Troubleshoot TestData data source issues

- Query errors
  - "No data" or empty panels
  - Exemplars don't appear
  - Wrong data type for the visualization
  - Predictable Pulse pattern doesn't match expectations
  - Slow Query appears stuck
  - Unexpected error from Conditional Error
  - Error with source returns an error
  - Flaky Query returns errors intermittently
- CSV and data input errors
  - CSV Content or CSV Metric Values return unexpected results
  - Raw Frames JSON parse errors
- Streaming errors
  - Streaming Client doesn't update
  - Grafana Live channel shows no data
- Template variable errors
  - Variable drop-down is empty
  - Chained variable doesn't update
  - Variable value not interpolated in queries
- Alerting errors
  - Alert rule shows "no data" unexpectedly
  - Alert rule never fires
  - Template variables don't resolve in alert rules
  - Alert evaluation timeout
- Enable debug logging
- Get additional help

This document provides solutions to common issues you may encounter when using the TestData data source. TestData is a built-in data source with no external dependencies, so most issues relate to scenario configuration or data formatting.

Query errors

These errors occur when running TestData scenarios.

"No data" or empty panels

Symptoms:

- Panel displays "No data" message.
- Query executes without error but returns nothing.

Possible causes and solutions:

Cause	Solution
No Data Points scenario selected	The No Data Points scenario returns empty results by design. Select a different scenario such as Random Walk.
Datapoints Outside Range selected	This scenario returns a data point one hour before the query time range. Expand the time range or select a different scenario.
Empty String Input for CSV scenarios	CSV Metric Values and CSV Content require input data. Add comma-separated values or CSV content.
Time range doesn't contain data	Some scenarios generate data relative to the query time range. Expand the dashboard time range.

Exemplars don't appear

Symptoms:

- The Exemplars scenario runs without error but no diamond markers appear on the panel.
- Markers disappear when you interact with the legend.

Possible causes and solutions:

Cause	Solution

No series query in the panel	Exemplars returns only markers and no series of its own. Add a second query, such as Random Walk, for the exemplars to annotate.
Series hidden in the legend	The legend filter matches on exemplar labels, so hiding a series in the legend also hides its markers. Show the series to restore the markers.
Markers fall outside the panel value range	With **Min** and **Max** empty, the value range is derived from a reference random walk and markers can land off-panel. Set **Min** and **Max** explicitly, or align them with the sibling series.

Wrong data type for the visualization

Symptoms:

- Panel shows "Data does not have a time field" or similar type mismatch errors.
- Visualization renders but looks wrong (for example, a table when you expected a graph).

Solutions:

1. Match the scenario to the visualization type. For example, use the Logs scenario with the Logs panel, Node Graph with the Node Graph panel, and Trace with the Traces panel.
2. Time-series panels require scenarios that produce time-series data (Random Walk, Predictable Pulse, CSV Metric Values, Predictable CSV Wave).
3. For table panels, use Table Static, Random Walk Table, or CSV File.

Predictable Pulse pattern doesn't match expectations

Symptoms:

- The on/off pattern doesn't align with expected times.
- Values appear at unexpected intervals.

Solutions:

1. Predictable Pulse aligns timestamps to the step interval based on absolute time from the epoch, not from the start of the time range. This means the pattern starts at the same point regardless of when you load the dashboard.
2. Verify the cycle duration matches your intent: the full cycle is `Step * (On Count + Off Count)` seconds.

3. Check that **On Value** and **Off Value** are set correctly. The defaults are `2` and `1`, not `1` and `0`.

Slow Query appears stuck

Symptoms:

- Query runs for an extended period.
- Panel shows a loading spinner indefinitely.

Solutions:

1. Check the **String Input** field, which controls the delay duration. The default is `5s`.
2. Reduce the value to a shorter duration (for example, `1s` or `500ms`).
3. The field accepts Go duration syntax: `5s` (seconds), `1m` (minutes), `500ms` (milliseconds).

Unexpected error from Conditional Error

Symptoms:

- Query returns a server error or panic message.
- Error appears immediately when running the query.

Solutions:

1. The Conditional Error scenario triggers a server panic when the **String Input** field is empty. This is the intended behavior for error testing.
2. To return data instead of an error, populate the **String Input** field with comma-separated values (for example, `1,20,90,30,5,0`).
3. To change the error type, use the **Error type** drop-down to select between Server panic, Frontend exception, and Frontend observable.

Error with source returns an error

The Error with source scenario intentionally returns errors for testing how Grafana handles different error sources. This isn't a misconfiguration.

- **Plugin** error source simulates an error originating from the plugin itself.
- **Downstream** error source simulates an error from a downstream service.

Use this scenario to test alerting rules, error displays, and error handling in custom panels.

Flaky Query returns errors intermittently

The Flaky Query scenario returns errors for a configurable percentage of requests. Intermittent errors are the intended behavior, not a misconfiguration.

Solutions:

1. Adjust the **Error rate** field to control how often requests fail. Set it to `0` for no errors or `100` to fail every request.
2. Use the **Status code** and **Error source** fields to shape the returned error.
3. If you're using Flaky Query with an alert rule, be aware that intermittent failures can flip the rule between states across evaluations. For alerting behavior, refer to TestData alerting.

CSV and data input errors

These errors occur with scenarios that accept user-provided data.

CSV Content or CSV Metric Values return unexpected results

Symptoms:

- Data doesn't match expectations.
- Fewer data points than expected.
- Parsing errors.

Solutions:

1. Verify CSV formatting: headers in the first row, commas as delimiters, numeric values unquoted.
2. For CSV Metric Values, enter only comma-separated numbers in the **String Input** field (for example, `1,20,90,30,5,0`).
3. The special values `null` and `nan` are supported in Predictable CSV Wave values. Use `null` for missing data points and `nan` for Not-a-Number values.
4. Check for trailing commas or extra whitespace in your input.

Raw Frames JSON parse errors

Symptoms:

- Error message about invalid JSON.
- Panel fails to render.

Solutions:

1. Validate that the JSON matches the Grafana data frame format.
2. Use the paste helpers in the editor to import data from panel JSON or raw query results.
3. Verify that field types are consistent within each field array.

Streaming errors

These errors occur with the Streaming Client and Grafana Live scenarios.

Streaming Client doesn't update

Symptoms:

- Panel shows initial data but doesn't update.
- No new data points appear over time.

Solutions:

1. Verify the **Speed** field is set to a reasonable value (in milliseconds). Lower values produce faster updates.
2. For the **Fetch** type, verify the URL is accessible from the browser.
3. Check the browser console for WebSocket or network errors.

Grafana Live channel shows no data

Symptoms:

- Panel remains empty after selecting a channel.
- No streaming data appears.

Solutions:

1. Confirm the selected channel matches one of the available options: `random-2s-stream`, `random-flakey-stream`, `random-labeled-stream`, or `random-20Hz-stream`.
2. Verify that Grafana Live is enabled in your Grafana instance.

3. Check the Grafana server logs for Live connection errors.

Template variable errors

These errors occur when using TestData with dashboard template variables.

Variable drop-down is empty

Symptoms:

- Variable drop-down shows no options.
- Preview in the variable editor returns zero values.

Solutions:

1. Verify the query string navigates valid nodes in the metric tree. Use `*` to return all top-level nodes.
2. Check that dot-separated path segments match the tree structure (for example, `A.*` returns children of node A). An invalid path like `Z.*` returns nothing because `Z` doesn't exist in the tree.
3. Verify the TestData data source is selected in the variable editor.
4. Click **Run query** in the variable editor preview to test.

Chained variable doesn't update

Symptoms:

- A dependent variable (for example, `$region.*`) doesn't refresh when the parent variable changes.
- The drop-down shows stale values from a previous selection.

Solutions:

1. Set the dependent variable's **Refresh** option to **On dashboard load** or **On time range change** in the variable editor.
2. Verify the query uses the correct parent variable syntax (for example, `$region.*` not `region.*`).

Variable value not interpolated in queries

Symptoms:

- Panel shows the literal string `$varname` instead of the variable value.
- Labels or data don't change when the variable selection changes.

Solutions:

1. Verify you're using the correct variable syntax: `$varname` or `${varname}`.
2. TestData interpolates variables in these fields only: **Labels**, **Alias**, **Scenario**, **String Input**, **CSV Content**, and **Raw Frame Content**. Other fields don't support variable substitution.

Alerting errors

These errors occur when using TestData with Grafana Alerting.

Alert rule shows "no data" unexpectedly

Symptoms:

- Alert rule status shows **No Data** even though the scenario works in a panel.
- Alert never transitions to firing or normal.

Solutions:

1. Verify you're using a backend-evaluated scenario. Browser-only scenarios (Streaming Client, Grafana Live, Grafana API, Steps, Annotations, Node Graph, Flame Graph, Trace, and Raw Frames) return empty results when evaluated by the alerting engine. Use a backend scenario like Random Walk or Predictable Pulse instead.
2. Confirm the scenario is returning time-series data. Scenarios that produce logs (Logs), traces (Trace), graphs (Node Graph, Flame Graph), or tables (Table Static) can't be used as alert conditions.
3. Check that the **String Input** field isn't empty for scenarios that require it (for example, CSV Metric Values).

Alert rule never fires

Symptoms:

- Alert rule stays in **Normal** state.
- Data is visible in the panel but the alert condition is never met.

Solutions:

1. If using Random Walk, the values are non-deterministic and may not exceed your threshold. Use Predictable Pulse or CSV Metric Values with values that reliably cross the threshold.
2. Verify the **Reduce** expression is aggregating correctly. For example, **Last** returns the most recent value, while **Mean** averages the series.
3. Check the **Threshold** expression. Click **Preview** in the alert rule editor to see the evaluated value.
4. Ensure the evaluation interval is long enough for the scenario to produce data points. A Predictable Pulse with Step=60 needs at least a 60-second evaluation window.

Template variables don't resolve in alert rules

Symptoms:

- Alert rule error mentions unresolved variables.
- Alert query uses `$varname` syntax but the variable isn't replaced.

Solutions:

Grafana evaluates alert rules on the backend without dashboard context. Template variables like `$varname` aren't resolved during alert evaluation. Replace variable references with fixed values in the scenario options.

For more information, refer to the alerting limitations.

Alert evaluation timeout

Symptoms:

- Alert rule transitions to **Error** state.
- Logs show a timeout during query evaluation.

Solutions:

1. If you're using the Slow Query scenario, reduce the delay in the **String Input** field to a value shorter than the evaluation interval.
2. For other scenarios, verify that the query isn't producing an excessive number of data points. Reduce the time range or increase the step interval.

Enable debug logging

To capture detailed error information for troubleshooting:

1. Set the Grafana log level to `debug` in the configuration file:

   ```
   [log]
   level = debug
   ```

2. Review logs in `/var/log/grafana/grafana.log` (or your configured log location).
3. Look for entries containing `testdata` or `testdatasource` for scenario-specific details.
4. Reset the log level to `info` after troubleshooting to avoid excessive log volume.

Get additional help

If you've tried the solutions in this document and still encounter issues:

1. Check the Grafana community forums for similar issues.
2. Review Grafana GitHub issues for known bugs.
3. When reporting issues, include:
   - Grafana version
   - Selected scenario and its configuration
   - Error messages (redact sensitive information)
   - Steps to reproduce

□□□ #2
□ □□□ □□□ 2026-10-31 16:05:02 UTC
□ □□□ □□□ 2026-10-31 16:05:04 UTC