Pyroscope template variables

- Query variable types
- Create a query variable
- Query variable examples
- Use variables in queries
- Where you can use template variables
- Filters variable
- Related resources

Template variables let you create dynamic, reusable dashboards by replacing hard-coded values, such as service names or profile types, with selectable variables. Grafana displays these variables as drop-down menus at the top of the dashboard, letting viewers change the displayed data without editing queries.

For an introduction to templating and template variables, refer to Templating and Add and manage variables.

Query variable types

Query variables query the Pyroscope data source to populate drop-down values. When you create a query variable, select a Pyroscope data source and choose a query type.

Query type	Required inputs	Description
Profile type	None	Returns the profile types available from the data source. The drop-down displays a readable label, while the variable value is the full profile type ID, such as `process_cpu:cpu:nanoseconds:cpu:nanoseconds`.

Query type	Required inputs	Description
Label	Profile type	Returns the label names available for the selected profile type, such as `service_name`, `namespace`, and `region`.
Label value	Profile type, label	Returns the values for the selected label, scoped to the selected profile type.

Because the **Label** and **Label value** query types require a profile type, and **Label value** also requires a label, you can chain these variables so that each one depends on the previous selection.

Create a query variable

To create a query variable:

1. Navigate to **Dashboard settings** > **Variables**.
2. Select **Add variable**.
3. Set **Select variable type** to **Query**.
4. Select the Pyroscope data source.
5. Select a **Query type** of **Profile type**, **Label**, or **Label value**.
6. For a **Label** or **Label value** query, select a **Profile type**.
7. For a **Label value** query, select a **Label**.
8. Select **Apply** to save the variable.

Query variable examples

The following examples show common ways to use Pyroscope query variables.

Select a profile type from a drop-down:

Create a variable named `profile_type` to let viewers switch the profile type without editing the query:

1. Set **Query type** to **Profile type**.
2. Save the variable. The drop-down displays readable labels, while each value is the full profile type ID, such as `process_cpu:cpu:nanoseconds:cpu:nanoseconds`.

You can then reference `$profile_type` in the query editor's profile type field.

Populate a drop-down with services:

Create a variable named `service` that lists the services sending profiles:

1. Set **Query type** to **Label value**.
2. Set **Profile type** to a profile type, such as `process_cpu:cpu:nanoseconds:cpu:nanoseconds`.
3. Set **Label** to `service_name`.
4. Save the variable. The drop-down now lists the values of the `service_name` label.

Chain dependent variables:

Combine variables so each one narrows the next. This example builds a profile type selector, a label selector, and a label value selector that all update together:

1. Create a `profile_type` variable with **Query type** set to **Profile type**.
2. Create a `label` variable with **Query type** set to **Label** and **Profile type** set to `$profile_type`.
3. Create a `label_value` variable with **Query type** set to **Label value**, **Profile type** set to `$profile_type`, and **Label** set to `$label`.

When a viewer changes `profile_type`, the `label` drop-down refreshes with the labels for that profile type, and `label_value` refreshes with the values for the selected label.

Use variables in queries

After you create a variable, reference it in the query editor using the `$variable` or `${variable}` syntax. You can reference variables in both the profile type field and the label selector.

For example, if you have a variable named `service` created with the **Label value** query type on the `service_name` label, reference it in the label selector:

```
{service_name="$service"}
```

To let viewers switch the profile type, enter `$profile_type` in the query editor's profile type field.

When **Multi-value** or **Include All** is enabled for the variable, the value becomes a regular expression pattern, such as `value1|value2`. Use the `=~` operator instead of `=` in your label selector:

```
{service_name=~"$service"}
```

Where you can use template variables

The Pyroscope data source interpolates template variables in only some parts of the query editor. The following table lists which fields support variables.

Query editor field	Supports template variables
Profile type	Yes
Label selector	Yes
Group by	No
Span ID	No
Max Nodes	No
Limit	No

Query variable definitions also support variables. You can reference one variable in another to chain **Label** and **Label value** queries, as shown in Chain dependent variables.

{{< admonition type="note" >}} Fields other than the profile type and label selector don't interpolate template variables. If you enter a variable such as `$group` in the **Group by**, **Span ID**, **Max Nodes**, or **Limit** fields, Grafana passes the literal text to the backend instead of the variable's value. {{< /admonition >}}

Filters variable

The Pyroscope data source supports the Filters variable type, which lets dashboard viewers add label filters without editing queries.

To set up a Filters variable:

1. Navigate to **Dashboard settings** > **Variables**.
2. Select **Add variable**.
3. Set **Select variable type** to **Filters**.
4. Select the Pyroscope data source.
5. Select **Apply** to save the variable.

After you add the variable, a filter bar appears at the top of the dashboard. Viewers can add filters by selecting a label, an operator (`=`, `!=`, `=~`, `!~`), and a value. Grafana applies these filters to all

Pyroscope queries on the dashboard.

Related resources

- Query profile data: Use variables in the label selector.
- Troubleshoot the Pyroscope data source: Resolve variable-related query issues.

□□□ #2
□ □□□ □□□ 2026-10-31 16:02:33 UTC
□ □□□ □□□ 2026-10-31 16:02:35 UTC