
📑 Analyzing Apache Access Logs Over Time in Sumo Logic

🔹 Query 1: Tracking Overall Traffic Changes

Query:

```
_sourceCategory=Labs/Apache/Access
| timeslice 1m
| count by _timeslice
| sort by _timeslice asc
| diff _count
```

✅ Explanation

1. **timeslice 1m**
   - Groups logs into **1-minute intervals**.
   - Each _timeslice bucket represents all logs received in that minute.
2. **count by _timeslice**
   - Counts the number of logs in each 1-minute bucket.
   - Gives you total log volume per minute.
3. **sort by _timeslice asc**
   - Ensures the results are displayed in chronological order.
4. **diff _count** ⭐
   - Calculates the **difference between consecutive rows** for the _count field.
   - This shows how log volume changes from one minute to the next.

Example:

_timeslice	_count	diff(_count)
10:01	100	-
10:02	120	20
10:03	90	-30
10:04	150	60

- At 10:02, traffic increased by **20 logs**.
- At 10:03, traffic dropped by **30 logs**.

📌 **Why Useful?**

- Helps identify **spikes or drops** in traffic.
- Can indicate issues like sudden error bursts, attacks, or unusual user behavior.

🔹 **Query 2: Tracking 404 Errors with Rolling Average**

Query:

_sourceCategory=Labs/Apache/Access and status_code=404

| timeslice 1m

| count as error_count by _timeslice

| sort by _timeslice asc

| smooth error_count as rolling_avg

✅ **Explanation**

1. **status_code=404**
   - Filters only logs where HTTP status = **404 Not Found**.
2. **timeslice 1m**
   - Groups these error logs into **1-minute intervals**.
3. **count as error_count by _timeslice**
   - Counts the number of 404 errors per minute.
4. **sort by _timeslice asc**
   - Displays results chronologically.
5. **smooth error_count as rolling_avg** ⭐
   - Applies a **moving average** (rolling average) to smooth out fluctuations.
   - Instead of raw spikes, you see a **trend line** of errors.

Example:

_timeslice error_count rolling_avg

10:01 5 -

10:02	7	6
10:03	2	4.67
10:04	8	5.67

📌 **Why Useful?**

- Raw error counts can be **noisy** (sudden spikes/drops).
- smooth helps identify **overall trends** in error rates.
- Useful for spotting **gradual increases in 404s** (e.g., broken links, bots scanning).

📝 **General Explanation of diff and smooth**

- **diff Operator**
  - Calculates the **difference between values in consecutive rows** of your results.
  - Helps you see **how fast something is changing** (increase/decrease).
  - Best for detecting **spikes, drops, or sudden shifts** in logs or metrics.
  - Example use cases: traffic surges, sudden error bursts, unusual login attempts.
- **smooth Operator**
  - Applies a **rolling (moving) average** across data points.
  - Reduces **noise** in the results by balancing out sudden spikes.
  - Best for spotting **trends over time** rather than reacting to small fluctuations.
  - Example use cases: tracking long-term error rates, monitoring gradual increases in latency, identifying steady attack patterns.

📌 **In short:**

- diff shows **short-term changes** (what just changed compared to the last point).
- smooth shows **long-term trends** (what's the overall direction).
