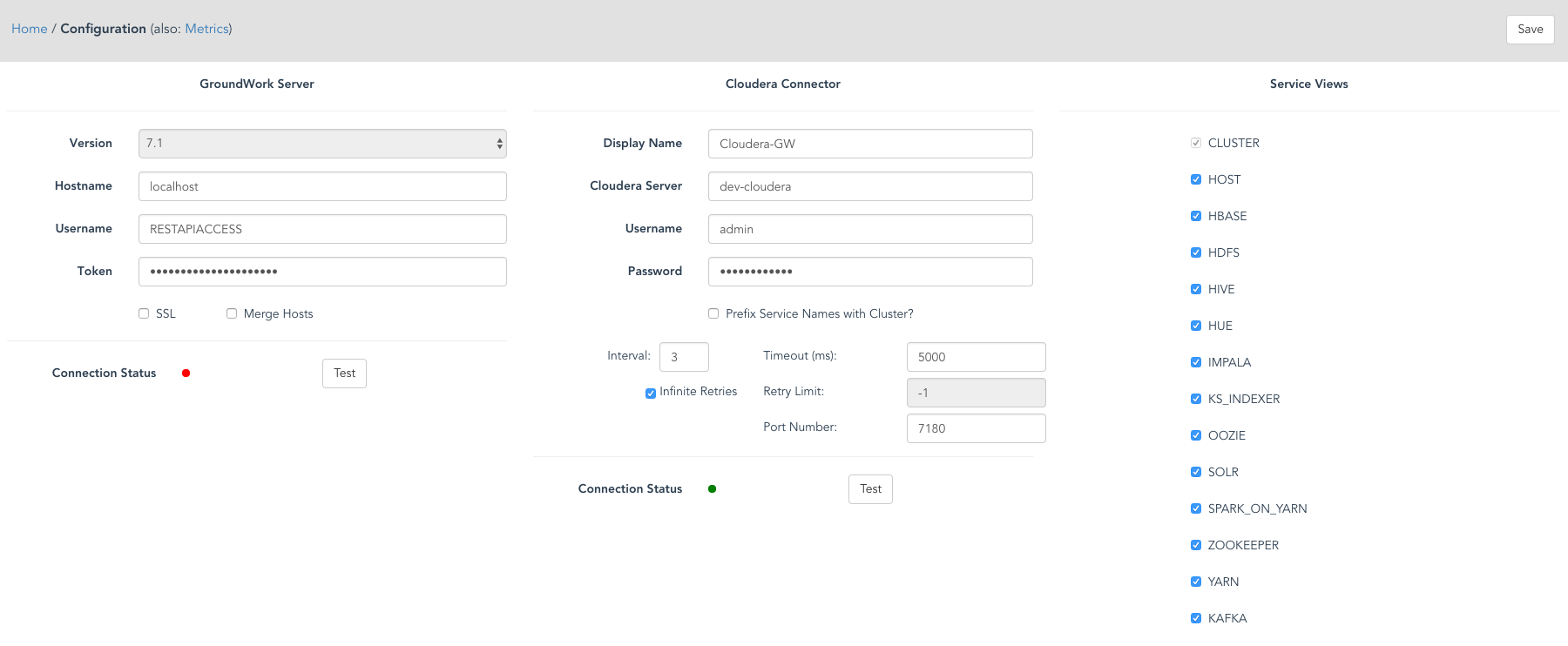
Cloudera Connector

# Configuring the Cloudera Connection

The Cloudera Configuration page is where you enter server connection parameters such as hostnames and credentials. Here you configure two servers:

1. The Groundwork Server
2. The Cloudera Server



## Groundwork Server Parameters

The Groundwork server is where CloudHub will store Cloudera metrics. Often, this is the same server as where CloudHub is running. However, CloudHub can also be run in a distributed environment, on its own node in a Groundwork cluster.

### Version

The Groundwork server version number. Usually you cannot change this value, and it will default to latest release installed. CloudHub can be configured to talk to versions going back to 7.0. CloudHub auto-detects which versions are available.

### Hostname

The host name or IP address where a Groundwork server is running. A port number should not be entered here. If Groundwork is running on the same server, you can enter localhost.

### Username

The provisioned Username granted API access on the Groundwork server.

### Token

The corresponding API Token (password) for the given Username on the Groundwork server.

### SSL

Check the SSL checkbox if your Groundwork server is provisioned with a secure HTTPS transport.

### Merge Hosts

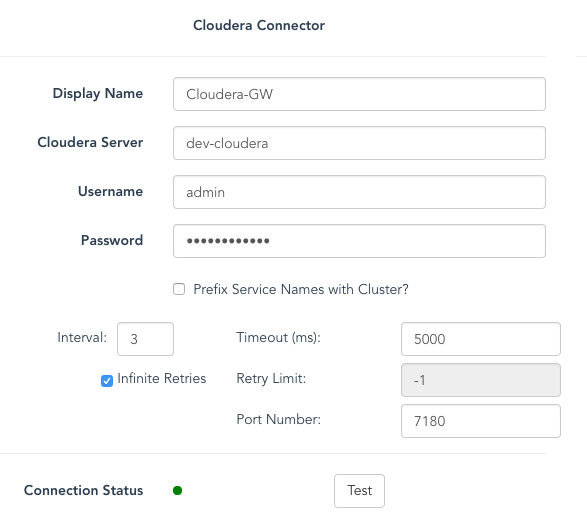
 If checked, this option combines all metrics of same named hosts under one host. For example, if there is a Nagios configured host named *demo1* and a Cloud Hub discovered host named *demo1*, the services for both configured and discovered hosts will be combined under the hostname *demo1* (case-sensitive).

### Connection Status and Test Button

After entering the Groundwork server parameters, press the Test button to test the connection. A dialog will be displayed with either a success message or, if the server cannot be contacted, an error message will be displayed with information describing why the connection failed. When a successful connection is made, the Connection Status button will change to green.

## Cloudera Connection Parameters

Here we enter values for the remote Cloudera server.



### Display Name

This is the configuration’s name displayed in the list of CloudHub connectors on the CloudHub home page.

### Hostname

The host name or IP address where a Cloudera server is running. A port number should not be entered here

### Username

The provisioned Username granted API access on the Cloudera server.

### Password

The corresponding Password for the given Username on the Cloudera server.

### Prefix Service Names with Cluster

This option is useful when are running two or more Cloudera clusters. Without this option checked, Cloudera services are stored as hosts with a host name directly corresponding to the Cloudera service name. For example, the HDFS service is stored in Groundwork as a host named “hdfs”. If this checkbox is checked, the hostname will be prefixed by the name of the cluster it is running under. For example, given a cluster named “cluster1”, the hostname for HDFS will be stored as “cluster1-hdfs”. Similarly, the SOLR service will be stored as a hostname “cluster1-solr”. The default setting is to disable this feature. If you find Cloudera services are not being mapped to unique Groundwork hostnames, you can use this feature even with a single-cluster Cloudera deployment

### Interval

This is the metric gathering interval for collecting monitoring data from Cloudera and sending it to the GroundWork server. The value is in minutes.

### Infinite Retries

Check this box if you want CloudHub to infinitely retry connection to Cloudera when the connection fails. When this box is checked, the Retry Limit field is disabled. When this box is unchecked, the Retry Limit field is enabled.

### Retry Limit

This entry is the number of retries for the connection and sets a limit on how many attempts are made after a failure. The number set indicates how many connections are attempted before the connection is left in an inactive state. At this point, the connection is suspended and you will need to manually restart it. When a retry limit is exhausted, all hosts managed by this connection are set to the monitor status “Unreachable” and all services for the matched hosts are set to the status of “Unknown”.

### Timeout

The connection timeout in milliseconds. Normally the default value is sufficient. when you have a slow network connection, you may want to increase the default value.

### Port

The optional port number for the Cloudera server API

## Cloudera Service Views

Optional features of Cloudera are called Cloudera Services. These features are the core components, or services that are managed by Cloudera. Services include:

* HBase
* HDFS
* Hive
* Hue
* Impala
* KSIndexer
* Oozie
* Solr
* Spark
* Zookeeper
* Yarn

Each of these services has their own rich set of metrics. By default, all services are selected. If you do not want to monitor one or more of these services, uncheck the corresponding check box and press Save.

Cloudera also provides Cluster and Host metrics. These metrics can also be optionally collected. If there are one or more clusters or hosts in the system, they will be automatically detected and collected.

Note that if you were collecting metrics for a service, and then unchecked that Cloudera service, the existing hosts and metrics stored in the Groundwork server will be deleted.

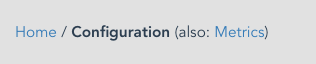
### Connection Status and Test Button

After entering the Cloudera server parameters, press the Test button to test the connection. A dialog will be displayed with either a success message or, if the server cannot be contacted, an error message will be displayed with information describing why the connection failed.

When a successful connection is made, the Connection Status button will change to green.

## Navigations

From the Configuration page, navigations are on displayed in the top navigation bar:



From here, you can navigate to:

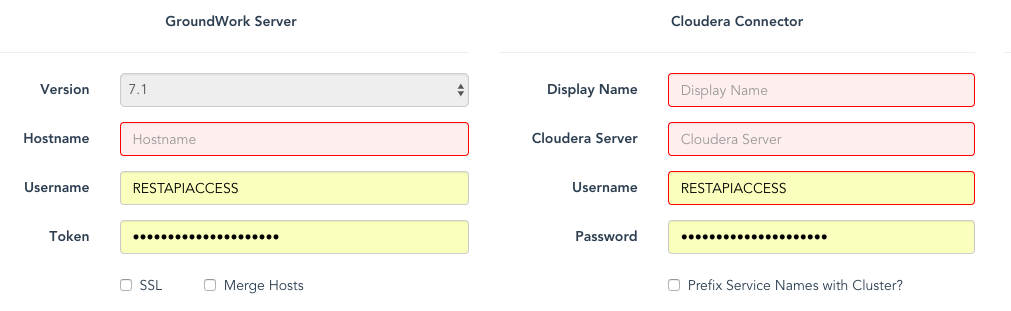
* Home - the CloudHub home page
* Metrics - the Metrics configuration page associated with this CloudHub connection.

When creating a new Cloudera configuration, the Metrics link is not visible until you successfully save the configuration parameters:

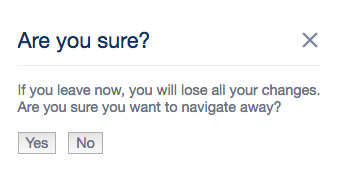
/Users/dtaylor/Desktop/temp/Screen Shot 2017-07-06 at 4.43.50 PM.png

Also, the Save button is not enabled until all required fields are validated. Here is a new configuration, where you will need to minimally enter the fields displayed in red:

* Groundwork Server hostname
* Cloudera Display name
* Cloudera Server



Note that configuration changes are not saved until you press the Save button in the top navigation. if you make changes on the configuration page, and forget to press Save, you will be prompted:



Once you are satisfied with your configuration settings, Press Save, then press the Metrics link in the navigation bar to start customizing your metrics for this connection.

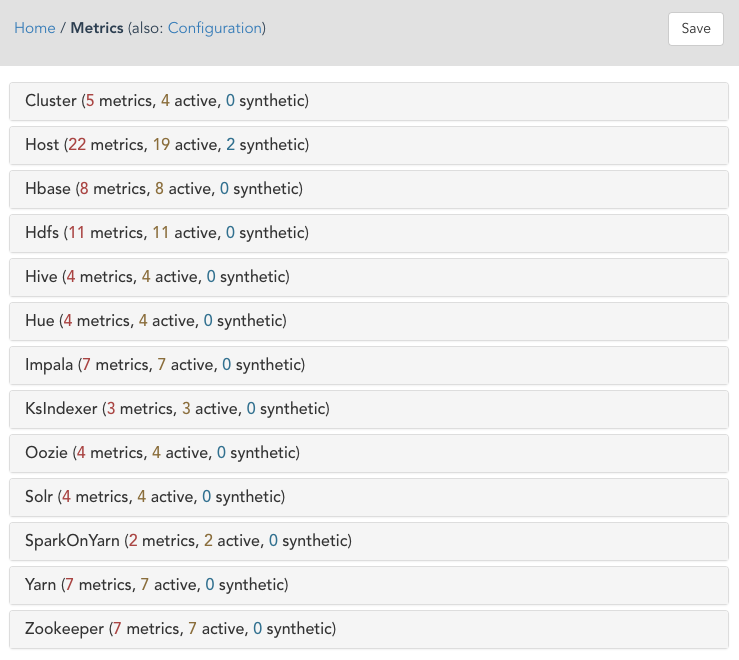
# Configuring Cloudera Metrics

The Cloudera Metrics page is where you customize the lists of Metrics being gathered for a connection. Out of the box, a complete list of metrics is provided for clusters, hosts, and Cloudera services. You can customize these metric lists by adding metrics to the list, deleting metrics, as well as creating calculated metric fields called Synthetic metrics.

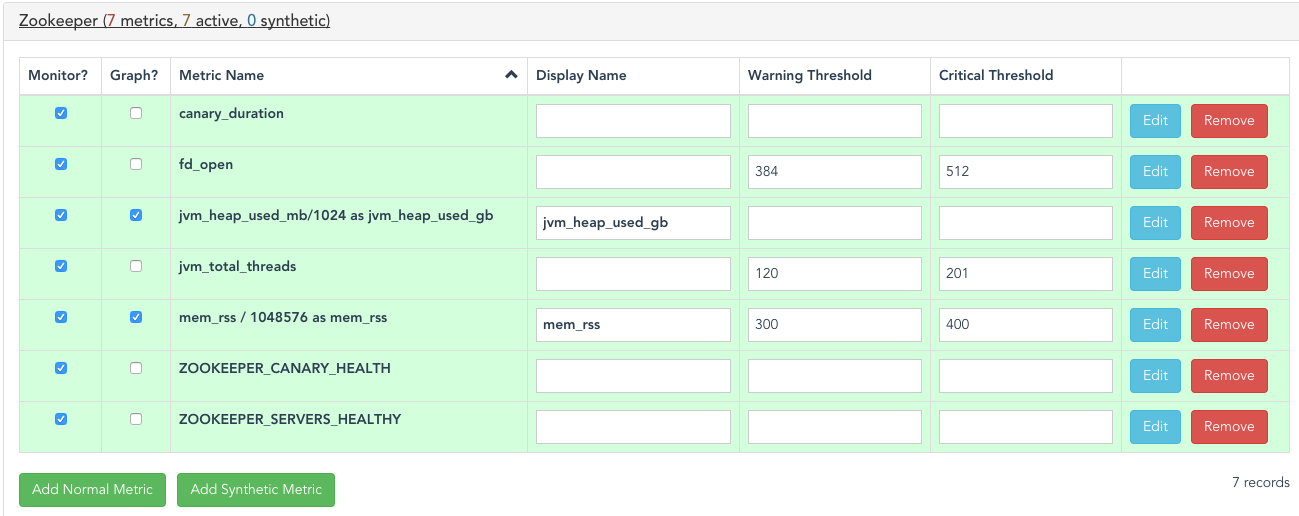
The Metrics page is displayed in groups of metrics grouped by Cluster, Host, and Cloudera Service collections. The counts of Metrics are displayed in the Group bar, summarized by

* Total metrics per group
* Active metrics per group
* Synthetic metrics per group

A metric is considered inactive if it is not monitored. (see section on Synthetics below)



You can configure the metrics for any group by clicking on the group bar. For example, if we click on the bottom Zookeeper group bar, the display automatically expands to show all metrics for the Zookeeper Cloudera service:



Each row in the grid represents a metric. Metrics can be added, edited or deleted. You can directly edit metrics in the grid or use the advanced metric dialog by pressing the Add or Edit buttons and then configuring all properties of a metric in the dialog. When editing metrics in the grid directly, you will need to press the Save button in the top navigation to commit your changes to CloudHub. The UI will know if you made changes and remind you to save your changes if you forgot.

## Grid Fields

The grid displays the following fields:

* Monitor? – check this if you want to enable monitoring of this metric
* Graph? – check this if you want to graph the values of this metric in time series
* Metric Name – the exact Cloudera metric name or a Cloudera metric expression. This field is read-only. Press the Edit button to modify it.
* Display Name – overrides the metric name and stores the metric in Groundwork as a service with this name
* Warning Threshold – metric value that will trigger a Groundwork Warning alert
* Critical Threshold – metric value that will trigger a Groundwork Critical alert

Leaving the threshold fields blank will disable threshold triggers.

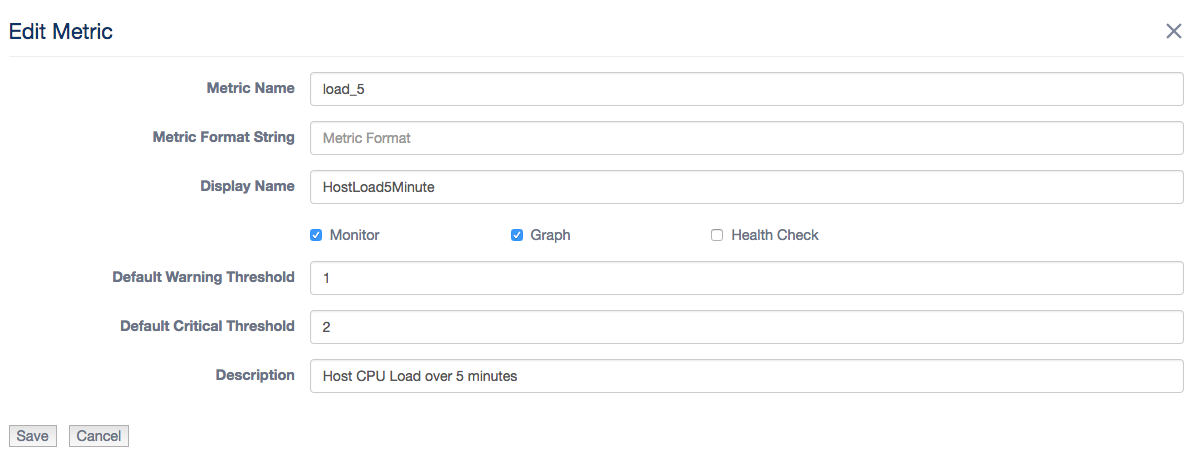
Metrics come in two flavors: they can either be Normal or Synthetic metrics.

## Normal Metrics

Normal metrics can be:

* Single Metric Names
* Computed Metric Names
* Health Checks metrics
* Configuration metrics (not monitored)

### Single Metric Name with Display Name



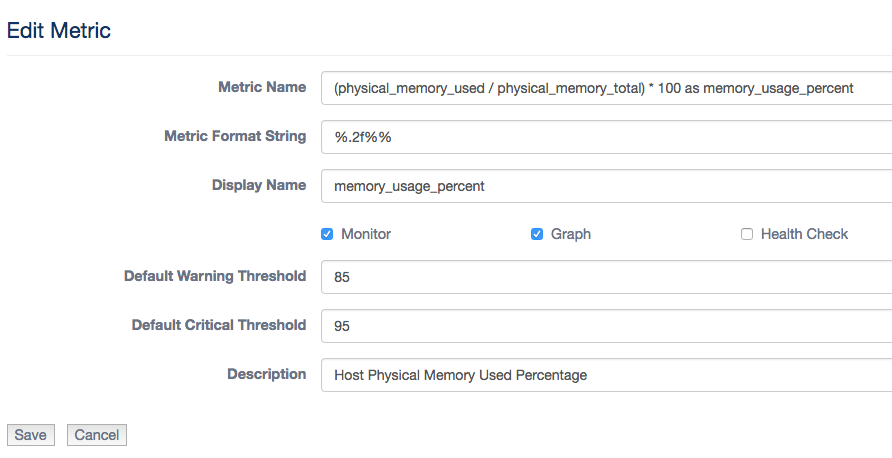
### In this example, we have a Host metric named load\_5. This is the unique name of the metric in Cloudera. In the Display field, we renamed this metric to HostLoad5Minute. Renaming metrics is an optional feature. In this case, we renamed load\_5 to have a more descriptive metric name displayed in the Groundwork Status Viewer. We recommend filling out the description field to describe the metric. This metric represents the Host CPU Load averaged over 5 minutes. We have also setup warning and critical thresholds. Note that the metric will be monitored and graphed.

Note that we never use dashes in metric names, only underscores. This is because dashes are not valid variable names in a Cloudera or synthetic expression.

As you type into the Metric Name field, the valid names of metrics available are automatically auto-suggested. This ensures that you use a valid Cloudera metric.

### Computed Normal Metrics

Normal metrics can also be computed. They differ from Synthetic metrics in that the value of the metric is an expression, and it is computed on the Cloudera server, not by CloudHub.



In this example, the Metric name is a computed Cloudera expression. The expression includes two Cloudera metrics: physical\_memory\_used and physical\_memory\_total. The expression takes the memory used metric, divides it by the total memory metric and multiplies that by 100 to return a computed metric named memory\_usage\_percent. The AS keyword is required. It defines an alias for the expression to uniquely name the metric:

|  |
| --- |
| (physical\_memory\_used / physical\_memory\_total) \* 100 as memory\_usage\_percent |

IMPORTANT: When working with computed metrics, make sure to include the AS clause (alias) in your computed expression. Aliases are required on computed metrics. Additionally, the metric Display name must match the alias.

The Metric Format String is an optional C-style formatting string. Here we limit the floating point number to 2 decimal places, and then append a percent sign to the computed metric value:

|  |
| --- |
| %.2f%% |

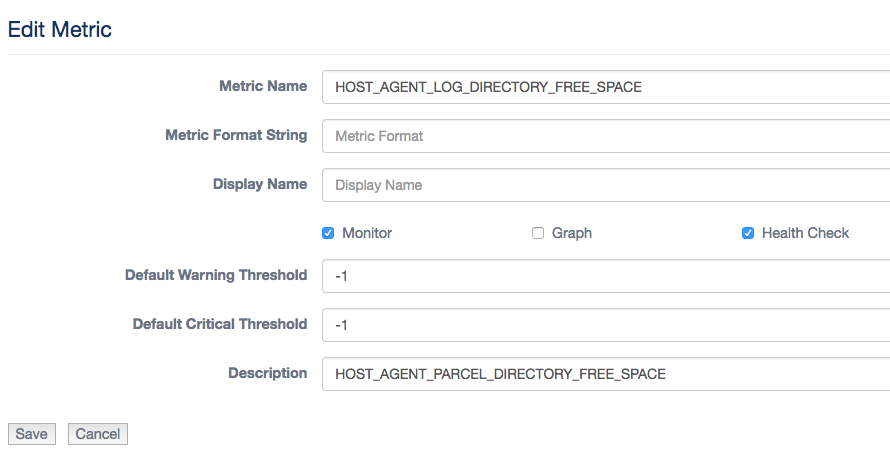
See the section below on Example Formatting for more examples.

As you type into the Metric Name field, the valid names of metrics available are automatically auto-suggested. This ensures that you use a valid Cloudera metric name in your expression.

### Health Check Metrics

Health Check metrics are a special type of metric that only report back Health Check status.

These metrics do not have numeric values, but instead have health check statuses that map to Groundwork statuses.



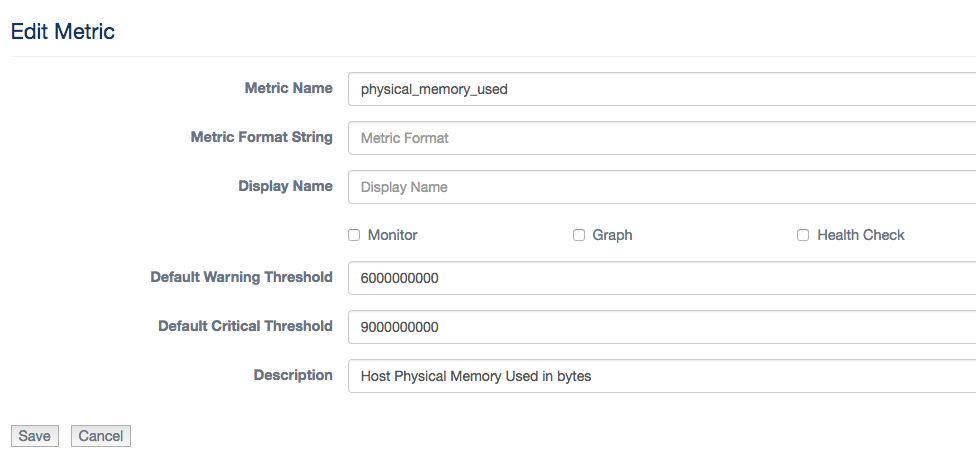
Health Check metrics are flagged with the Health Check checkbox. As you type into the Metric Name field, the valid names of Health Check metrics available are automatically auto-suggested. This ensures that you use a valid Cloudera Health Check metric.

See the section below on Health Check Status Mappings for the complete list of health check status mappings.

### Configuration Metrics

Configuration metrics are only used in synthetic computations. They are not reported back to the Groundwork server. To create a configuration metric, simply do not check the Monitor checkbox.

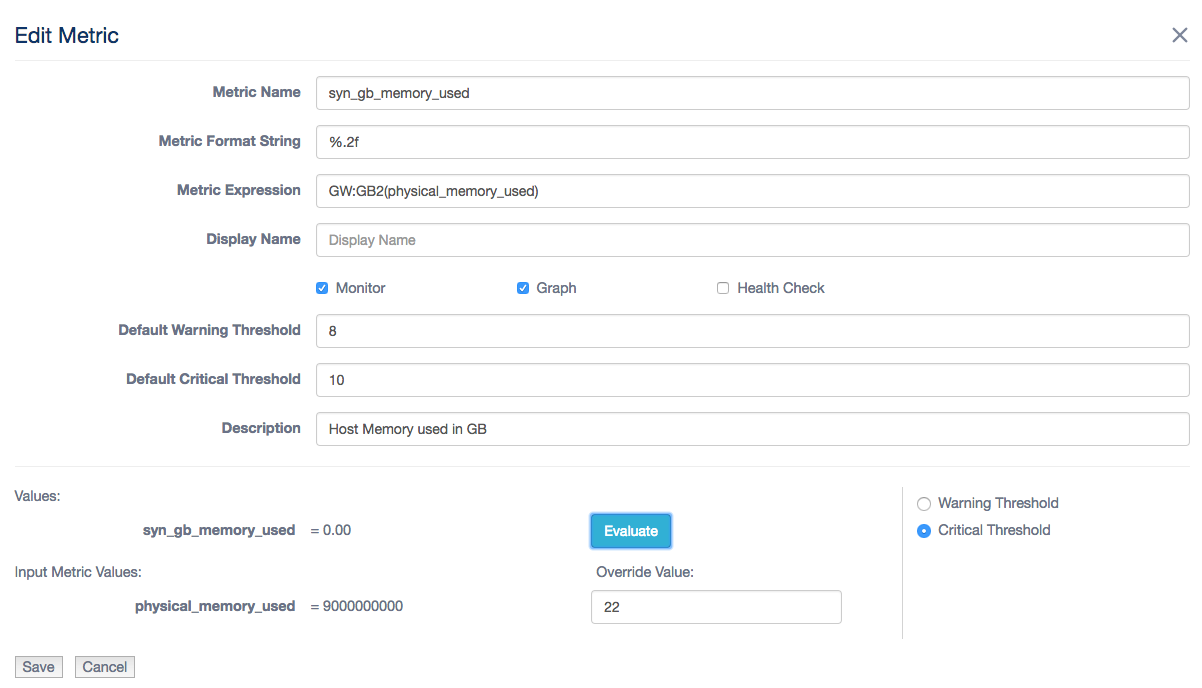
Configuration metrics are used in synthetic calculations, where the value is required, for example to perform a to megabyte or to gigabyte conversion, but you do not want to report back the byte value to the Groundwork server.



The Monitor check box is left unchecked. Note that we still use thresholds, as they are useful in the Synthetic Expression evaluator.

# Synthetic Metrics

A Synthetic metric is a metric that is computed by CloudHub. It has one additional field, expression, that normal metrics do not have.



The synthetic metric name is a simple metric name conforming to the Groundwork service name requirements. No spaces are allowed. By convention, we name synthetic metrics with the prefix “syn\_”.

The Metric expression field contains the synthetic expression. In this example, we use a Groundwork function, GW:GB2 to convert the value of the physical\_memory\_used Cloudera metric, a value in bytes, to a gigabyte value:

|  |
| --- |
| GW:GB2(physical\_memory\_used) |

The Metric Format String is an optional C-style formatting string. Here we limit the floating-point number to 2 decimal places:

|  |
| --- |
| %.2f |

See the section below on Example Formatting for more examples.

As you type into the Metric Expression field, the valid names of metrics available are automatically auto-suggested. This ensures that you use a valid Cloudera metric name in your expression. Synthetic expressions are limited to the Normal metrics defined for the current group. Additionally, the auto-suggest feature displays all Groundwork functions.

## The Synthetic Expression

This field contains an actual programmable expression that is parsed by CloudHub. The expression is made up of:

* Normal Metrics (not health checks)
* Expression Operations (addition, subtraction, multiplication, division, parenthesis for grouping)
* Groundwork Functions
* Math Functions

Example expression with division and multiplier operators, parenthesis for grouping, and data conversion of integers to double values. The two normal Host metrics are fd\_open and fd\_max. Note that both normal metrics must be defined for this group. Other synthetic metrics cannot be included in a synthetic expression.

|  |
| --- |
| (GW:toDouble(fd\_open) / GW:toDouble(fd\_max)) \* 100.0 |

The data types of Cloudera Metrics are typically floating point (double) values for any measurements. For counters, like the example above, are usually integers or longs. Consult the CloudHub documentation for a complete reference guide to metrics.

<https://www.cloudera.com/documentation/enterprise/5-5-x/topics/cm_metrics.html>

Type conversion is supported as Groundwork functions for both floating point and integer numbers. See the section below Groundwork Functions for a complete list.

## The Expression Evaluator

The synthetics dialog has an expression evaluator to try out and test your expressions before saving them. The evaluator is displayed at the bottom of the dialog. Each variable in the expression is evaluated based on the check boxes

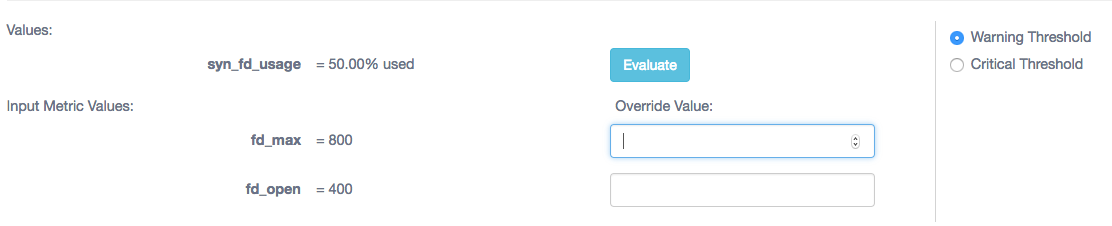
Given the expression:

(GW:toDouble(fd\_open) / GW:toDouble(fd\_max)) \* 100.0

there are two variables, fd\_open and fd\_max. These variables are displayed in the Input Metric Values section of the dialog. There are three ways to evaluate the expression based on:

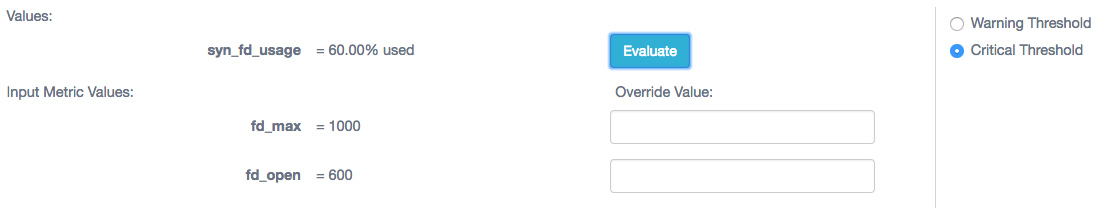
* Warning Threshold – click the Warning Threshold option then press evaluate
* Critical Threshold – click the Critical Threshold option then press evaluate
* Override – enter values into the Override Value fields, then press evaluate

#### Warning Threshold



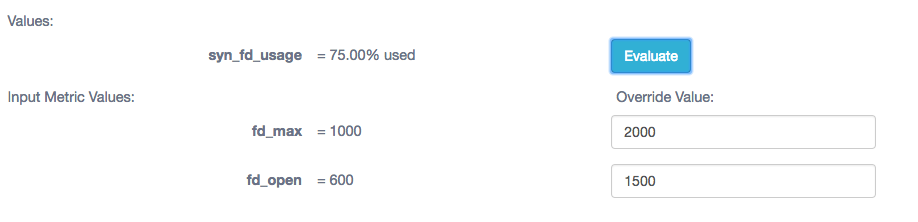
The fd\_max and fd\_open metric fields are predefined with warning threshold values of 800 and 400. Pressing evaluate yields the formatted output: 50.00% used

#### Critical Threshold



The fd\_max and fd\_open metric fields are predefined with critical threshold values of 1000 and 600. Pressing evaluate yields the formatted output: 60.00% used

#### Override Values



The fd\_max and fd\_open metric fields are entered with values of 2000 and 1500. Pressing evaluate yields the formatted output: 75.00% used

# Cloudera Computed Examples

Cloudera computed (normal) metrics are calculated in the Cloudera server. Here are some examples of Cloudera computed metrics…

Example 1: Cloudera computes the memory usage of physical memory of a host

|  |
| --- |
| Computed Host Metric:  (physical\_memory\_used / physical\_memory\_total) \* 100 as memory\_usage\_percent  Format:  %.2f%%  Display Name:  memory\_usage\_percentage  Warning Threshold: 85  Critical Threshold: 95  Description:  Host Physical Memory Used Percentage |

Example 2: Cloudera converts bytes to MB for a host metric

|  |
| --- |
| Computed Host Metric:  physical\_memory\_used / 1048576 as memory\_used\_mb  Display Name:  memory\_used\_mb  Warning Threshold: 8182  Critical Threshold: 10240  Description:  Host Physical Memory Used in Megabytes |

Example 3: Cloudera calculates Host CPU Load Percentage over 1 minute

|  |
| --- |
| Computed Host Metric:  cpu\_user\_rate / getHostFact(numCores, 1) \* 100 as cpu\_rate\_user  Display Name:  cpu\_rate\_user  Warning Threshold: 75  Critical Threshold: 90  Description:  Host CPU Load Percentage over 1 Minute |

Note that Cloudera currently has the following functions that can be used in a metric computation:

dt(metric) - Derivative with negative values.

The change of the underlying metric expression, per second.

Example:

|  |
| --- |
| dt(jvm\_gc\_count) |

dt0(metric) **-** Derivative where negative values are skipped (useful for dealing with counter resets). The change of the underlying metric expression, per second.

Example:

|  |
| --- |
| dt0(jvm\_gc\_time\_ms) / 10 |

getHostFact(string factName, double defaultValue) - Retrieves a fact about a host.

Example:

|  |
| --- |
| dt(total\_cpu\_user) / getHostFact(numCores, 2) |

This example divides the results of dt(total\_cpu\_user) by the current number of cores for each host. If the number of cores cannot be determined, the default "2" will be used.

getHostFact currently supports one fact, numCores.

# Synthetics examples

Example 1: CloudHub computes the physical memory used from bytes to GB with GW function

|  |
| --- |
| Metric Name:  syn\_gb\_memory\_used  Expression:  GW:GB2(physical\_memory\_used)  Format:  %.2f% GB  Warning Threshold: 8  Critical Threshold: 10  Description:  Host Memory Used in GB |

Example 2: CloudHub computes the physical memory used from bytes to GB with GW functions to convert integer values to double values

|  |
| --- |
| Metric Name:  syn\_fd\_usage  Expression:  (GW:toDouble(fd\_open) / GW:toDouble(fd\_max)) \* 100.0  Format:  %.2f%%%  Warning Threshold: 700  Critical Threshold: 1000  Description:  Percentage of File Descriptors Used |

Example 3: CloudHub computes the percentage of memory used with the divideToPercentage function. Note this function returns an integer.

|  |
| --- |
| Metric Name:  syn\_physical\_mem\_percent  Expression:  GW:divideToPercentage(physical\_memory\_used,physical\_memory\_total)  Format:  %d %% used  Description:  Percentage of Host Memory Used |

# Groundwork Functions

### Byte Conversion Functions Using Strict Hexadecimal Values (1024..)

|  |  |
| --- | --- |
| GW:KB(bytes) | Convert bytes to kilobytes |
| GW:MB(bytes) | Convert bytes to megabytes |
| GW:GB(bytes) | Convert bytes to gigabytes |
| GW:TB(bytes) | Convert bytes to terabytes |

### Byte Conversion Functions Using Decimal Values (1000..)

|  |  |
| --- | --- |
| GW:KB2(bytes) | Convert bytes to kilobytes |
| GW:MB2(bytes) | Convert bytes to megabytes |
| GW:GB2(bytes) | Convert bytes to gigabytes |
| GW:TB2(bytes) | Convert bytes to terabytes |

### Byte Conversion Functions Using Decimal Values (1000..)

|  |  |
| --- | --- |
| GW:min(x,y) | Returns the minimum value of two numbers |
| GW:max(x,y) | Returns the maximum value of two numbers |

### Type Conversion

|  |  |
| --- | --- |
| GW:toDouble(m) | Converts a number to double precision |
| GW:toInteger | Converts a number to an integer |
| GW:toLong | Converts a number to a long integer |

#### GW:scalePercentageUsed

*This Function provides percentage usage synthetic values.  
Calculates the usage percentage for a given used metric and a corresponding available metric. Both the used metric and available metric can be scaled by corresponding scale factor parameters.  
  
Example:* scalePercentageUsed(summary.quickStats.overallMemoryUsage,summary.hardware.memorySize, 1.0, 1.0)

*Parameters***used** - Represents a 'used' metric value of how much of this resource has been used such as 'overallMemoryUsage'

**available** - Represents the totality of a resource, such as all memory available

**usedScaleFactor -** multiply usage parameter by this value, or pass in null to not scale. Passing in 1.0 will also not scale  
**availableScaleFactor** - multiply available parameter by this value, or pass in null to not scale. Passing in 1.0 will also not scale

*Returns the percentage usage as an integer*

#### GW:scalePercentageUnused

*This Function provides percentage unused/free synthetic values.  
Calculates the unused(free) percentage for a given unused metric and a corresponding available metric.  
Both the unused metric and available metric can be scaled by corresponding scale factor parameters.  
  
 Example:* scalePercentageUnused(summary.freeSpace,summary.capacity, 1.0, null, true)

Parameters

**unused** - Represents a metric reference value of how much of this resource has not be used (free)  
**available** Represents the totality of a resource, such as all disk space available  
**usageScaleFactor** - multiply usage parameter by this value, or pass in null to not scale. Passing in 1.0 will also not scale  
**availableScaleFactor -** multiply available parameter by this value, or pass in null to not scale. Passing in 1.0 will also not scale

*Returns the percentage not used (free) as an integer*

#### GW:percentageUsed

*This Function provides percentage usage synthetic values.  
Calculates the usage percentage for a given used metric and a corresponding available metric.*

*Example:* scalePercentageUsed(summary.quickStats.overallMemoryUsage,summary.hardware.memorySize)

Parameters

**used** - Represents a 'used' metric value of how much of this resource has been used such as 'overallMemoryUsage'  
**available** - Represents the totality of a resource, such as all memory available

*Returns the percentage usage as an integer*

#### GW:percentageUnused

*This Function provides percentage unused/free synthetic values.  
Calculates the unused(free) percentage for a given unused metric and a corresponding available metric.  
Both the unused metric and available metric can be scaled by corresponding scale factor parameters.*

*Example:*

scalePercentageUnused(summary.freeSpace, summary.capacity)

*Parameters:*  
***unused*** *Represents a metric reference value of how much of this resource has not be used (free)****available*** *Represents the totality of a resource, such as all disk space available*

*Returns he percentage not used (free) as an integer*

#### GW:divideToPercentage

*Given two metrics, dividend and divisor divides them and returns a percentage ratio  
  
Example:* GW:divideToPercentage(summary.quickStats.overallMemoryUsage,summary.hardware.memorySize)

*Parameters*

***dividend*** *typically a usage or free type metric****divisor*** *typically a totality type metric, such as total disk space*

***Returns t****he percentage ratio as an integer*

#### GW:toPercentage

*Turns a number such as .87 into an integer percentage (87). Also handles rounding of percentages*

***Example***

GW:toPercentage(summary.quickStats.overallMemoryUsage)

***Parameters***

***value*** *the value to be rounded to a full integer percentage*

*Returns the percentage value as an integer*

# Math Functions

Functions from Java Math library sample:

* min(n1,n2), max(n1,n2)
* abs
* cos, sin, tanexp, log, sort
* ceil, floor, round
* rint
* pow

See docs

<https://docs.oracle.com/javase/8/docs/api/java/lang/Math.html>

Math functions should be prefixed by

Math:

Example

Math:abs(metric)

# Example Formatting

The formatting field uses standard C/Java style formatting strings. Typically, you will only be formatting one number, so the formatting strings should be very simple. Data types used are:

* Integer Numbers %d
* Floating Point Numbers %f

Example of formatting an integer value 2175:

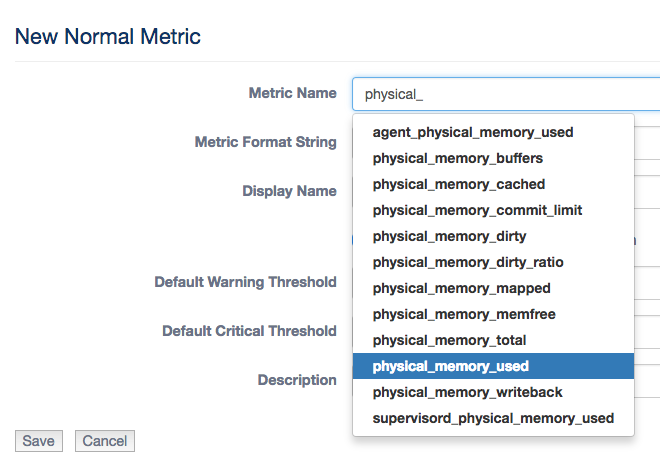
|  |  |
| --- | --- |
| **Format String** | **Output** |
| %d | 2175 |
| %05d | 02175 |
| %+5d | +2175 |
| %,d | 2,175 |
| %d%% percent | 2175% percent |

Example of formatting a floating point value 3.141593:

|  |  |
| --- | --- |
| **Format String** | **Output** |
| %f | 3.141593 |
| %.2f | 3.14 |
| %2.3f | 3.141 |
| %.2f%% | 3.14% |
| %.2f percent | 3.14 percent |

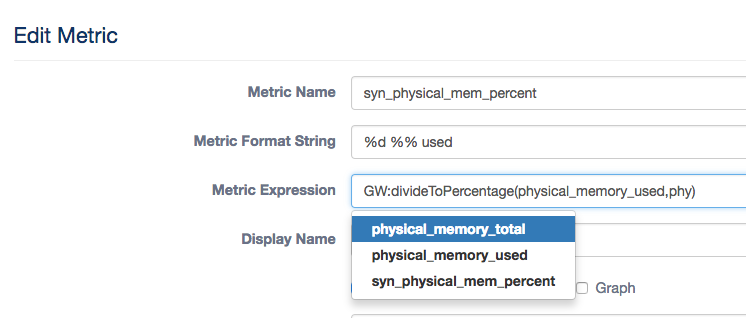
# Normal Metric Discovery

When entering a metric name in the Metric Name field, metrics are auto-discovered. As you type into the Metric name field, the names of metrics will be auto-suggested. Cloudera has thousands of metrics. The auto-discovery feature can be very useful in finding the right metric.

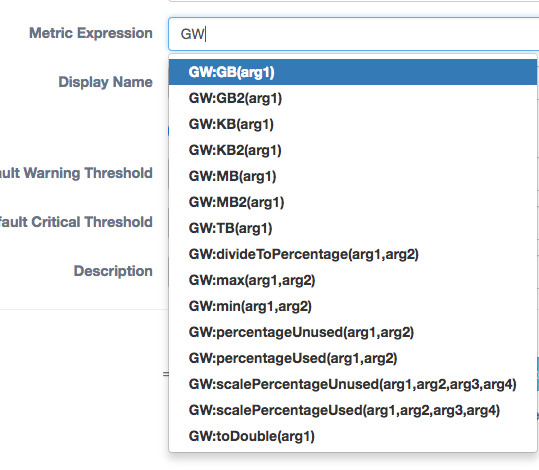


# Synthetic Metric Auto Suggest

When entering a synthetic expression, configured metrics will be auto-suggested. As you type into the Metric name field, the names of metrics will be auto-suggested.



Functions are also available in the auto-suggestion list:



# Health Check Status Mappings

Cloudera Health Check statuses are mapped to Groundwork monitor status values in the Status Viewer based on the tables below:

#### Cluster Status Mapping

|  |  |
| --- | --- |
| **Cloudera Cluster Status** | **Mapped to Groundwork Host Status** |
| UNKNOWN | UNREACHABLE |
| NONE | UNREACHABLE |
| STOPPED | SUSPENDED |
| DOWN | DOWN |
| UNKNOWN\_HEALTH | WARNING |
| DISABLED\_HEALTH | WARNING |
| CONCERNING\_HEALTH | WARNING |
| BAD\_HEALTH | WARNING |
| GOOD\_HEALTH | UP |
| STARTING | PENDING |
| STOPPING | DOWN |
| HISTORY\_NOT\_AVAILABLE | WARNING |

#### Host and Cloudera Service Status Mapping

|  |  |
| --- | --- |
| **Cloudera Host Status** | **Mapped to Groundwork Host Status** |
| HISTORY\_NOT\_AVAILABLE | UNREACHABLE |
| NOT\_AVAILABLE | UNREACHABLE |
| DISABLED | SUSPENDED |
| GOOD | UP |
| CONCERNING | WARNING |
| BAD | DOWN |

#### Metric Status Mapping

|  |  |
| --- | --- |
| **Cloudera Metric Status** | **Mapped to Groundwork Service Status** |
| HISTORY\_NOT\_AVAILABLE | UNKNOWN |
| NOT\_AVAILABLE | UNKNOWN |
| DISABLED | PENDING |
| GOOD | OK |
| CONCERNING | WARNING |
| BAD | CRITICAL |