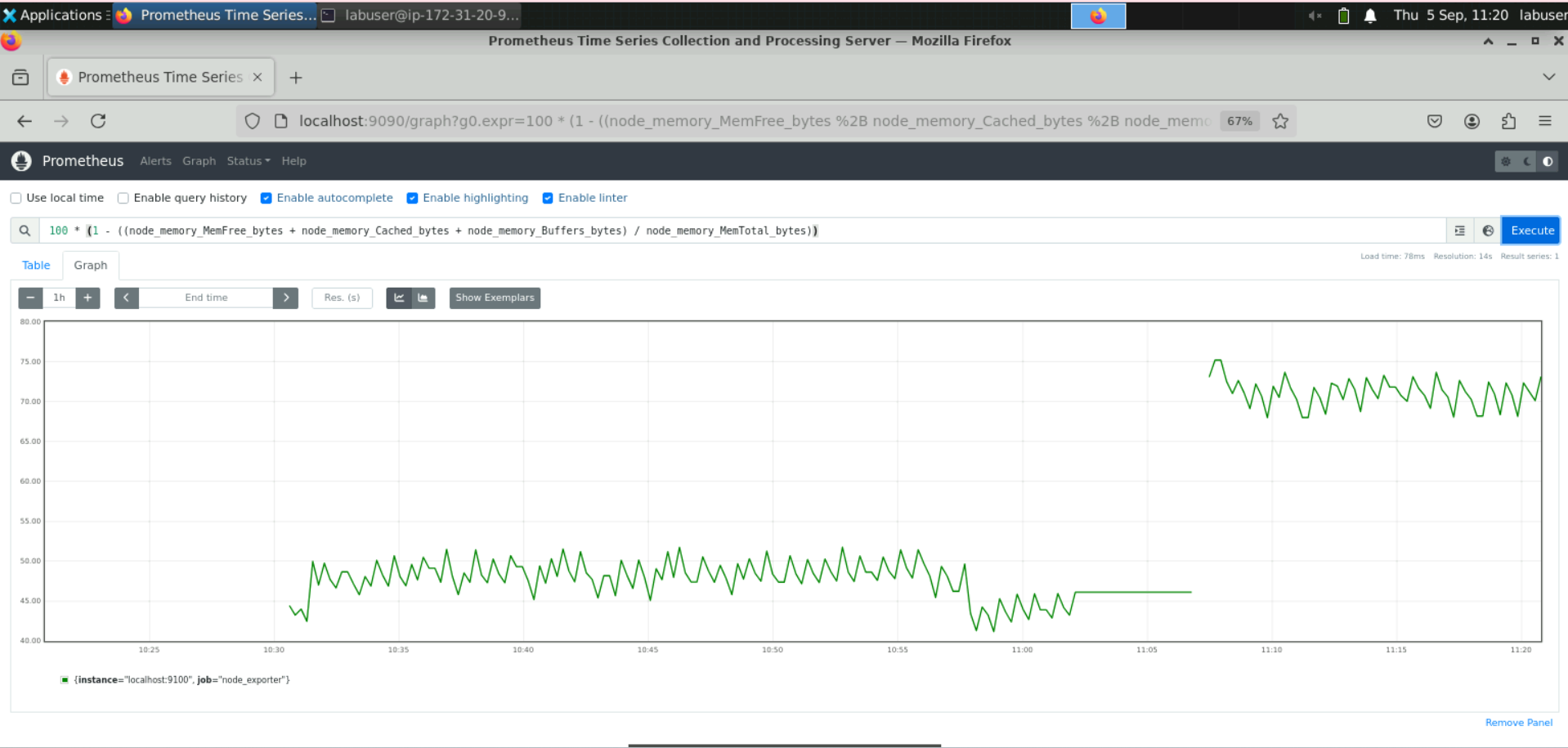
**Explanation:**

**This query calculates the percentage of memory used on a system, considering memory that is "free," "cached," or "buffered" as available.**

1. **node\_memory\_MemFree\_bytes:**
   * **Amount of free memory in bytes.**
2. **node\_memory\_Cached\_bytes:**
   * **Memory in the page cache that can be reclaimed if needed.**
3. **node\_memory\_Buffers\_bytes:**
   * **Memory used for temporary buffers (e.g., disk I/O).**
4. **node\_memory\_MemTotal\_bytes:**
   * **Total physical memory available in the system.**
5. **Formula Breakdown:**
   * **(node\_memory\_MemFree\_bytes + node\_memory\_Cached\_bytes + node\_memory\_Buffers\_bytes): Total available memory (including free, cached, and buffered memory).**
   * **... / node\_memory\_MemTotal\_bytes: Proportion of memory that is available relative to the total memory.**
   * **1 - ...: Proportion of memory that is used.**
   * **100 \* ...: Converts the result into a percentage.**

**Practical Meaning:**

**This query provides the percentage of memory currently being used on the system, factoring in cached and buffered memory as available. It gives a more accurate representation of memory utilization than simply using MemFree.**



**Note:** This query calculates the memory utilization percentage for each node by subtracting the ratio of available memory from the total memory. An alert rule can also be set to notify when memory utilization exceeds a defined threshold.

**Step 2: Analyze disk I/O patterns for performance issues**

1. Execute the following query to analyze the disk I/O patterns across the infrastructure:

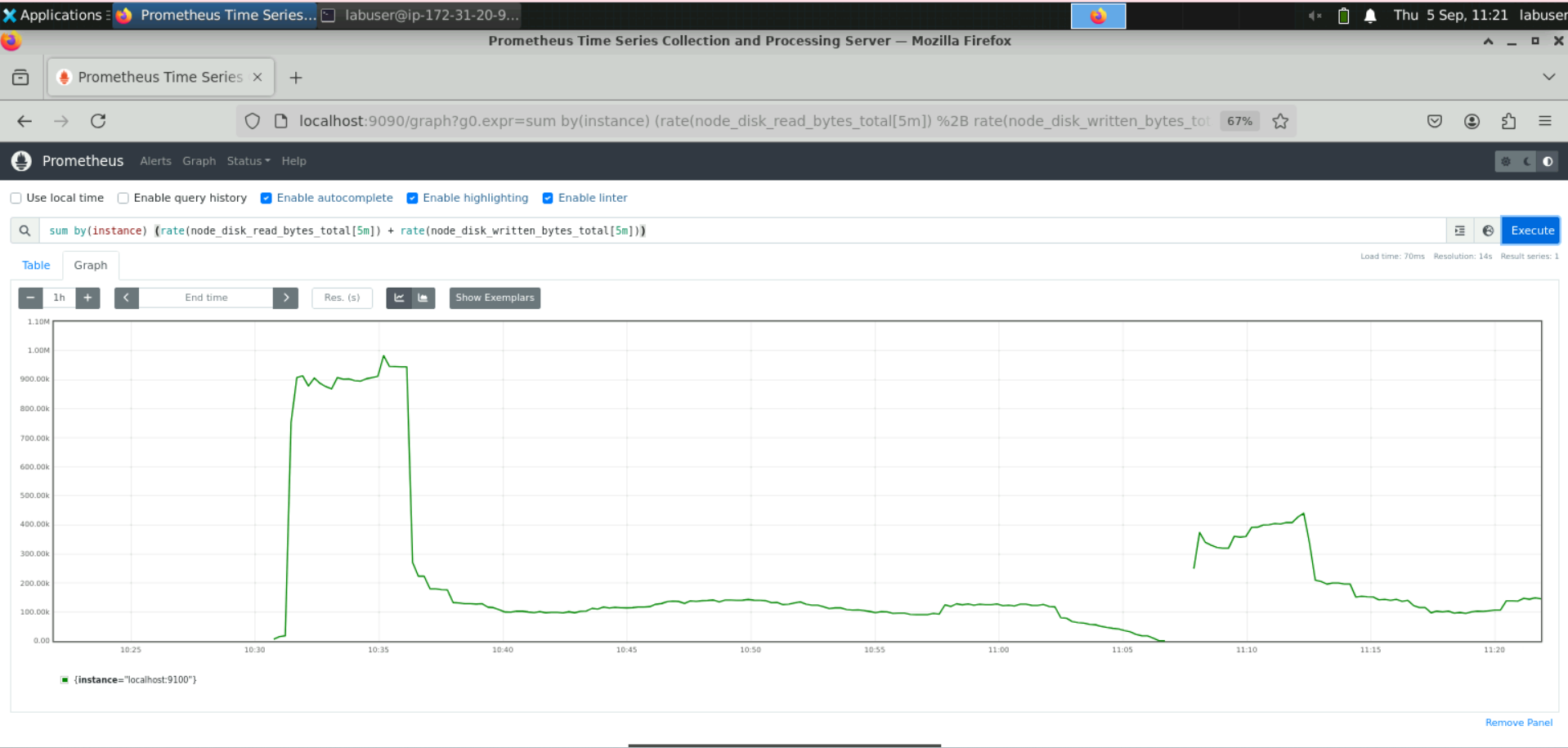
**sum by(instance) (rate(node\_disk\_read\_bytes\_total[5m]) + rate(node\_disk\_written\_bytes\_total[5m]))**

**Explanation:**

1. **node\_disk\_read\_bytes\_total:**
   * **Cumulative counter for the total number of bytes read from disk.**
2. **node\_disk\_written\_bytes\_total:**
   * **Cumulative counter for the total number of bytes written to disk.**
3. **rate(metric[5m]):**
   * **Calculates the per-second rate of change of the metric over a 5-minute window. This gives the read/write throughput in bytes per second.**
4. **rate(node\_disk\_read\_bytes\_total[5m]) + rate(node\_disk\_written\_bytes\_total[5m]):**
   * **Adds the per-second read and write rates to compute the total disk I/O rate for a single disk.**
5. **sum by(instance):**
   * **Aggregates the total disk I/O rate across all disks for each instance (node).**

**Practical Meaning:**

**This query calculates the total disk I/O throughput (read + write) in bytes per second for each instance (node) over the last 5 minutes.**



**Note:** This query sums up the per-second disk read and write rates for each instance over the last 5 minutes, grouped by instance label. It helps identify unusual spikes or trends in disk I/O activity across nodes or the infrastructure.