Policy Designer

Contents

Copyright Digital, part of GE Vernova

Chapter

1

Overview

Topics:

Overview of the Policy Designer Module

Policy Designer is a visual business rules design tool that enables users with no programming knowledge to automate asset management workflows to reduce the operational cost and eliminate delays in identifying and responding to asset conditions that may impact reliability, availability, productivity, or operating cost.

Policy Designer allows you to define and deploy policies that apply calculations and logic to specific input records in APM, for example, assets or health indicators, and act on the results.

Policies may be executed on a schedule, on demand, or triggered by a change in an input data source.

Note: For information on policies that execute when any record in a APM Family is inserted, modified or deleted, see Family Policies.

Use Case Example

Traditionally, companies attempted to improve asset performance by identifying historical trends and using them to define future actions. For example, the results of a Reliability Distribution Analysis may indicate that a motor fails about every six months, so a strategy might be implemented to replace motors of that type every five months (that is, one month prior to the next predicted failure). However, some motors could last longer than six months (for example, for eight months), and in these cases, replacing the motor too early would result in unnecessary costs. In other cases, the motor might fail before the planned replacement, resulting in more expensive repairs and lost production.

An engineer might analyze historical reliability and performance data and determine that when the motor temperature reaches 200 degrees, it will begin to fail. Rather than implementing a strategy to replace the motor after five months, as suggested by the reliability analysis, he can create a policy to take action when the high temperature condition is met. If the policy retrieves a reading from a process historian indicating that the motor temperature has reached 200 degrees, policy logic can create a recommendation automatically for a technician to replace the motor and send an email message to notify the engineer. In this way, the company can save money by avoiding unnecessary replacements and still prevent failures that will occur if the motor is replaced too late.

Policies can be configured to monitor asset conditions continuously and perform a variety of actions automatically based on criteria defined in the policy. In addition to creating recommendations and sending email messages, a policy can create Event records to track the occurrence of certain conditions. You can also use Policy Designer to create and update composite health indicators that indicate the overall health of a piece of equipment.

The basic elements of a policy

A policy model consists of a network of connected nodes:

- Input nodes that retrieve information, for example, a APM Family record, a query result, or readings from a process historian.
- Condition, Logic and Calculation nodes that evaluate the information.
- Action nodes that automate asset management workflows, for example; send an email, create a recommendation or work request, or add a value to a health indicator.

Connections between nodes define the order in which nodes are executed. Connections from Condition and Logic nodes also define whether the successor node is executed depending on the result (true or false) of the preceding condition.

A policy is deployed by creating policy instances that define the specific records to which the policy will be applied. At least one policy instance is required for each policy.

Access the Policy Designer Overview Page

Procedure

In the **Applications** menu, navigate to the **TOOLS** section, and then select **Policy Designer**.

The **Policy Designer Overview** page appears.
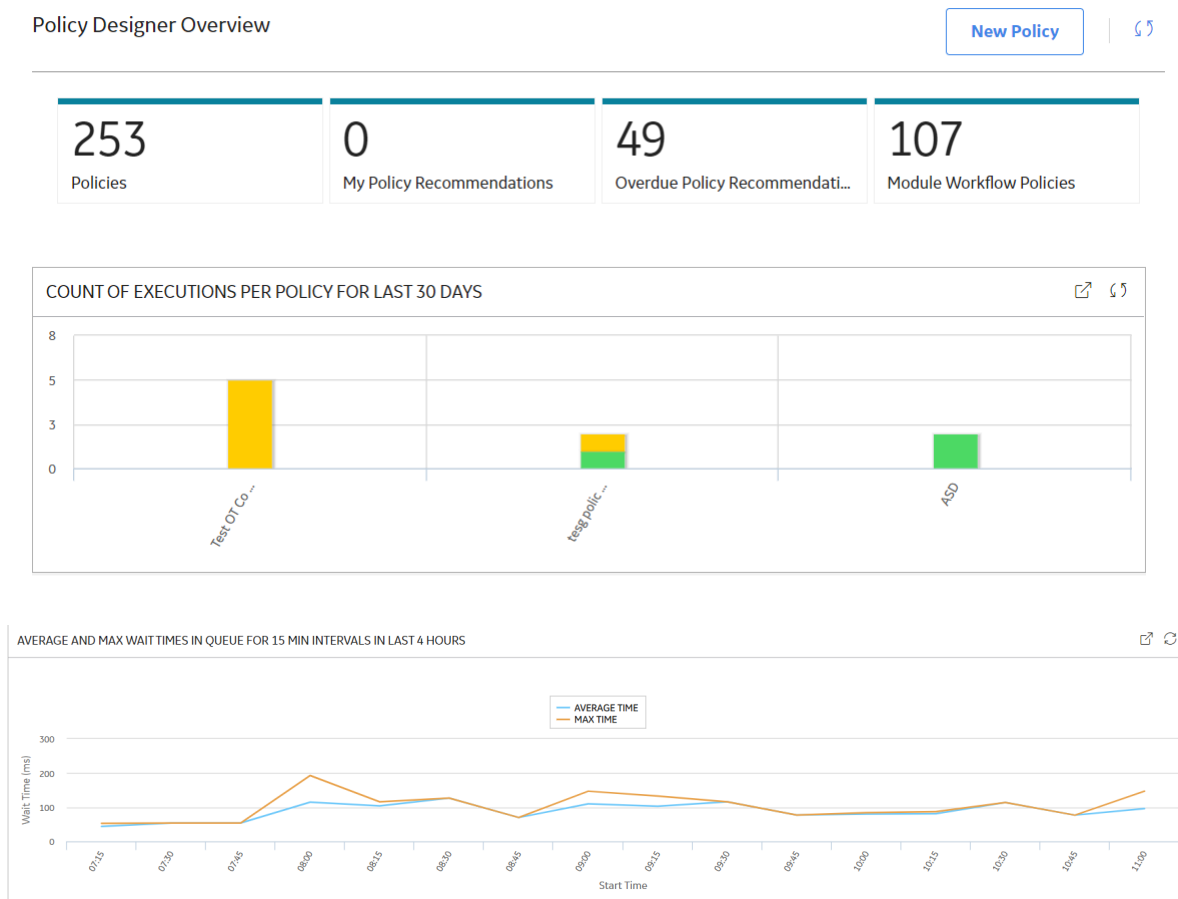
The **Policy Designer Overview** page contains the following sections that summarize the number of Policies and Policy Recommendations:

- **Policies**: Contains a list of all active and inactive Policies.
- **My Policy Recommendations**: Contains a list of the Policy Recommendation records assigned to you.
- **Overdue Policy Recommendations**: Contains a list of the Policy Recommendation records that are past target completion dates with pending recommended actions.
- **Module Workflow Policies**: Contains a list of the Policies that support APM module workflows.

    Note: The module workflow Policies do not appear in the other sections of this page.

Note: The **Policy Designer Overview** page is not updated automatically when you return to the previously opened tab. You can select ⟳ to update the page.

The **Count of Executions per Policy for Last 30 Days** section in the **Policy Designer Overview** page displays a graphical summary of the execution results for the 20 Policies that were most active in the past 30 days. You can select an execution result to access the corresponding Policy.

The average and max wait time graph displays a graphical summary of the time in queue, that is the time taken from the Trigger Sent time to the Execution Start time in the past four hours (4 hrs).

Policy Designer Workflow

This workflow provides the basic, high-level steps for using Policy Designer. The steps and links in this workflow do not necessarily reference every possible procedure.

1. Begin a policy by providing a name and description for the policy.
2. Build a policy model, which includes nodes and connections that represent the inputs, logic, and resulting actions for the policy.

   Note: Interaction with the model design canvas, such as adding and moving nodes, is not available on touch-screen devices.
3. Create policy instances, which define the specific records to which logic in the policy model will be applied.
4. Run the validation process to confirm that the policy logic is working correctly.
5. Define execution settings to specify whether you want the policy to be executed automatically or executed according to a predefined schedule (or both).
6. Define execution history settings to specify what types of policy execution records will be created.
7. Activate the policy and policy instances so that they will be executed according to your defined execution settings.

Chapter
2

Workflow

Topics:

-
-

Design Policy Workflow

Policy Designer: Design Policy Workflow

This workflow describes the process for designing and validating policies. Policies can be used to automate APM workflows by generating email alerts, as well as creating, deleting, or updating records used in the other workflows based on the input values and policy logic.

In the following workflow diagram, the blue text in a shape indicates that the corresponding description has been provided in the sections that follow the diagram. For more information, refer to the Interpreting the Workflow Diagrams topic in the APM Product Workflows documentation.

Note: For information on the personas associated with a APM module, refer to the APM Product Workflows documentation.
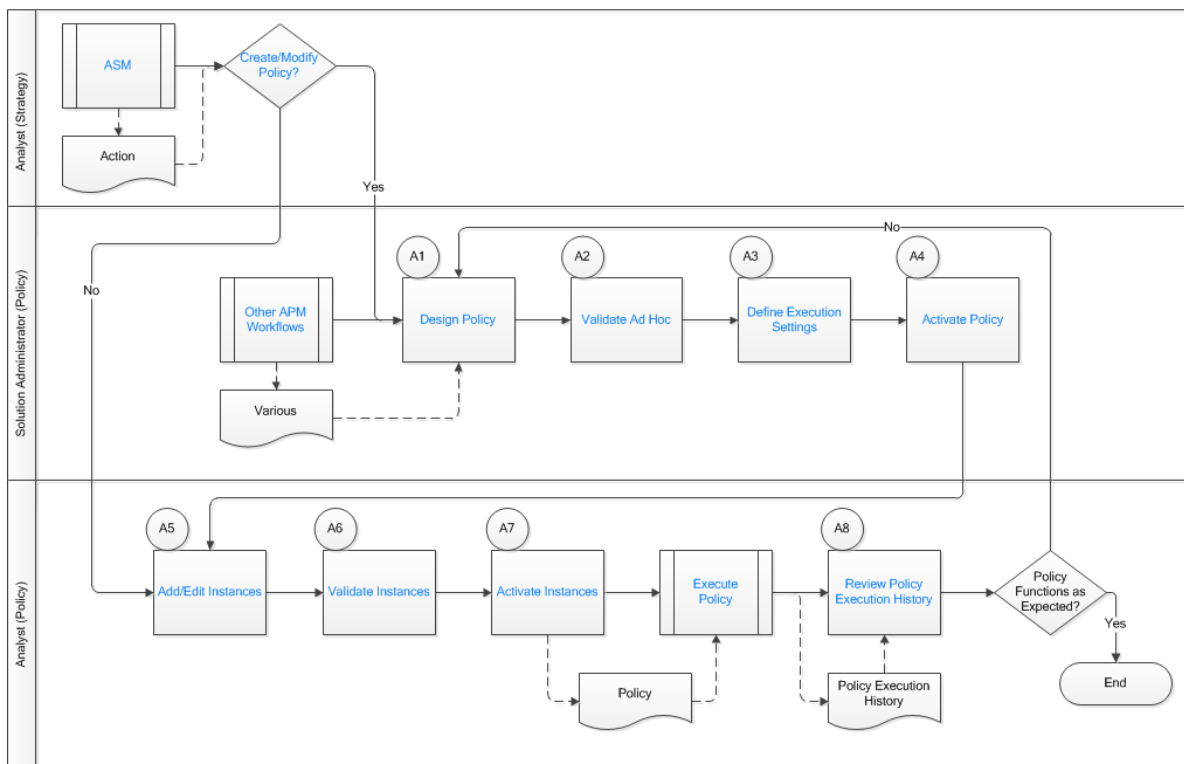
1. ASM (Asset Strategy Management) on page 7
2. Create / Modify Policy? on page 7
3. Design Policy on page 7
4. Validate Ad Hoc on page 7
5. Define Execution Settings on page 8
6. Activate Policy on page 8
7. Add / Edit Instances on page 8
8. Validate Instances on page 8
9. Activate Instances on page 8
10. Execute Policy on page 8
11. Review Policy Execution History on page 9

ASM (Asset Strategy Management)

Persona: Analyst (Strategy)

During implementation of an Asset Strategy, the Strategy Analyst may identify policies needed to address Strategy Actions. For example, policies can calculate health indicators, create event logs, and/or provide automated decision support. Actions may be implemented as policy instances or health indicators that are updated by a policy.

Create / Modify Policy?

Persona: Analyst (Strategy)

Determine whether you can modify instances associated with an existing policy to address strategy actions, or if you must create a new policy or modify a policy model or execution settings.

Other Workflows

Persona: Various

Policies may use content that is produced in other workflows:

- Queries to access data for analysis.
- R Scripts for custom calculations.
- Rules to implement custom actions.
- Lookup or conversion tables created in APM (especially where a mathematical function defining the values to use is unavailable, for example, derived empirically; defined in regulations; looked up from manufacturer data such as a pump curve, etc.).

Design Policy

Persona: Solution Administrator (Policy)

Once the required policy logic is outlined and required inputs from other APM workflows are defined, design the policy by adding nodes and links to the policy design canvas and configuring them appropriately. Where a policy will take action such as creating a recommendation or sending an email alert, consideration should be given to whether the policy instance needs to be deactivated once the action is taken, to avoid duplicate requests and alerts.

Tip: The overall policy design should be outlined in advance, fully defining all calculations, queries, and look-ups that will be used, as well as decision criteria and actions to be taken in each case. For complex calculations, it may be helpful to prototype the policy using Excel or a similar program. To assist future users in understanding how the policy works, it is good practice to document the policy design in an "as-built" document, as well as assign descriptive names to policy nodes within the model itself.

Validate Ad Hoc

Persona: Solution Administrator (Policy)

Validate the policy using Ad Hoc inputs to verify that all of the required inputs to each node are configured and that the policy returns the correct values for a simple use case. After this is verified, proceed to verify that the policy handles more complex edge cases (for example, if a query used in the policy returns no rows, dividing by zero, etc).

To facilitate Ad Hoc validation, you can set up a policy instance and use the Copy to Ad Hoc feature to use instance values as a starting point. You can then edit the values as needed to test various scenarios. This

approach is necessary in certain scenarios, as some nodes (for example, the AMS Asset input node) do not feature complete Ad Hoc validation capability. In these cases, you must use an Instance to access test records in the database (for example, example event records).

Tip: When creating complex policies, it is good practice to build and validate small sections of the policy design to check your progress, rather than attempting to build the entire policy in one step.

Define Execution Settings

Persona: Solution Administrator (Policy)

Configure the policy to be executed automatically (when a change occurs in a policy instance input that is set to trigger the policy) or according to a predefined schedule. It is also possible to set a policy to execute both automatically and on a schedule, but the use cases where this is required are rare.

Activate Policy

Persona: Solution Administrator (Policy)

Once the policy and its instances have been thoroughly validated, activate the policy so that the active policy instances associated with the policy will be executed according to the defined execution settings.

Add / Edit Instances

Persona: Analyst (Policy)

Add policy instance(s) to associate a set of existing, related records in the database with the policy.

Typically, each instance of a policy would represent a different asset on which the policy will operate. In some cases, a policy could have multiple instances per asset, where each instance is related to a different input data source (for example, Measurement Location or Health Indicator).

Validate Instances

Persona: Analyst (Policy)

Validate each instance to ensure that the correct records are specified and that the expected results are returned. If you are adding a large numbers of instances, at least a significant sample of them should be validated.

Activate Instances

Persona: Analyst (Policy)

Activate the instances that you want to be executed (according to the policy's defined execution settings).

In some policies, not all instances may be active at all times. For example, you may deactivate instances related to assets that are out of service or undergoing testing or maintenance. Or, an instance may be deactivated by a Deactivate This Instance node included in the policy design to prevent repeated alerts or recommendations being generated.

Note: Activating and deactivating policy instances can be done from Asset Health Manager as well as Policy Designer.

Execute Policy

Persona: Automated Process

The policy is executed according to the execution settings.

Go to the Execute Policy workflow.

Review Policy Execution History

Persona: Analyst (Policy)

Periodically review policy execution history to ensure that policies are functioning as expected. This review may reveal policy instances that need to be reactivated, edge cases that were not considered during policy design, policy execution settings that are inappropriate for the dynamics of the process being monitored, etc.

Note: When initially deploying a new policy, or significant batches of new policy instances, reviewing the initial policy execution results should be part of the final acceptance testing of the policy. Depending on customer requirements, the initial policy execution and review might be done in a test system prior to migrating the policy into a production environment.

Execute Policy Workflow

Policy Designer: Execute Policy Workflow

This workflow describes the automated policy execution process. Depending on the input values and policy logic, policy execution may generate email alerts, as well as create, delete, or update records used in other APM workflows.

In the following workflow diagram, the blue text in a shape indicates that the corresponding description has been provided in the sections that follow the diagram. For more information, refer to the Interpreting the Workflow Diagrams topic in the APM Product Workflows documentation.

Note: For information on the personas associated with a APM module, refer to the APM Product Workflows documentation.
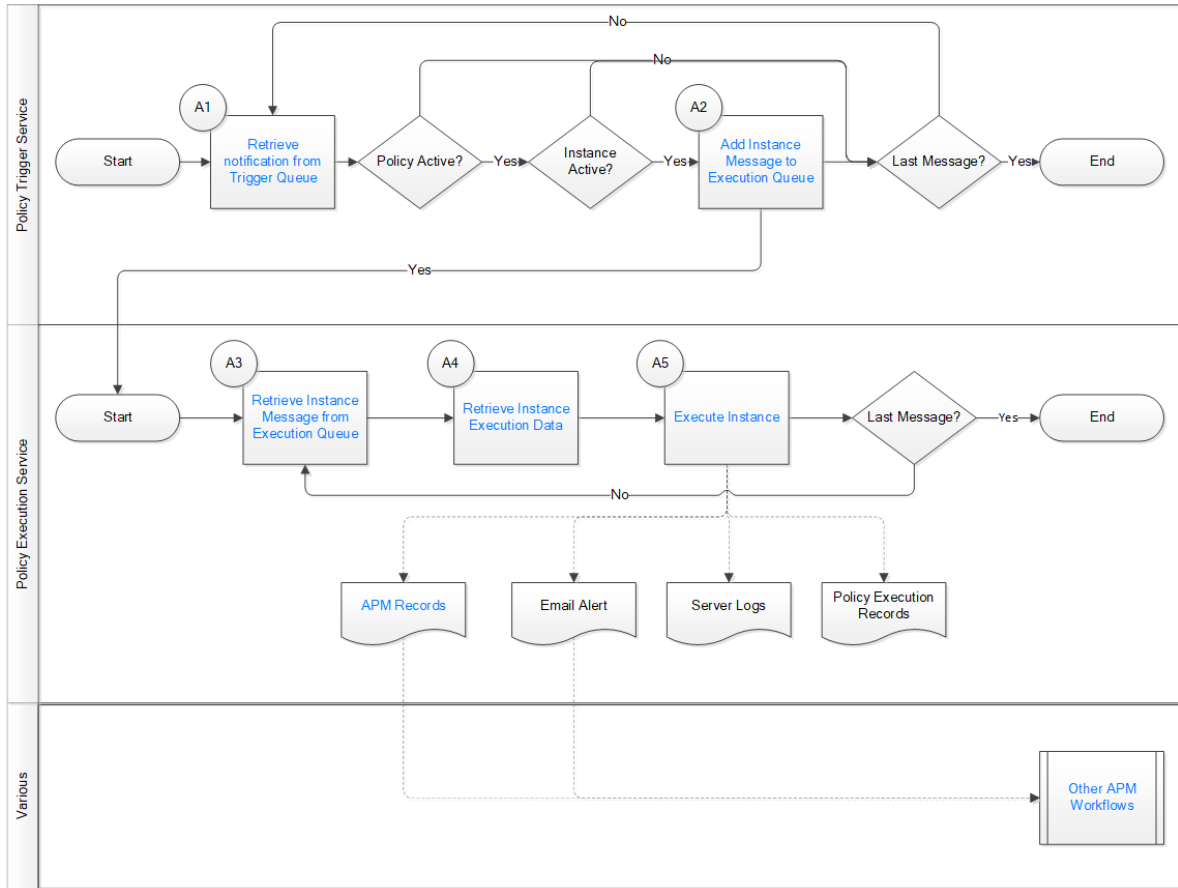
Start

Persona: Automated Process

For policies configured for scheduled execution, the policy is executed when the schedule is due (i.e., date and time configured in a policy schedule is reached).

For policies configured for automatic execution, the policy is executed when a policy trigger occurs (i.e., a change occurs in an active policy instance input that is set to trigger the policy).

Retrieve Notification from Trigger Queue

Persona: Automated Process

When any of the following occurs, a notification is added to the policy trigger queue:

- A scheduled policy is due.
- A user requests execution by selecting Execute Now in Policy Designer or selecting a hyperlink configured to execute a policy or policy instance.
- A APM record is inserted, updated, or deleted. For example, a reading is added to a health indicator, or an equipment record is modified.
- A reading is added to a process historian (OT Source Tag) that is associated with one or more policy instances through a Content Map record.

Add Instance to Policy Execution Queue

Persona: Automated Process

The policy trigger service processes each trigger message in turn to identify the relevant policy instances to be executed. Messages for inactive policies are ignored.

For a schedule trigger, all the active instances of the relevant policy will be executed.

For other triggers, the active policy instances that use a record that has changed, or that use a measurement location, health indicator, OT Connect Tag, GE Tag, or AMS Asset that has a new reading or event, as an input that is configured to trigger the policy, will be executed.

The Policy Trigger Service sends a message for each policy instance that must be executed to the policy execution queue.

Retrieve Instance Message from Execution Queue

Persona: Automated Process

One or more Policy Execution Services monitor the policy execution queue and execute the policy instances that are added to it.

Messages are processed in the order that they were added to the execution queue. However, policy execution is a multi-thread operation and may be distributed across multiple APM servers. Therefore, it is possible for a given policy execution to end after another execution that was after it in the queue.

Retrieve Instance Execution Data

Persona: Automated Process

The input data needed for each instance execution is retrieved from the APM database or from the OT Connect service.

Execute Instance

Persona: Automated Process

Depending on the policy design, the policy instance execution may create, update, or delete records in the APM database, or it may send email alerts. In addition, the policy instance may be deactivated after execution to prevent generation of multiple email messages, events, or recommendations in response to a continuing asset condition.

For each instance execution, a policy execution record may be created, depending on the policy execution history log settings, and server log messages may be added (these messages are written according to the settings in the Policy Execution Service configuration settings, and their level of detail may vary).

APM Records

Persona: Automated Process

Policies can create, update, or delete records used in any other APM workflow.

Other Workflows

Persona: Various

The results from a policy execution may be used in, or trigger, any other APM workflow.

Chapter
3

Policy Management

Topics:

About Policies

Policies are strategic plans that define actions that should be taken when certain conditions exist. A policy is made up of multiple components. Technically, a policy can exist with only a name and description, which are stored in a Policy record. However, a complete policy includes a policy model and one or more policy instances.

Policy Model

The following image shows an example of a policy model:
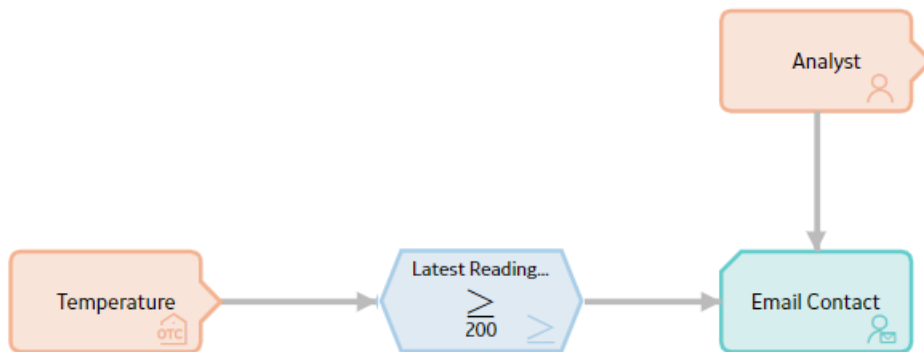
A policy model is made up of nodes and connections that define the policy logic. Specifically, the nodes in a model represent:

- The items that you want to monitor (e.g., equipment temperature).
- The conditions that should trigger actions to be taken (e.g., temperature rises above 200 degrees).
- The actions that should be taken (e.g., send an email message).

A policy model does not function like a typical logic diagram. For example, a node does not automatically evaluate the values from the immediately preceding node. Rather, for each node, you can specify an input value that is associated with any predecessor node, even if the nodes are not directly connected. Before you create a policy model, make sure that you understand the Policy Model Basic Principles.

Policy Instances

Policy instances identify the records to which logic in the policy model will be applied. In other words, you will use policy instances to identify the specific records whose values you want to monitor.

Each policy instance references one record for each input node in the policy model. For example, if a policy model contains two input nodes, one that represents the Equipment family and one that represents the Work History family, each policy instance that is associated with the policy will reference two records. The following image illustrates this example. In this image, boxes represent individual records and dotted lines represent associations between records.
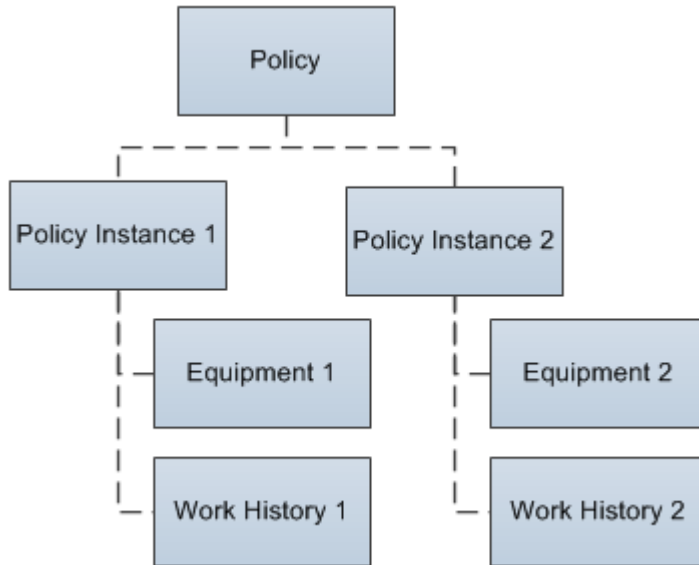
Each policy can be associated with an unlimited number of policy instances, so you can use one policy to apply the same logic to multiple records.

About Module Workflow Policies

Various policies are added during the deployment of certain modules in APM. These policies exist to support the respective module workflows, for example:

- Policies used by RBI 580, RBI 581, and Pipeline Management are used to generate recommendations based on various criteria.
- Policies used by GAA are used to perform calculations for the module.

Important: You should not modify or delete module workflow policies unless you are an expert user of the respective module. You cannot modify the name of a module workflow policy.

A list of module workflow policies is available on the **Policy Designer Overview** page. Module workflow policies are identified by the text `Module Workflow Policy for module <ModuleName>`, which appears at the top of the **Details** workspace when you open the policy.

For additional details about module workflow policies, including any required configuration steps, refer to the relevant module-specific documentation.

Reverting a Module Workflow Policy to the Baseline Version

In the event that a module workflow policy has been changed, but you want to reinstate the original baseline version (that is, the version delivered with the APM distribution package that was imported during the deployment of the respective module), you can revert the policy to baseline.

The action of reverting a policy to baseline affects only the values stored in the Policy record, not the values stored in Policy Instance records. This means that adding or editing instances does not affect whether or not a policy is considered baseline. In addition, when you revert a policy to baseline, any policy instances will be unaffected.

Similarly, policy security settings do not affect whether or not a policy is considered baseline, and any security settings will be unaffected when you revert to baseline.

Note: Taking ownership of a policy will cause the policy to be considered non-baseline. However, if you want to change only the instances or security settings of a policy, you can take ownership, make the necessary changes, and then revert to baseline.

Dates and Times Displayed in Policy Designer

The date format and time zone that are displayed on the Policy Designer interface may differ from other user or system settings that you have configured. This topic describes the specific differences you may see.

Policy Designer Date Format

Certain dates and times in the Policy Designer interface are displayed and saved in the standard ISO date format YYYY-MM-DD HH: mm: ss, where the time values use a 24-hour time format. This applies to the dates and times that are:

- Written into the body of email messages.
- Written into the **Description** field of Policy Recommendation and Policy Event records that are generated by policies.
- Included in the validation and execution result text for the Return Value node.

In these areas, the standard ISO date format is used regardless of any global date format settings or operating system settings.

Policy Time Zone

For each policy, you can specify the time zone in which the Policy Designer interface displays dates and times. For example, the time zone setting determines the dates and times that are:

- Displayed in the **Execution History** pane.
- Displayed in validation messages.
- Written into the body of email messages.
- Written into the **Description** field of Policy Recommendation and Policy Event records that are generated by policies.
- Evaluated when you specify a relative date and time.

The time zone that you specify for a policy will be used irrespective of any other user or system time zone settings.

Create a New Policy

Procedure

1. Access the **Policy Designer Overview** page, and then select **New Policy**.

   The **Policy Designer** page appears, displaying the **Details** workspace.

   Tip: If you are already viewing a policy, you can also create a new policy by selecting ⬙ in the **Details** workspace.

2. In the **Details** workspace, enter a name and description for the new policy. The policy name is required and must be unique.
3. In the **Time Zone** list, select the policy time zone.
4. On the toolbar, select 💾

A new policy is created, and you can now build a *policy model* and create one or more *policy instances* to complete the policy.

Results

- The Policy record belonging to the policy is saved to the APM database.
- You are assigned automatically as the policy owner.
- Dates and times that are associated with the policy are set to display in the time zone that you specified.
- The policy is set to inactive.

Access a Policy

Procedure

Choose one of the following methods to access a policy:

- Access the **Policy Designer Overview** page, and then select a policy name in one of the lists.
- Search for a policy using the APM search tool.

The **Policy Designer** page appears, displaying the **Details** workspace.

Take Ownership of a Policy

Procedure

1. Access the policy for which you want to take ownership.
2. In the **Details** workspace, on the toolbar, select ⬚.

   Your name appears next to the ⬚ button.
3. On the toolbar, select 🖫.

   The policy is saved.

Results

- You are set as the owner of the policy.
- You can modify the policy.

Copy a Policy

About This Task

You can create a new policy by copying an existing policy, either with or without its associated instances.

Procedure

1. Access the policy that you want to copy.
2. In the **Details** workspace:

- If you want to copy the policy only, select 🖫.
- If you want to copy the policy and its associated instances, select ⧉.

The **Save Policy As** dialog box appears.
3. In the **Policy Name** box, enter a name for the new policy. The policy name is required and must be unique.
4. Select **OK**.
   The new policy appears.

Results

- You are assigned automatically as the owner of the new policy.
- The new policy is set to inactive.
- All other policy and instance (if applicable) settings match the original policy.

Refresh Metadata for Policies

About This Task

If the metadata used by a policy or node of a policy model is modified, you can refresh the metadata for the policy and its nodes so that they use the updated metadata. You can refresh the metadata for the policy or for specific nodes of the policy model.

Procedure

1. Access the policy for which you want to refresh the metadata.
2. In the **Design** workspace, perform one of the following steps:

   - If you want to refresh the metadata for the policy, in the **Edit** section of the toolbar, select 🔄.
     The metadata for the policy is refreshed.
   - If you want to refresh the metadata for a node, select the node, and then in the **Properties** window for the node, select 🔄.
     The metadata for the node is refreshed.

Delete a Policy

Procedure

1. Access the policy that you want to delete.

2. On the toolbar, select 🗑.
   A dialog box with a confirmation message appears.
3. Select **OK**.
   The policy is deleted from the APM database.

Revert a Module Workflow Policy to the Baseline Version

Procedure

1. Access the policy that you want to revert to the baseline version.
2. In the **Details** workspace, select **Revert to Baseline**.

   Note: This button appears only for module workflow policies and is enabled only when the policy has been modified from its original version.

   The **Confirm Revert to Baseline** dialog box appears.
3. Select **OK**.

   The **Close Policy Tab** dialog box appears.
4. Select **OK**.
5. Close the tab. If you want to view the baseline version, open the policy again.

Chapter
4

Security Settings

Topics:

Configure Security Settings for a Policy

Before You Begin

Ensure that you understand how individual policy security will affect access to the policy for you and all users.

About This Task

If you want only certain users within a Policy Designer Security Group to be able to access a policy, you can optionally configure security settings on individual policies.

Procedure

1. Access the policy for which you want to configure security.
2. Select the **Security** tab.

   The **Security** workspace appears.
3. Specify security for a single user or a group:

   a) To specify security for a single user, select ⛁.

      The **Assign Users** window appears.

   b) To specify security for users within a Security Group, select 👥.

      The **Assign Groups** window appears.
4. Select the users or groups for which you want to specify security, and then select **Update**.

   The selected users or groups appear in the grid in the **Security** workspace.
5. In the **Permissions** column of the grid, select the level of security you want to apply.

   Important: Ensure that the selections you make do not inadvertently remove your own access to the policy. If you do, you will no longer be able to make changes to the security. In that case, only another user designated as a Designer for this policy or a Super User would be able to make changes. See About Policy Security Ownership for more information.
6. Select 💾.

   Your changes are saved.
7. If the security settings result in you no longer having Designer permission on the policy, the Confirm Security Settings window is displayed.

   If the permissions change is as intended, select **OK** to save the policy.

   -or-

   If the permission change is not as intended, select **Cancel** and repeat step 3 through step 6 to correct the issue.

Results

- Access to the policy is restricted based on your selections and the Policy Designer Security Groups to which users belong.
- On the **Policy Designer Overview** page, Yes appears in the **Locked** column of any list that contains the policy.

Modify or Remove Security Settings for a Policy

Before You Begin

Ensure that you understand how individual policy security will affect access to the policy for you and all users.

Procedure

1. Access the policy for which you want to modify or remove security settings.
2. Select the **Security** tab.

   The **Security** workspace appears.
3. To remove or modify security settings for a user or group, choose one of the following:

Fields	Description
Remove security settings	a. In the grid, select the check box in the row for the user or group whose security settings you want to remove. b. Select 🗑. c. In the **Confirm Delete** dialog box, select **Yes**.
Modify security settings	a. In the **Permissions** column of the grid, select or clear the check boxes as appropriate. b. Select 💾.

Important: Ensure that the selections you make do not inadvertently remove your own access to the policy. If you do, you will no longer be able to make changes to the security. In that case, only another user designated as a Designer for this policy or a Super User would be able to make changes. See About Policy Security Ownership for more information.

Your changes are saved.
4. If the security settings result in you no longer having Designer permission on the policy, the Confirm Security Settings window is displayed.

   If the permissions change is as intended, select **OK** to save the policy.

   -or-

   If the permission change is not as intended, select **Cancel** and repeat step 3 through step 6 to correct the issue.

Results

- Access to the policy is restricted based on your selections and the Policy Designer Security Groups to which users belong.
- If you removed all specific security settings for the policy, on the **Policy Designer Overview** page, Yes no longer appears in the **Locked** column of any list that contains the policy.

Chapter

5

Upgrade Logs

Topics:

- Review Upgrade Logs for a Policy
- Delete Upgrade Logs for a Policy

Review Upgrade Logs for a Policy

About This Task

When the APM database is upgraded to V5.0.0.0.0 or later from an earlier version, the logs that provide information, such as modifications to existing policy models, upgrade issues that were not addressed automatically, and the applicable APM version number, are saved to the policy records. You can review this information and identify changes that are required to ensure that the policy continues to function as expected after the upgrade.

Note: The Upgrade Logs tab is displayed only if the policy has a value in the Upgrade Log field. When you first execute or access the policy after your APM system is upgraded, the policy record may be upgraded, and the Upgrade Log field updated. However, if you use a query to view the value in this field, it may not reflect the final upgrade status.

Procedure

1. Access the policy for which you want to review the logs.
2. Select the **Upgrade Logs** tab.
   The **Upgrade Logs** workspace appears.
3. Review the upgrade log information.

Results

The log includes information and warning messages. If there are any warning messages, you may need to modify the policy model for the policy to function as originally intended.

Delete Upgrade Logs for a Policy

Before You Begin

You must be a member of the MI Policy Designer security group to delete the upgrade log.

About This Task

You can delete the policy upgrade log information that is no longer needed.

Procedure

1. Access the policy for which you want to delete the upgrade logs.
2. Select the **Upgrade Logs** tab.
   The **Upgrade Logs** workspace appears.
3. Select **Delete Upgrade Logs**.
   A window with a confirmation message appears.
4. Select **OK**.

Results

The upgrade log information is deleted from the APM database; it is replaced with the APM version number, user ID, and the timestamp indicating when the upgrade log was deleted.

Chapter

6

Policy Models

Topics:

Policy Model Basic Principles

A policy model is made up of nodes and connections that define the policy logic. In order to build a functioning policy model, you must understand several basic principles.

Configuring a Policy Model

The following principles apply to working with a policy model:

- Policy models do not reference specific records. Rather, they contain nodes that reference families. You must use *policy instances* to identify the individual records whose values are evaluated when the policy is executed.
- The initial nodes in a policy model, that is, nodes with no predecessors, must be Input nodes other than Query nodes.
- A family policy must contain one Current Entity node for entity families, or one Current Relationship node for relationship families.
- A node can use an input from any predecessor node in the same logic path, even if the nodes are not directly connected.
- Any number of nodes can use an input from the same predecessor node.
- With the exception of the Or node and the Case node, a node will be executed only when all necessary preceding nodes have been successfully executed.
- A policy model can often be arranged in various configurations without impacting the execution results. You may want to arrange the policy model in the configuration that provides the best visual representation of the policy.
- Connections that start at a Condition or Logic node can be configured to create separate logic paths in a policy model. Specifically:

  ◦ If the connection property is Yes, the corresponding path will be followed when the logical result of the Condition of Logic node is yes. If you do not configure a logic path for a connection, a Yes path is assumed but does not appear on the model.
  ◦ If the connection property is No, the corresponding path will be followed when the logical result of the Condition or Logic node is no.

Configuring Node Properties

- Most nodes have outputs that successor nodes can use as inputs. You must specify inputs for each successor node using the **Properties** window that appears when you select the node in the policy model.

  Note: Outputs and inputs may represent either a single value or a *collection* of values. The types of outputs that each node generates and the types of inputs that each node accepts is different for each node. When building a policy model, you must use corresponding input and output types.
- Any numeric values entered in Calculation nodes should be entered in the format matching the user's culture setting. For example, a user with German culture would enter 4,5 to represent four and a half, whereas a user with US-English culture would enter 4.5.
- There are specific formats in which you can enter dates and times and amounts of time (that is, time spans). Refer to the respective topics for details.
- When using a policy node to update values in a record:

  ◦ If a field has complex behavior defined by field-level rules (for example, rules for valid values) and field-level behaviors, this behavior will not be reflected in the **Properties** window or detected by policy validation. Therefore, you are responsible for ensuring that the values you specify are valid according to any baseline or custom field-level rules for the corresponding field.

◦ If a field value is defined by a system code, the value that you specify in the corresponding section must be the system code, not the value that is displayed to the end user.

Policy Model Principles Illustrated

The principles for working with a policy model can be illustrated through the following example model:

In this example model, the node named Temperature is an OT Connect Tag node which represents a process historian tag. When this policy is executed, if the Latest Reading Value that is associated with the process historian tag is greater than or equal to 200, the APM system will send an email message to the email address that is specified in the Human Resource record that is associated with the Analyst User node.

The following table describes each of the policy model basic principles in the context of this example:

Policy Model Principle	Example
Policy models do not reference specific records. Rather, they contain nodes that reference families. You must use *policy instances* to identify the individual records whose values are evaluated when the policy is executed.	The Temperature and Analyst nodes represent families. The specific records whose values are evaluated are determined by policy instances.
The initial nodes in a policy model, that is, nodes with no predecessors, must be Input nodes other than Query nodes. A family policy must contain one Current Entity node for entity families, or one Current Relationship node for relationship families.	The Reading in Error node is a Current Entity node, which is a type of Input node. The Current Entity node is required for an entity family policy.

Policy Model Principle	Example
Most nodes have outputs that successor nodes can use as inputs. You must specify inputs for each successor node using the **Properties** window that appears when you select the node in the policy model. **Note:** Outputs and inputs may represent either a single value or a collection of values. The types of outputs that each node generates and the types of inputs that each node accepts is different for each node. When building a policy model, you must use corresponding input and output types.	The output Latest Reading Value from the Temperature node is used as an input to the Condition node. This output represents a single value, which corresponds to the type of input that the Condition node accepts. The value 200 is used as the second input to the Condition node, but it is not an output from another node. Instead, it is a constant value that is specified directly in the **Properties** window for the Condition node.
A node can use an input from any predecessor node in the same logic path, even if the nodes are not directly connected.	The Email Contact node can use an input from the Temperature node even though the two nodes are not directly connected.
Any number of nodes can use an input from the same predecessor node.	The Email Contact node and the Condition node can both use inputs from the Temperature node.
With the exception of the Or node and the Case node, a node will be executed only when all necessary preceding nodes have been successfully executed.	The Email Contact node will be executed only when all of the nodes preceding it have been successfully executed. If, for example, the condition defined in the Condition node was not met or if an error occurred when executing the Analyst node, the Email Contact node would not be executed.

Policy Model Principle	Example
A policy model can often be arranged in various configurations without impacting the execution results. You may want to arrange the policy model in the configuration that provides the best visual representation of the policy.	The execution results of the policy would be identical even if the Analyst node were connected to the Temperature node or the Condition node. The current configuration, however, provides a clear visual representation of the policy because the Email Contact node is the only node in the model that uses an input value from the Analyst node.
Connections that start at a Condition or Logic node can be configured to create separate logic paths in a policy model. Specifically: • If the connection property is Yes, the corresponding path will be followed when the logical result of the Condition of Logic node is yes. If you do not configure a logic path for a connection, a Yes path is assumed but does not appear on the model. • If the connection property is No, the corresponding path will be followed when the logical result of the Condition or Logic node is no.	A property is not defined for the connection between the Condition node and the Email Contact node, therefore, a value of Yes is assumed. This means that an email message is sent only if the preceding condition is true. If the condition is false, policy execution will not continue.

Add Nodes to the Model Canvas

Before You Begin

- Create a policy.
- Make sure you understand the basic principles of working with a policy model.

Procedure

1. Access the policy to which you want to add nodes.
2. In the **Design** workspace, in the section of the toolbar for the type of node that you want to add, select the button for the node and drag it to the model canvas.

For example, if you want to add a Constant node, on the **Inputs** toolbar, select the ● button and drag it to the model canvas.
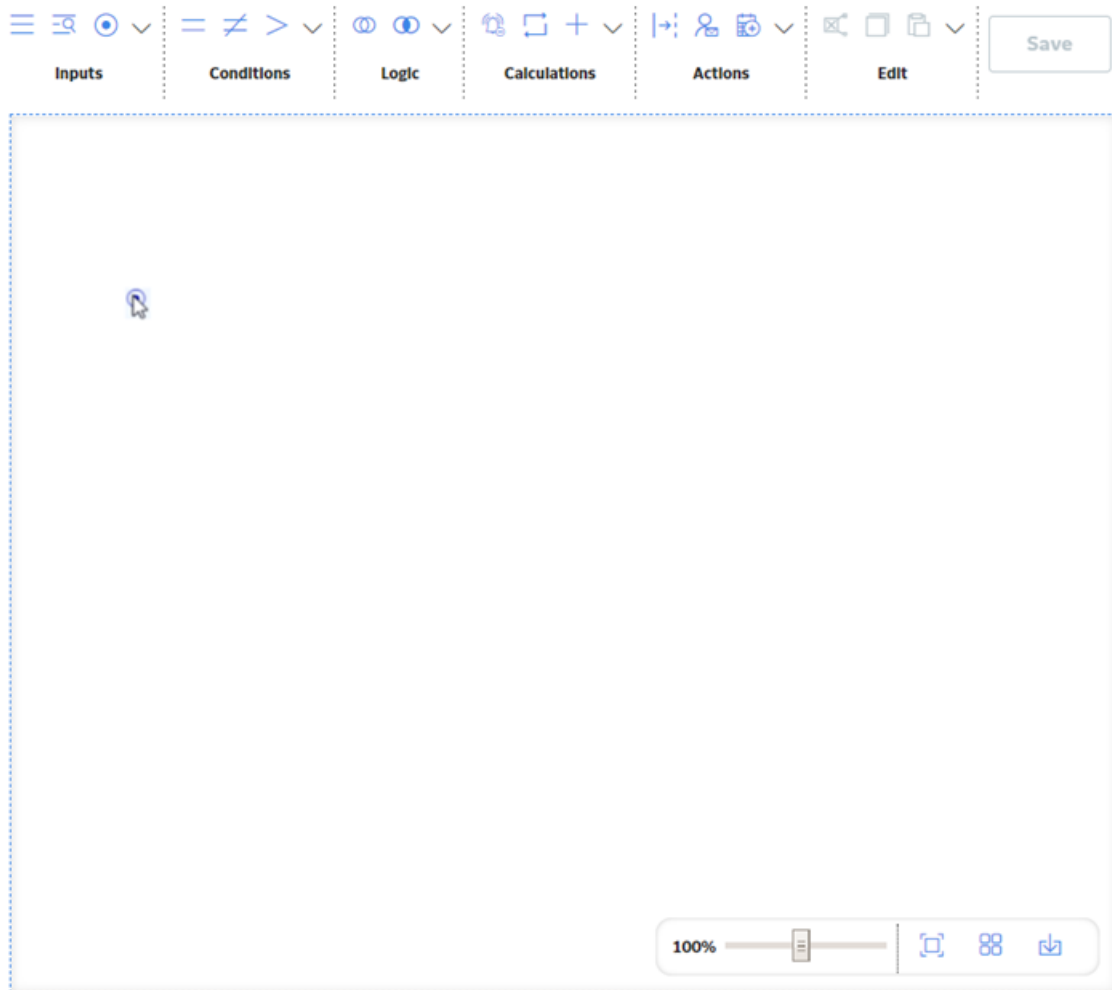
Tip: In each section of the toolbar, you can select the ▶ button to see additional nodes and to see what each button represents.

3. In the location where you want the node to appear, release the pointer.

The node appears on the model canvas and the **Properties** window for the node is displayed.
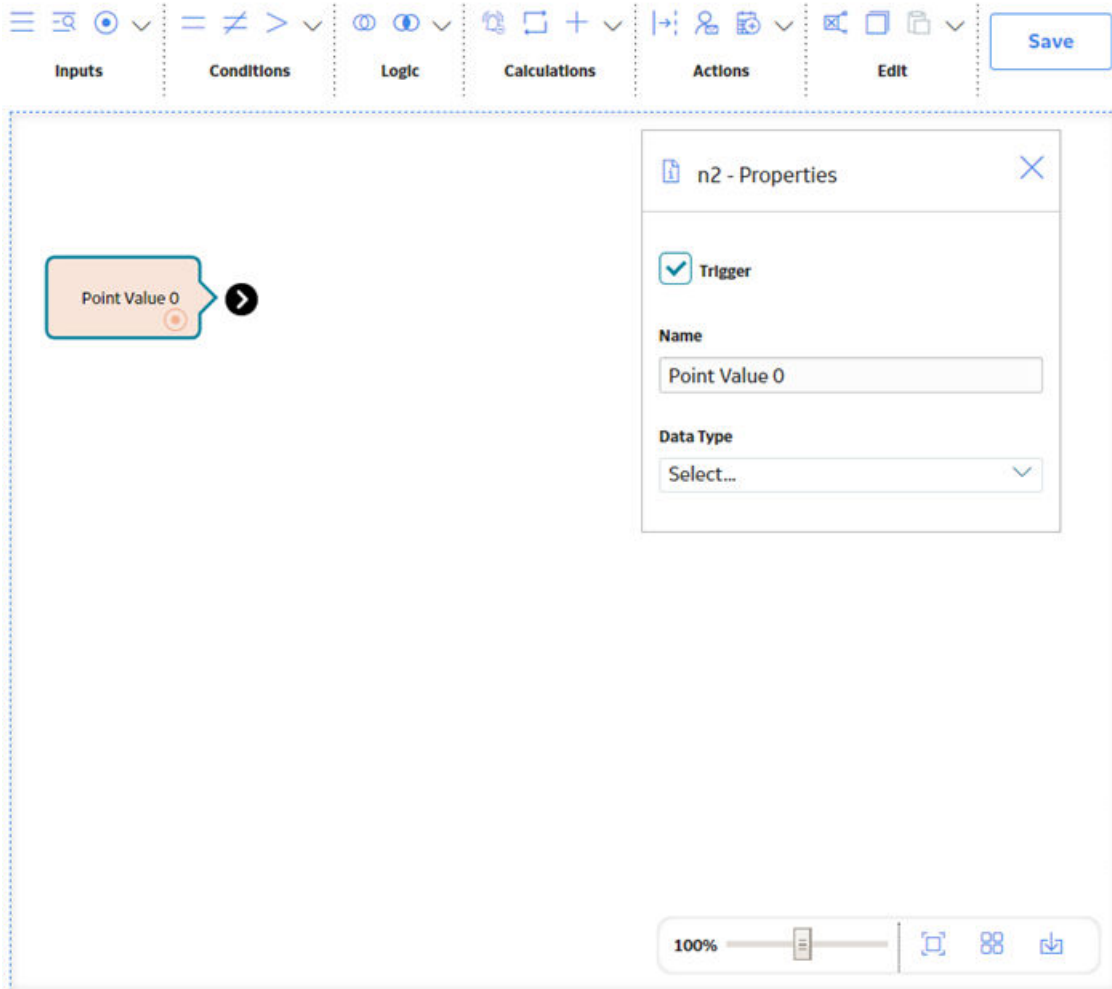
4. Select **Save**.
   The policy is saved.

Enable Grid in Model Canvas

About This Task

You can enable the grid in the model canvas. The grid allows you to easily align the nodes in the policy model with the help of the alignment lines that appear on moving the nodes on the canvas.

Procedure

1. Access the policy containing the policy model for which you want to enable the grid.

2. In the **Design** workspace, on the model canvas, select ⊞.
   The grid appears in the model canvas.

Connect Nodes in a Policy Model

Before You Begin

- Make sure you understand the basic principles of working with a policy model.
- Add at least two nodes to the model canvas
- Note the following limitations with connections between nodes:
  - You cannot connect nodes in a circular execution path (for example, if Node 1 is connected to Node 2, and Node 2 is connected to Node 3, you cannot connect Node 3 to Node 1).
  - You cannot connect a node to itself.
  - There cannot be more than one connections between two nodes.

Procedure

1. Access the policy in which you want to connect the nodes.
2. In the **Design** workspace, select the node that you want to connect to another node.

   The ⊗ icon appears for the node.

3. Select ⊗, and then drag the connector to the centre of the successor node.

   The nodes are connected.

   Note: You can select a point on the connector to add a vertex and bend the connector. You can also drag the vertex to any position on the model canvas.

4. Select **Save**.
   The policy is saved.

Configure Node Properties

Before You Begin

- Make sure you understand the basic principles of working with a policy model.
- Add and connect nodes in the policy model.

Procedure

1. Access the policy that contains the node whose properties you want to configure.
2. In the **Design** workspace, select the node whose properties you want to configure.

   The **Properties** window for the node appears. The following image shows an example of the **Properties** window for a Threshold Statistics node.

3. If the **Properties** window contains a **Name** text box and you want to use a name other than the default name, enter a name for the node. For some nodes, the name that you specify also appears as the node label in the policy model.
4. Define any input values for the node.
5. Configure any additional properties that are specific to the node.

   Tip: For details about the properties that you can configure for each node, refer to the following sections in this documentation: Input Nodes, Condition, Logic, and Calculation Nodes, and Action Nodes.
6. Repeat these steps for each node in the policy model.
7. Select **Save**.

Define Input Values

Before You Begin

- Make sure you understand the basic principles of working with a policy model.
- If you want the input value to be defined by a predecessor node, the nodes must be connected, either directly or indirectly.

About This Task

For each node that requires an input, in each input section of the node's **Properties** window, you can specify a user-defined constant value or a value or collection that is the output of a predecessor node.

Steps:

Procedure

- Specify a user-defined constant value

  1. Access the policy that contains the node for which you want to define an input value.

  2. On the **Properties** window for the node, verify that the ✎ icon appears next to the input text box.

  3. In the input text box, enter or select the value that you want to use as an input.

     Note: If the input section supports multiple values, you must separate each value with a comma.

     The following image shows an example of a constant input value.

  4. In the **Policy** section of the toolbar, select 🖫.
     The policy is saved.

- Specify a value or collection that is an output of a predecessor node:

  1. Access the policy that contains the node for which you want to define an input value.

  2. On the **Properties** window for the node, select the ✎ icon that appears next to the input text box.

     The icon changes to ☰ and the text box changes to a drop-down list. This list contains the node's predecessor nodes.

  3. In the list, select the node that is associated with the value or collection that you want to use as the input.

     A second list appears. This list contains the output options that are associated with the selected predecessor node.

  4. In the second list, select the field or collection that you want to use as the input.
     The following image shows an example of an input value that is defined by a predecessor node.
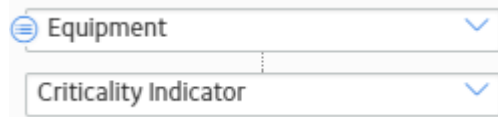
     In this image:

     ◦ The input node is Equipment.
     ◦ The input value comes from the value in the Criticality Indicator field in the Equipment record (the specific Equipment record will be defined by a policy instance).

     Some input fields allow you to map a specific value from a collection. In this case, a third list appears.

  5. If applicable, in the third list, select the column in the collection that contains the values that you want to use as inputs.

  6. In the **Policy** section of the toolbar, select 🖫.
     The policy is saved.

Define Input Values

To illustrate the different ways to define input values, consider the following nodes and connections.
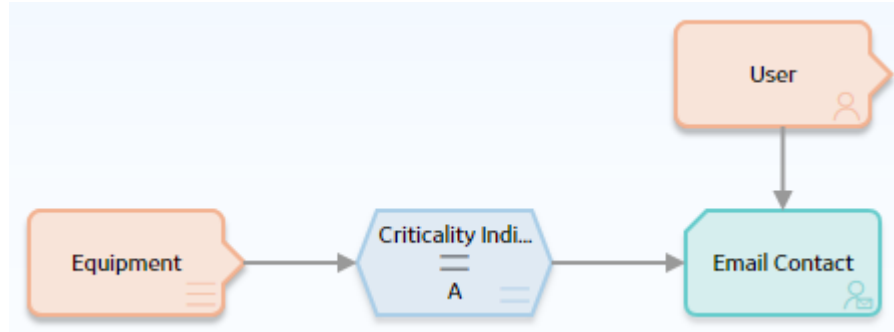
In this example, the Equipment Entity node is connected to an Equal Condition node. The **Properties** window for the Condition node, therefore, allows you to select the Equipment family and one of its fields. As shown in the following image:

- The ⊜ icon appears in the first input section, indicating that the input is defined by a predecessor node.

- The ⊘ icon appears in the second input section, indicating that the input is a user-defined constant value.

Together, the Equipment Entity node and the Condition node to which it is connected indicate that an email message should be sent when an Equipment record has a value of A in the Criticality indicator field.

Note: You will use policy instances to identify the specific record(s) whose values you want to use in the policy. For the example provided in this topic, you would use a specific Equipment record in each policy instance that is associated with the policy.

Configure Logic Paths

Before You Begin

- Make sure you understand the basic principles of working with a policy model.
- Add and connect nodes in the policy model.

About This Task

Connections that start at nodes that have a logical result output can be configured to create separate logic paths in a policy model. Specifically, you can specify whether or not a successor node will be executed based on the logical result of the preceding node. The APM system will execute only the branches of a policy model where the logical result of a node matches the logic path defined for the corresponding connection.

Note: If you do not configure a logic path for a connection, a Yes path is assumed but does not appear on the model.

Procedure

1. Access the policy within which you want to configure logic paths.
2. In the **Design** workspace, perform one of the following steps:

    - Select the connection for which you want to specify a logic path.

    - Point to the connection for which you want to specify a logic path, and then select ⚙.

    The **Properties** window for the connection appears.
3. In the **Logic Path** box, specify whether the connection should be followed if the logical result of the preceding condition is true or false. Specifically:

    - Select **Yes** to specify that the connection should be followed when the logical result of the preceding node is true.
    - Select **No** to specify that the connection should be followed when the logical result of the preceding node is false.
4. Select **Save**.
   The policy is saved.

> **Separate Logic Paths**
>
> Consider the following policy model:

In this example, you can see that:

- The Condition node evaluates the DA Reading Value that is associated with the OPC Tag record to determine if it is greater than or equal to 900.
- The Condition node is connected to the following nodes:
    - Email Analysis (through the Yes connection)
    - Create Recommendation (through the No connection)

Based on the connection properties, the following logic will be applied when the policy is executed:

- If the DA Reading Value is greater than or equal to 900, an email message will be sent. The logic is determined by the Yes logic path specified for the connection to the Email Analyst node.
- If the DA Reading Value is not greater than or equal to 900, a Policy Recommendation record will be created. This logic is determined by the No logic path specified for the connection to the Create Recommendation node.

Copy and Paste Nodes and Connections

Before You Begin

- Make sure you understand the basic principles of working with a policy model.
- Note the following:
    - If you copy a node that contains a mapped field value from another node, the mapped field value will not be copied unless you also copy the node from which the values are mapped and the connector node between the two nodes.
    - Connections will be copied only if you select the connection and both nodes that it connects.
    - After you paste a node that requires a unique name (example; Point Value node), you must change the name of the copied node before you can save the policy.
    - You cannot copy nodes and connections from one policy to another.

Procedure

1. Access the policy containing the nodes that you want to copy and paste.

2. In the **Design** workspace, on the model canvas, press the Ctrl key and select all the nodes and connections that you want to copy.

   Tip:
   - You can press Ctrl + A to select all nodes and connectors in the policy model.
   - You can select a point on the model canvas and then drag the pointer to select all nodes and connections within a rectangular region.

   The selected nodes and connections are outlined in blue, as shown in the following image.
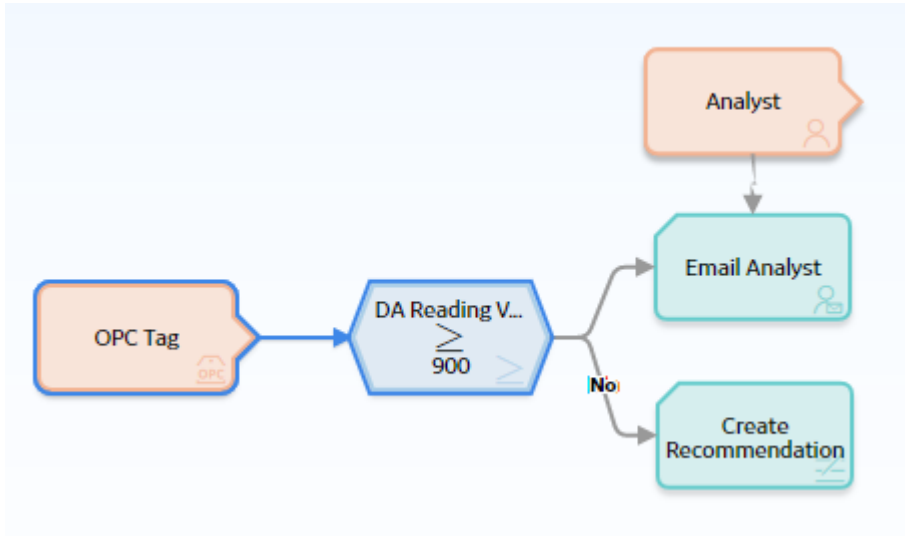
3. In the **Edit** section of the toolbar, select .

   The selected nodes and connections are copied.

4. In the **Edit** section of the toolbar, select .

   Copies of the selected nodes and connections are pasted to the model canvas and are selected automatically.
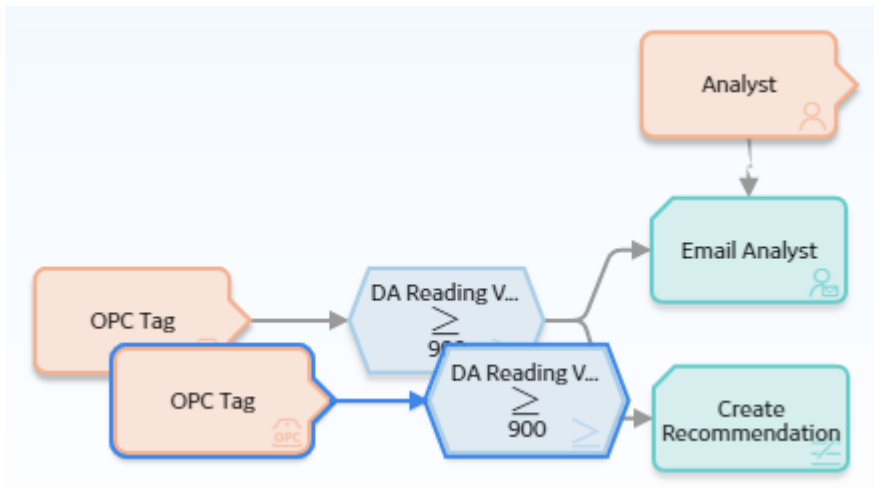
5. Drag the pasted nodes and connections to a new location as necessary.
6. Connect and configure the pasted nodes as needed.
7. Select **Save**.
   The policy is saved.

Download Image of Policy Model

About This Task

You can download the image of a policy model to your local drive.

Procedure

1. Access the policy containing the policy model for which you want to download the image.

2. In the **Design** workspace, on the model canvas, select ⤓.
   The image of the policy model is downloaded in the Portable Network Graphics (PNG) format to the default download location specified for your internet browser.

Chapter

7

Policy Instances

Topics:

About Primary Records and Primary Nodes

The primary record is a record in a *policy instance* to which most or all other records in the instance are linked. The primary record may or may not be represented by a node in the *policy model*. Not every policy will be configured such that its instances have primary records, but if it is, you can use the primary record feature to streamline the process of creating policy instances.

Typically when creating a policy instance, for each node in the policy model, you must access a **Search** window and configure the appropriate search criteria in order to locate the record that you want to assign to the node. If, however, you specify a primary record, the search criteria are configured automatically to return only records that are linked to the primary record. Further, rather than using a **Search** window at all, you can simply select a record from a list of records that are linked to the primary record.

If the primary record is represented by a node in the policy model, before you create a policy instance, you can specify that node as the primary node. Then, in each policy that you create, the record that you assign to the primary node will become the primary record automatically.

If the primary record is not represented by a node in the policy model, you can specify the primary record directly in each policy instance.

Primary record represented by a node in the policy model

Consider the following policy that monitors various types of information for a piece of equipment.

The Equipment node in this model represents an Equipment record to which all of the other input records (represented by other Input nodes) are linked. Therefore, the Equipment node has been specified as the primary node, as indicated by the blue star that appears on the node in the model.

In a policy instance associated with this policy, a blue star appears next to the node name, indicating that the record specified for the Equipment node will be the primary record. In the following image, an Equipment record with the Record ID ~ #1 STEAM REHEATER ~ HXST90 has been specified for the Equipment node and is, as a result, also the primary record for this policy instance.

Primary record not represented by a node in the policy model

Consider the following policy that monitors various types of information for a piece of equipment.
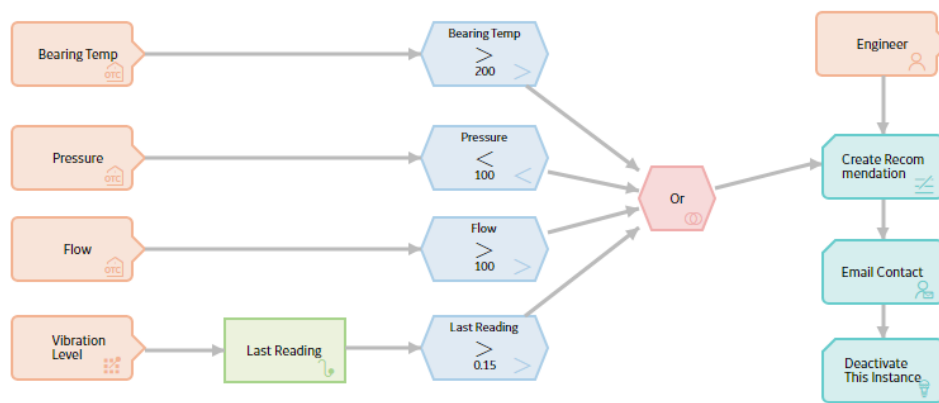
The Input nodes in this policy model represent records that are related to a single Equipment record, but the Equipment record itself is not represented by a node in the model. Therefore, there is not a primary node in this policy model. However, you can still specify a primary record in each policy instance.

In a policy instance that is associated with this policy, the top row contains a blue star that indicates the primary record, but the row does not contain a node name because there is not a primary node in the model. In the following image, an Equipment record with the Record ID ~ #1 STEAM REHEATER ~ HXST90 is specified as the primary record.
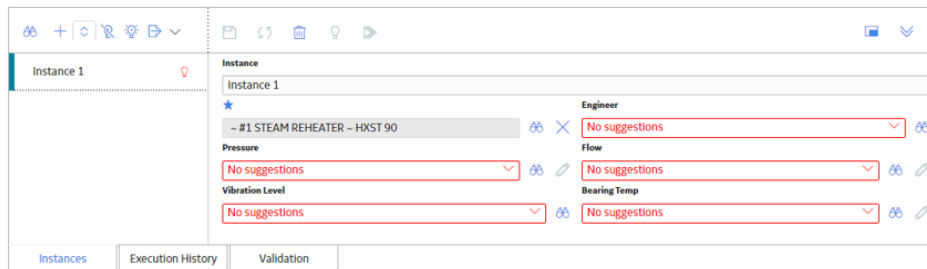
Assigning records to nodes after a primary record is specified

Continuing with the previous examples in this topic, after a primary record has been specified, when you assign records to the remaining nodes in the model, you can

select the record you want from the list that is enabled. As shown in the following image, the list contains records in the database that are linked to the Equipment record ~ #1 STEAM REHEATER ~ HXST90.
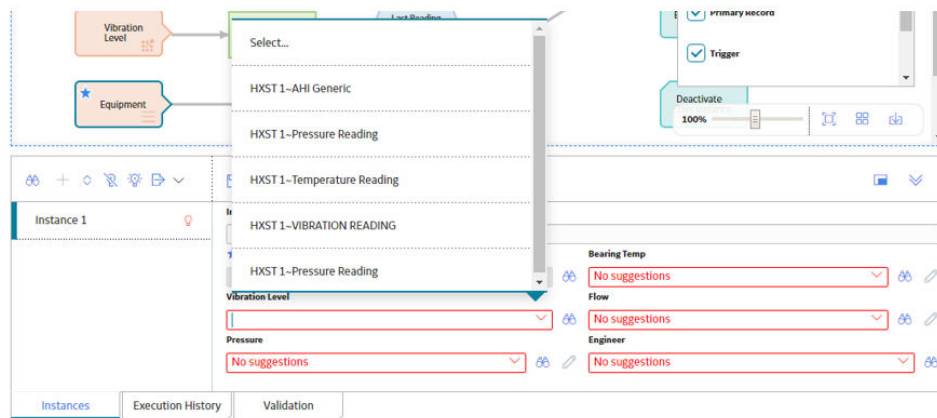
Note: For OT Connect Tag nodes, the list is never populated, even if the Source Tags have been previously associated with the primary record through a policy instance or health indicator. You must use the **Search** window to select a Source Tag.

You can also enter a Record ID directly in the cell. When you do this, the list is re-populated with the linked records whose Record IDs contain the value that you entered, as shown in the following image.
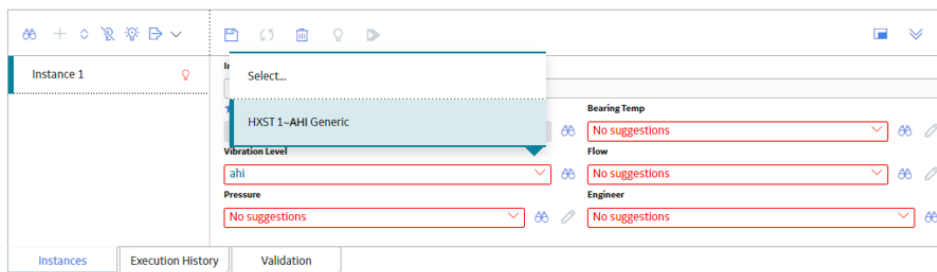
Alternately, you can use the **Search** window to search for and select a record to assign to an Input node. When you do this, the search criteria that are defined by default will return only records that are linked to the Equipment record specified in the instance.

Note: The layout of the **Search** window varies depending on the type of the input node.

Specify the Primary Node in a Policy Model

Before You Begin

- Build a policy model.
- Note the following limitations that apply to primary nodes:
  ◦ You can specify only one primary node in a policy.
  ◦ Any Input node that represents a record in the APM database, for example, an Entity node, can be specified as the primary node.

About This Task

If a policy contains a node that represents a primary record, you can specify that node as the primary node in a policy model. Doing so will streamline the process of creating policy instances.

Procedure

1. Access the policy for which you want to specify a primary node.
2. In the **Design** workspace, select the Input node that you want to specify as the primary node.
   The **Properties** window for the Input node appears.
3. On the **Properties** window, select the **Primary Record** check box.

   A blue star appears on the upper-left corner of the node in the policy model, indicating that the node is the primary node in the policy.

4. Select **Save**.
   The policy is saved.

Results

- When you create a policy instance, the record that you assign to the primary node will be the primary record for that instance.

Create a Policy Instance

Before You Begin

- Build a policy model.
- Specify the primary node, if there is one.

Procedure

1. Access the policy for which you want to create a policy instance.
2. At the bottom of the **Design** workspace, select the **Instances** tab.
   The **Instances** pane appears.

   Tip:

   - To resize the pane, drag the top edge of the pane.
   - To maximize the pane, select ⬈.
   - To restore the pane to its original size, select ⬂.
   - To close the pane, select ⌄.

3. On the left side of the **Instances** pane, select ＋.

   A new policy instance appears, which includes options to assign a record to each Input node in the policy (excluding Query and Constant nodes whose values are not specified in policy instances).

4. Optionally, in the **Instance** box, enter a name for the instance other than the default name.

5. If you want to use the primary record feature to streamline the process for assigning records to nodes, select 🔍 next to the text box under the blue star.

   Note: Depending on whether or not you have specified a primary node, this text box may or may not correspond to a node in the policy model.

   or

   If you do not want to use the primary record feature, select 🔍 next to the text box corresponding to a node in the policy model.

   The **Search** window appears. The search criteria are defined automatically to return only records in the family represented by the selected node.

   Important: If a family has been excluded from the global search, you will not be able to assign records from that family to a policy instance.

   Note: The appearance of the **Search** window varies depending on the type of the selected input node.

6. Specify additional search criteria as needed, and then select 🔍.

7. In the search results, select the record that you want to assign to the policy instance, and then select **OK**.

   The **Search** window closes, and the Record ID of the selected record appears in the corresponding text box.

   The following image shows an example in which the Equipment record with the Record ID ~ #1 STEAM REHEATER ~ HXST130 has been assigned to the node Equipment. The blue star above the text box indicates that ~ #1 STEAM REHEATER ~ HXST130 is also the primary record.
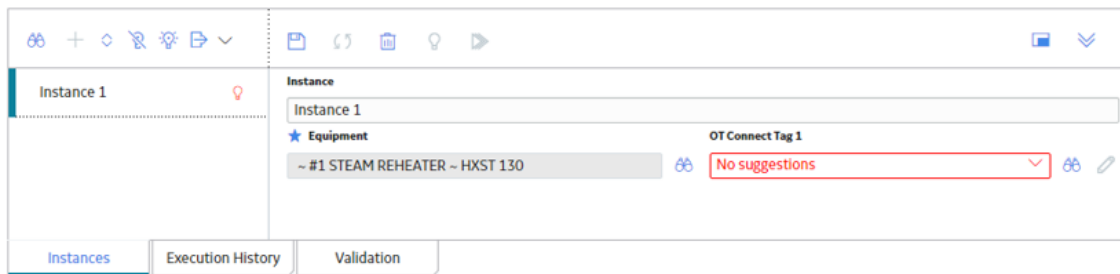
8. Choose one of the following methods to assign a record to each remaining node included on the right side of the **Instances** pane.

   • If you have specified a primary record, you can do this in one of the following ways:

   ◦ Select a record from the list in the corresponding text box. This list contains the records in the corresponding family that are linked to the primary record.

   ◦ Enter part of a Record ID directly in the text box. When you do this, the list is re-populated with the records whose Record IDs contain the value that you entered.

   ◦ Select 🔍 to access the **Search** window. By default, the search criteria are defined to return only records that are linked to the primary record.

   • If you have not specified a primary record in the policy, repeat steps 5 through 7 (that is, use the Search window to assign records to each node).

   Note: If a policy model contains a Health Indicator node, within a policy instance, instead of assigning an existing Health Indicator record to the node, you can optionally create a new Health Indicator record to assign to the node.

9. On the **Instances** pane, select 💾.

The policy instance is saved.

Results

- A Policy Instance record representing the policy instance is saved to the APM database.

Create a New Health Indicator Record from the Instances Pane

Before You Begin

- Build a policy model.
- Create a policy instance.

About This Task

If a policy model contains a Health Indicator node, within a policy instance that is associated with that policy, you can assign an existing Health Indicator record to that node or create a new Health Indicator record to assign to that node. The following instructions explain how to create a new Health Indicator record that will be assigned to a Health Indicator node. You can then use the Add Value to Health Indicator node to create Health Indicator Value records and link them to the new Health Indicator record.

Procedure

1. Access the policy that is associated with the policy instance for which you want to assign a new Health Indicator record to a Health Indicator node.
2. At the bottom of the **Design** workspace, select the **Instances** tab.
   The **Instances** pane appears.
3. On the right side of the **Instances** pane, next to the text box for the Health Indicator node to which you want to assign a new Health Indicator record, select ╋.
   The **Create Health Indicator** window appears.
4. Next to the **Asset** text box, select 👓, and then use the **Hierarchy Finder** window to select the asset record to which you want to link the Health Indicator record (e.g., a record in the Equipment family).
5. In the **Name** text box, enter a name for the health indicator.
6. Optionally, in the **Description** text box, enter a description for the health indicator.
7. In the remaining text boxes, enter values to indicate the thresholds against which values associated with this health indicator will be compared to determine the status of the health indicator.
8. Select **Create**
   The new Health Indicator record is created, and the **Instances** pane is updated to show the Record ID of the new record.
9. On the **Instances** pane, select 💾.
   The policy instance is saved.
10. Repeat these steps for each instance that is associated with the policy.

Configure OT Connect Tag Limits from the Instances Pane

Before You Begin

- Build a policy model.
- Create a policy instance.

About This Task

If a policy model contains an OT Connect Tag node, within a policy instance that is associated with that policy, you can configure the limit values that must be evaluated during the policy execution. These limit values are associated with the use of the selected Source Tag only in this specific policy instance and are stored in a Content Map record associated with the policy instance and the Source Tag. This means that you can use different limits in each policy instance where a Source Tag is used, depending on the purpose of the policy.

Procedure

1. Access the policy that is associated with the policy instance for which you want to configure the OT Connect Tag limits.
2. At the bottom of the **Design** workspace, select the **Instances** tab.
   The **Instances** pane appears.
3. On the right side of the **Instances** pane, next to the text box for the OT Connect Tag node for which you want to configure the limit values, select ✎ .

   Note: The ✎ icon is enabled for the field only if a record is assigned to the OT Connect Tag node.

   The **Edit Numeric Limits** or the **Edit Character Limits** window appears, based on the data type of the Source Tag, numeric or character, which is assigned to the OT Connect Tag node.
4. Enter the required limit values.

   Note: It is not mandatory to specify all the limit values. However, for numeric limits, the values entered must be in the following order:

   - The value assigned to Upper level 3 field must be greater than that assigned to Upper level 2 field.
   - The value assigned to Upper level 2 field must be greater than that assigned to Upper level 1 field.
   - The value assigned to Upper level 1 field must be greater than that assigned to Lower level 1 field.
   - The value assigned to Lower level 1 field must be greater than that assigned to Lower level 2 field.
   - The value assigned to Lower level 2 field must be greater than that assigned to Lower level 3 field.

   The following table provides sample values for the numeric limits:

Field	Value	Field	Value
Upper level 3	6	Lower level 1	3
Upper level 2	5	Lower level 2	2
Upper level 1	4	Lower level 3	1

5. Select **Save**

6. On the **Instances** pane, select 🖫.
   The policy instance is saved.

Activate or Deactivate a Single Policy Instance

Before You Begin

- Create a policy instance.

About This Task

You can activate or deactivate a single policy instance or all instances associated with a policy. This topic describes how to activate or deactivate a single policy instance.

Procedure

1. Access the policy that is associated with the policy instance that you want to activate or deactivate.
2. At the bottom of the **Design** workspace, select the **Instances** tab.
   The **Instances** pane appears.
3. On the left side of the **Instances** pane, select the policy instance that you want to activate or deactivate.
4. On the right side of the pane, above the grid, either the ⛭ button or the ⛭ button is displayed.

   - If the ⛭ button is displayed, the policy instance is active. To deactivate the policy instance, select this button.

   - If the ⛭ button is displayed, the policy instance is not active. To activate the policy instance, select this button.

   In the left pane, the corresponding active/inactive indicator changes to reflect your selection.

5. On the **Instances** pane, select 💾.
   The policy instance is saved.

Results

- If you activated the policy instance and if the policy is active, the instance will be executed according to the policy's execution settings.
- If you deactivated the policy instance, the instance will not be executed.

Activate or Deactivate All Policy Instances Associated with a Policy

Before You Begin

- Create a policy instance.

About This Task

You can activate or deactivate a single policy instance or all instances associated with a policy. This topic describes how to activate or deactivate all policy instances associated with a policy.

Procedure

1. Access the policy that is associated with the policy instances that you want to activate or deactivate.
2. At the bottom of the **Design** workspace, select the **Instances** tab.
   The **Instances** pane appears.
3. On the left side of the **Instances** pane, below the grid:

   - If you want to activate all policy instances, select the ⛭ button.

     Note: If a policy instance is not fully configured (indicated by the red ⛭), it will be excluded.

   - If you want to deactivate all policy instances, select the ⛭ button.

   In the left pane, the corresponding active/inactive indicators change to reflect your selection.

4. Select **Save**.
   The policy is saved.

Results

- If you activated all policy instances and the policy is active, all policy instances will be executed according to the policy's execution settings.
- If you deactivated all policy instances, no instances that are associated with the policy will be executed.

Export Policy Instances

About This Task

You can export the instances associated with a Policy to a Microsoft Excel (.xlsx) file.

Tip: You can modify the instance details in the exported file, and then import the same file using the Policy Instance Data Loader to update the instances in the APM database.

Procedure

1. Access the policy associated with the instances that you want to export.
2. In the **Design** workspace, select the **Instances** tab.
   The **Instances** section appears.
3. In the **Instances** section, select ▶ , and then select ⬆.
   All instances associated with the Policy are exported to a Microsoft Excel (.xslx) file and the file is downloaded to your local drive.

   Note: By default, the name of the file is in the `Policy_Instances_<PolicyName>.xlsx` format.

Duplicate a Policy Instance

Procedure

1. Access the policy that is associated with the policy instance that you want to delete.
2. At the bottom of the **Design** workspace, select the **Instances** tab.
   The **Instances** pane appears.
3. On the left side of the **Instances** pane, select the policy instance that you want to copy.
4. On the right side of the pane, select ⧉.
   A confirmation message appears.
5. Select **Yes**.
   The policy instance is copied from the **Instances** pane, and the Policy Instance record representing the policy instance is duplicated in the APM database.

Delete a Policy Instance

Procedure

1. Access the policy that is associated with the policy instance that you want to delete.
2. At the bottom of the **Design** workspace, select the **Instances** tab.
   The **Instances** pane appears.

3. On the left side of the **Instances** pane, select the policy instance that you want to delete.
4. On the right side of the pane, select 🗑.
   A confirmation message appears.
5. Select **Yes**.
   The policy instance is removed from the **Instances** pane, and the Policy Instance record representing the policy instance is deleted from the APM database.

Chapter

8

Policy Logic Validation

Topics:

About Validating the Policy Logic

After you build a policy model, you can run a validation process to ensure that the policy logic yields the results you want. Policy validation simulates policy execution, but no actions will be taken as a result (e.g., no Policy Recommendation records will be created). This prevents the policy from generating potentially invalid data while you are confirming that the policy logic is working as expected.

You can validate the policy model logic using ad hoc test values or values in the records belonging to a policy instance. After you run the validation process, detailed results of the simulated execution are displayed on the model canvas.

Validate Policy Logic Using Ad Hoc Test Values

Before You Begin

- Build a Policy Model.

About This Task

You can validate the policy model logic using ad hoc test values or values in the records belonging to a policy instance. This topic describes how to validate the policy logic with user-defined ad hoc test values.

Procedure

1. Access the policy that you want to validate.
2. In the **Design** workspace, select the **Validation** tab.
   The **Validation** pane appears with one section for each Input node in the policy model that requires an input value.

   Tip:
   - To resize the pane, drag the top edge of the pane.
   - To maximize the pane, select ⬈.
   - To restore the pane to its original size, select ⬊.
   - To close the pane, select ⌄.

3. Enter the test values that you want to use to validate the policy logic.

   For *collection* inputs, select **Add Collection Value** to display additional boxes in which you can specify collection values.

   Note:
   - Unspecified test values are considered as empty strings and not as null values for validation. To specify a null value as the test value, enter null.
   - You can copy values from a policy instance to use as a starting point. To do so, select the instance whose values you want to copy, and then select 🗐.

4. Select ▷.
   The validation process begins. When the validation is complete, the nodes in the policy model are color-coded to indicate the results of the validation.

5. In the policy model, select a node to view additional details on the execution of that node.

Validate Policy Logic Using Policy Instance Values

Before You Begin

- Create a policy instance.

About This Task

You can validate the policy model logic using ad hoc test values or values in the records belonging to a policy instance. This topic describes how to validate the policy logic with user-defined, ad hoc test values.

Procedure

1. Access the policy that you want to validate.
2. In the **Design** workspace, select the **Validation** tab.
   The **Validation** pane appears with one section for each Input node in the policy model that requires an input value.

   Tip:
   - To resize the pane, drag the top edge of the pane.
   - To maximize the pane, select ⬈.
   - To restore the pane to its original size, select ⬎.
   - To close the pane, select ⌄.
3. On the left side of the pane, select the policy instance whose values you want to use in the validation process.
   The right side of the pane displays values from the records that are included in that policy instance. For example, in the following image, you can see that the value A is stored in the Criticality Indicator field in the Equipment record that is included in the policy instance.
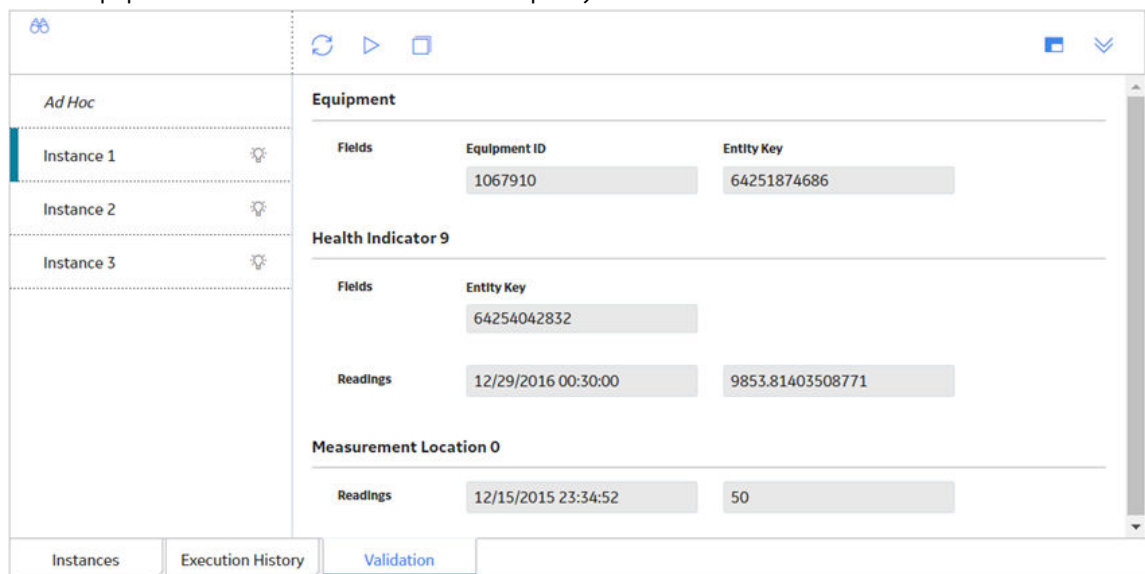
4. Select ▷.
   The validation process begins. When the validation is complete, the nodes in the policy model are color-coded to indicate the results of the validation.

5. In the policy model, select a node to view additional details on the execution of that node.

Chapter
9

Policy Execution

Topics:

About Policy Execution

When you execute a Policy, input values are evaluated against the logic in the Policy model. The input values for each execution are supplied by a specific Policy instance. Each time a Policy is executed, the results of the execution are recorded in the execution log, which you can view on the **Execution History pane**. You can use one or all the following methods to execute a Policy:

- Automatic execution
- Scheduled execution
- Manual execution

Important:

- You can execute only the active instances associated with active Policies.
- The policy execution log can grow quickly and significantly impact the size of the APM database. You can control how quickly the execution log grows by modifying the execution history settings for each policy.

Automatic Execution

When automatic execution is configured, individual Policy instances are executed when records belonging to the Policy instance are updated. Additionally:

- For records represented by a Measurement Location, OT Connect Tag, or Health Indicator node, Policy execution is also triggered by changes in related Reading records in the APM database. However, for Health Indicator nodes, if a reading with a timestamp earlier than the latest reading processed by the health indicator service is added or updated,the policy will not be triggered, because the Asset Health Indicator service does not update the health indicator record in this case.

  Note: When you configure a policy with a Health Indicator node for Automatic Execution, the Asset Health Indicator service must be running in order for new readings to trigger a policy execution.
- For process historian tags represented by an OT Connect Tag node, Policy execution is also triggered when new readings are added in the process historian.

  Note: When you configure a policy with an OT Connect Tag node for Automatic Execution, the OT Connect Adapter service must be running in order for new readings to trigger a policy execution.

Automatic execution works in conjunction with the selection in the **Trigger** check box that appears on the **Properties** window for all Input nodes except Query, Constant, and Point Value nodes. When the **Trigger** check box is:

- Selected: Changes in a record or related readings represented by the node will trigger Policy execution. The **Trigger** check box is selected by default.
- Cleared: Changes in a record or related readings represented by the node will not trigger Policy execution.

Important: You must ensure that the **Trigger** check box is selected for only the inputs where you want changes in the record or related readings, alerts or events to trigger Policy execution.

For example, you have a Policy that contains an Entity node that represents an Equipment record, but the Entity node does not influence any of the logic in the Policy model that determines if action is needed. Rather, other Entity nodes (for example, representing Measurement Location records that are linked to the Equipment record) influence the actual logic in the Policy model. In this case, you can exclude the associated Equipment record from triggering the Policy execution by clearing the **Trigger** check box for that node. By doing this, only the changes in the relevant records will trigger Policy execution.

When Should Automatic Execution Be Used?

Use automatic execution when the Policy is designed to monitor one or more values that change with time and when an action is needed in response to a specific change. Consider the following examples:

- The Policy monitors readings related to an OT Connect tag. When a reading value crosses a defined threshold, the Policy creates or closes a Policy event.
- The Policy monitors the status of a Health Indicator record. When the Health Indicator enters the Alert state, the Policy sends an email to the responsible user.

Scheduled Execution

When scheduled execution is configured, all active instances that are associated with a Policy are executed according to the schedule that you define.

When Should Scheduled Execution Be Used?

Use scheduled execution when the Policy is designed to evaluate data over a period or when automatic execution could produce misleading results.

Consider the following examples of Policies designed to evaluate data over a period:

- Using the Threshold Statistics node, the Policy analyzes threshold excursions during the previous month for values related to an OT Connect tag or Health Indicator.
- The Policy evaluates conformance to an asset strategy by comparing the actual count of readings added for a measurement location within the previous week to the expected count.

Consider the following example of a Policy monitoring values where automatic execution could produce misleading results:

The Policy evaluates combined information from a Rounds Route that includes multiple measurement locations for a single asset.

Because mobile users can enter readings in any order and could change a previously entered reading value before finalizing the Route, using automatic execution in this scenario could trigger executions where one or more readings are from an earlier Route execution or are otherwise invalid.

Instead of using automatic execution, the Policy could be scheduled to run at a similar frequency to the Route. For example, if the Route is completed during the morning shift on weekdays, the Policy could be scheduled to execute after the end of the shift. The Policy checks whether new Readings have been added for all the required inputs before proceeding with the evaluation of the results.

Considerations for Scheduled Policies

You can take steps to prevent delays in Policy execution by considering the number of Policies you are scheduling and the number of instances associated with each.

When scheduling multiple Policies, you can configure the Policy for different schedules so that the executions are staggered throughout the day.

All active instances of a scheduled Policy are submitted for execution at the same time. Therefore, when scheduling a Policy with a large number of instances, you can stagger the executions by creating multiple copies of the Policy, each with a subset of the instances. You can then schedule each of the Policies to be executed at a different time. This approach may be especially advantageous when users are located in multiple time zones, as you could configure relevant Policy instances to be executed outside of normal business hours such that results are available in different locations as needed.

Manual Execution

If a Policy is active, you can manually execute the Policy irrespective of the configured execution settings. You can execute an active instance or all the active instances associated with the Policy. If you manually execute a Policy, the execution settings configured for the Policy are not changed and the Policy continues to be executed according to the configured settings.

When Should Manual Execution Be Used?

If you want the Policy to trigger the specified actions once without changing the existing execution settings that are configured for the Policy or configuring additional one-time scheduled execution settings, you can use the manual execution method.

The following examples require the policy to be executed to fully validate the results:

- Sending an email
- Applying an Asset Strategy Template

For example, you designed a new Policy for an electric motor such that if the working temperature of a motor exceeds 800 K, an email notification will be triggered to the related engineer. You created two instances for the Policy and activated them without activating the Policy. Based on the requirement of the Policy, you configured it for automatic execution. Now, if you want to verify if the newly created Policy correctly triggers the required email notification, you must additionally configure the Policy for a one-time scheduled execution and activate the Policy. In this scenario, you can manually execute the Policy and verify if it triggers the correct action, instead of configuring additional one-time execution settings.

About Active Policies and Policy Instances

When a policy is active, the active instances that are associated with the policy will be executed according to the execution settings defined for the policy.

Consider a policy with two instances, where one instance is active and another instance is inactive.

- If the policy is active, the active instance will be executed according to the execution settings defined for the policy.
- If the policy is inactive, neither of the policy instances will be executed. However, the active instances associated with an inactive policy can be executed if the policy is manually executed and is not according to the pre-configured execution settings.

If a policy is active but does not contain an active instance, a warning message appears in the notification bar, indicating that there are no active instances to execute.

Configure Scheduled Execution

Before You Begin

- Run the validation process to confirm that the policy logic is working correctly.
- Verify that the correct time zone is specified for the policy.

About This Task

You can configure a policy to be executed automatically or executed according to a predefined schedule (or both). This topic describes how to configure a policy to be executed according to a predefined schedule.

Procedure

1. Access the policy that you want to configure to be executed according to a defined schedule.
2. In the **Details** workspace, in the **Execution Settings** section, select the **Scheduled Execution** check box.

   The options to schedule an execution appear.
3. Select either **One time** or **Recurrence**.
4. In the **Start** box, specify the date and time at which you want the first scheduled execution to occur.
5. If you selected **Recurrence**, in the **Every** section, select the frequency at which you want the policy to be executed.

   Note:

   You can configure a policy to execute as frequently as every minute. However, due to performance impacts, policies should not be scheduled to execute at high frequencies in most circumstances. If you choose to execute a policy at a high frequency, consider the following guidelines to limit performance impact:

   - The policy should have a limited number of instances.
   - The policy should minimize use of Query or R Script nodes.
   - If the policy uses tag inputs (such as GE Tag or OT Connect Tag), the range of the data retrieved should be limited.
   - A Collection Filter node should be used after the OT Connect Tag node to limit the range of readings retrieved.
6. Select the **Disregard Daylight Saving Time** check box if you observe daylight saving and you need not change scheduling time from the standard time to daylight saving. The schedule will then remain at the same scheduled time.

7. On the toolbar, select 💾.
   The policy is saved. A summary of the schedule and the next date on which the policy will be executed appears below the schedule settings.

Results

When the policy is active, the active policy instances that are associated with the policy will be executed based on the defined schedule. However, if the schedule is configured to start in the past, the active policy instances that are associated with the policy will be executed immediately. Execution will then continue based on the defined schedule.

Next Steps

- Activate policy instances
- Activate the policy
- View Execution History

Configure Automatic Execution

Before You Begin

- Run the validation process to confirm that the policy logic is working correctly.

About This Task

You can configure a policy to be executed automatically or executed according to a predefined schedule (or both). This topic describes how to configure a policy to be executed automatically.

Procedure

1. Access the policy that you want to configure to be executed automatically.
2. In the **Details** workspace, in the **Execution Settings** section, select the **Automatic Execution** check box.
3. In the **Design** workspace, on the **Properties** window for each Input node that contains a **Trigger** check box, select or clear the **Trigger** check box as needed. You must select the **Trigger** check box for at least one Input node.
   When the Trigger check box is:

   - Selected, changes in a record or related readings represented by the node will trigger policy execution.
   - Cleared, changes in a record or related readings represented by the node will not trigger policy execution.

   Important: You should ensure that the **Trigger** check box is selected for only the inputs where you want changes in the record to trigger policy execution
4. Select **Save**.
   The policy is saved.

Results

When the policy is active, the active policy instances associated with the policy will be executed automatically when the specified records or related reading values are updated.

Next Steps

- Activate policy instances
- Activate the policy
- View Execution History

Execute an Active Instance Manually

Before You Begin

- Run the validation process to confirm that the policy logic is working correctly.

About This Task

You can manually execute an active instance or all the active instances associated with a Policy. This topic describes how to execute a specific active instance associated with a Policy.

Note: You can add links to the Hyperlink widget in dashboards and configure them such that on selecting a link in a dashboard, specific active instances of a Policy are executed. The URL of the instances to be executed on selecting a link must be specified in the following format: `;`rte=Policy/Execute/<PolicyName>/<InstanceName1>/<InstanceName2>/...../<InstanceNameN>.

Procedure

1. Access the policy associated with the instance that you want to execute.
2. In the **Design** workspace, select the **Instances** tab.
   The **Instances** section appears.
3. In the **Instances** section, select the specific instance that you want to execute.
4. Select .

Note: ⟫ is enabled only for the active instances of a Policy.

The instance is queued for execution.

Execute all Active Instances Manually

Before You Begin

- Run the validation process to confirm that the policy logic is working correctly.

About This Task

You can manually execute an active instance or all the active instances associated with a Policy. This topic describes how to execute all the active instances associated with a Policy at once.

Note: You can add links to the Hyperlink widget in dashboards and configure them such that on selecting a link in a dashboard, all active instances of a Policy are executed. The URL of the Policy to be executed on selecting a link must be specified in the following format: `;rte=Policy/Execute/<PolicyName>`.

Procedure

1. Access the policy that you want to execute.
2. In the **Details** workspace, select **Execute Now**.

   Note: The **Execute Now** button is enabled only when one of the instances associated with the Policy is active.

   The **Confirm Policy Execution** window appears.
3. Select **OK**.

   All active instances associated with the Policy are queued for execution.

Activate or Deactivate a Policy

Before You Begin

- Run the validation process in order to confirm that the policy logic is working correctly.
- Configure the execution settings for the policy.

Procedure

1. Access the policy that you want to activate or deactivate.
2. In the Details workspace, on the toolbar, either the 💡 button or the 💡 button is displayed.

   - If the 💡 button is displayed, the policy is active. To deactivate the policy, select this button.

   - If the 💡 button is displayed, the policy is not active. To activate the policy, select this button.

   On the button and on the Details tab, the active/inactive indicator changes to reflect your selection.
3. Select **Save**.
   The policy is saved.

Results

- If you activated the policy, the active policy instances that are associated with the policy will be executed according to the policy's execution settings.
- If you deactivated the policy, none of the policy instances that are associated with the policy will be executed.

Next Steps

- View Execution History

Configure Policy Execution History Log Setting

About This Task

You can configure the policy to determine when execution history log records will be created.

Important: Policy execution history records can significantly impact the size of the APM database. To minimize the impact, you can select either the **Errors Only** or **Summary Only** option.

Procedure

1. Access the policy for which you want to configure the execution history log setting.
2. In the **Details** workspace, in the **Execution History Log Setting** section, select one of the following options.

Option	Description
Normal	Creates an execution history record for every execution of the policy. This option is selected by default for new policies.
Errors Only	Creates an execution history record only for the executions of the policy that result in an error. This option can be used to reduce the number of execution history records being added to the APM database, thus reducing the load on the database server.
	Note:
	If you select the **Errors Only** option:
	- You can only review the execution results in the design canvas for policy executions that result in errors.
	- The executions that resulted in an error are shown in the COUNT OF EXECUTIONS PER POLICY FOR LAST 30 DAYS chart of the **Policy Designer Overview** page.
	- The most recent execution that resulted in an error is listed in the **Policies** section of the **Policy Designer Overview** page.
	This setting does not affect the policy execution server log files.
Summary Only	Creates an execution history record for every execution of the policy and saves only the summary of the execution in the record. You can use this option to reduce the file size of the execution history records being added to the APM database.
	Note: If you select **Summary Only** in the **Execution History Log Setting** section and the policy execution resulted in an error, the node execution details are also saved along with the summary.

For policies which include Sub Policy nodes, execution history logs are created as follows:

- If the **Execution History Log Setting** is **Normal** for a calling policy, and **Errors Only** for the sub policy, an execution history record is created for every execution of the calling policy. The Sub Policy node in the execution history records displays execution details only for the executions of the sub policy, which resulted in an error.
- If the **Execution History Log Setting** is **Errors Only** for a calling policy, and **Normal** for the sub policy, an execution history record is created only for the executions of the calling policy, which resulted in an error. The Sub Policy node In the execution history records displays the execution details for every execution of the sub policy, regardless of whether the sub policy execution resulted in an error.
- If the **Execution History Log Setting** is **Errors Only** for both calling policy and sub policy, an execution history record is created only for the executions of the calling policy, which resulted in an error. The Sub Policy node in the execution history record displays execution details only for the executions of the sub policy, which resulted in an error.
- If the **Execution History Log Setting** is **Summary Only** for a calling policy, and **Errors Only** for the sub policy, an execution history record that contains the summary of the execution is created for every execution of the calling policy. The Sub Policy node in the execution history records displays the node execution details also when the sub policy execution resulted in an error.

3. On the toolbar, select 🖫

   The policy is saved. The execution history log setting is applied to subsequent policy executions.

Access Execution History

Procedure

1. Access the policy for which you want to view the execution history.
2. In the **Design** workspace, select the **Execution History** tab.
   The **Execution History** pane appears. On the left side of the pane, a list of the policy's instances appears. On the right side of the pane, execution summaries for all policy instances appear.

   Tip:
   - To resize the pane, drag the top edge of the pane.
   - To maximize the pane, select ⬈.
   - To restore the pane to its original size, select ⬊.
   - To close the pane, select ⌄.
3. On the left side of the pane, select a policy instance.
   On the right side of the pane, a summary of the execution results for the selected instance appears.
4. To filter the summary of the execution results, select 🔍, and then enter the search criteria.

   Note:
   - The 🔍 button is enabled only if there are at least two search results.
   - The filter is reset if you place the cursor outside the result grid.
5. On the right side of the pane, select a specific execution to view additional details on the policy canvas.

   Note:
   - If changes have been made to the policy model since the selected execution occurred, you will not be able to view the details of that execution on the canvas.

- You can use the **Actions** and **Errors and Warnings** check boxes to display only the executions that resulted in actions or only the executions that resulted in errors and warnings.
- Execution history records are retained for the duration specified in the **Policy Admin** page.

6. To export the summary of execution results, on the right side of the pane, select **Export data**.
   The summary is exported into an Excel spreadsheet. If you have applied a filter, only the filtered data is exported.

   Note: To avoid performance issues, we recommend that you export less than 10,000 rows.

7. In the policy model, select a node to view additional details about the specific node's execution.

   Note: You can now check the Execution Duration for each node in a policy. If you select each node of a policy in the Validation mode or in the Execution history entry record, the Node Execution Details display the Execution duration.

Monitor Policy Execution

About This Task

When you execute a policy, a notification is sent to the policy trigger queue. A policy trigger service processes the requests and sends a message to the policy execution queue. A policy execution service processes this message and executes the policy. Depending on the load on the policy execution service, the start time of the policy execution varies.

The policy execution history contains the time that the notification was sent to the policy trigger queue, the execution start time, and the execution end time for each policy execution.

You can monitor the average and maximum wait times and review recent policy executions on the **Policy Designer Overview** page and the **Recent Policy Executions** dashboard.

Procedure

- To view policy execution details on the **Policy Designer Overview** page:

  1. Access the Policy Designer Overview page.
  2. Access the **AVERAGE AND MAX WAIT TIMES IN QUEUE FOR 15 MIN INTERVALS IN LAST 4 HOURS** Graph.

- To view policy execution details on the **Recent Policy Executions** dashboard:

  1. In the **Applications** menu, navigate to **TOOLS** > **General Dashboards**.
  2. Select Browse.
  3. Navigate to the `Public\Meridium\Modules\Policy Manager` folder.
  4. Follow one of the steps mentioned below:

     If APM uses a SQL Server database, select the Recent Policy Executions dashboard.

     -or-

     If APM uses an Oracle schema, select the Recent Policy Executions(Oracle) dashboard.

     The dashboard opens in a new tab and the page filter window is displayed.
  5. Select the Policies for which you want to see recent executions.
  6. Enter the duration (in minutes) for which you want to see recent executions.
  7. Select Done.
     The **Recent Policy Executions** dashboard appears, displaying these results:

     ◦ A summary of execution results for the selected policies and review period.
     ◦ The progress of most recent scheduled policy execution request.

- The **AVERAGE AND MAX WAIT TIMES IN QUEUE FOR 15 MIN INTERVALS IN LAST 4 HOURS** Graph.
- A list of records that are policy triggers that have been edited in the selected review period, and for which no policy execution history exists. This could indicate either that the policy execution message is still in the queue, or, if enough time has elapsed based on the current maximum queue wait time, that the policy failed to trigger.

Note: This dashboard may exclude results for policies where the execution log setting is Errors Only.

About Execution History Logs

Execution history logs are stored in a non-family table, MI_POLICY_EXEC_LOG, in the APM database. You can access these records using a query by typing the code directly into the SQL tab of the query designer. This table contains logs for both Policy Designer and Family Policy executions.

Execution History Log Fields

The following table describes the fields in the MI_POLICY_EXEC_LOG table.

Table 1:

Field ID	Data Type	Description
MI_POLICY_EXEC_LOG.PLOG_KEY	String	Unique key of the execution history log record
MI_POLICY_EXEC_LOG.PLOG_KEY	String	Unique key of the execution history log record.
MI_POLICY_EXEC_LOG.POLICY_KEY	String	Entity Key of the related policy. Empty for family policy execution records.
MI_POLICY_EXEC_LOG.INST_KEY	String	Entity Key of the related policy instance. Empty for family policy execution records.
MI_POLICY_EXEC_LOG.PLOG_START_TM	Date and Time	Start of execution.
MI_POLICY_EXEC_LOG.PLOG_TRIG_TM	Date and Time	Start of processing by the policy trigger service. Empty for family policy execution records.
MI_POLICY_EXEC_LOG.PLOG_END_TM	Date and Time	End of execution.
MI_POLICY_EXEC_LOG.PLOG_ACTION_TAKEN_FLG	Boolean	True if an action node was executed.
MI_POLICY_EXEC_LOG.PLOG_ERRORS_FLG	Boolean	True if an error occurred.
MI_POLICY_EXEC_LOG.PLOG_WARNINGS_FLG	Boolean	True if a warning occurred.
MI_POLICY_EXEC_LOG.PLOG_SUMMARY_MEM	Text	Summary of the policy execution results. Displayed in the Execution History tab in Policy Designer or Family Policy.

Field ID	Data Type	Description
MI_POLICY_EXEC_LOG.PLOG_REV_NBR	Integer	Revision number of the policy model when the policy was executed. If this value is equal to the current policy model revision, the execution history details can be displayed in the design canvas.
MI_POLICY_EXEC_LOG.PLOG_DATA_TX	Text	Detailed policy execution results in JSON format.
MI_POLICY_EXEC_LOG.FAMPOLICY_KEY	String	Entity key of the related family policy. Empty for Policy Designer execution records.
MI_POLICY_EXEC_LOG.TRIGGERED_BY_CHR	String	Information about how the policy execution was triggered.

Example Execution History Log Query

The query shown below retrieves policy designer execution history log records where an Add Value to Health Indicator node was executed, with execution start time between March 1 and March 31, 2023:

```
SELECT [MI_POLICY].[MI_POLICY_ID_C] "Policy Name"
, MI_POLICY_EXEC_LOG.POLICY_KEY "Policy Key"
, MI_POLICY_EXEC_LOG.PLOG_START_TM "Start Time"
, MI_POLICY_EXEC_LOG.PLOG_SUMMARY_MEM "Summary"
FROM MI_POLICY_EXEC_LOG
 JOIN [MI_POLICY] ON MI_POLICY_EXEC_LOG.POLICY_KEY =
[MI_POLICY].ENTY_KEY
WHERE (MI_POLICY_EXEC_LOG.PLOG_START_TM >= '2023-03-01' AND
MI_POLICY_EXEC_LOG.PLOG_START_TM < '2020-04-01' AND
MI_POLICY_EXEC_LOG.PLOG_SUMMARY_MEM LIKE '%Add Value to HI%')
```

Chapter

10

Admin

Topics:

Access the Policy Admin Page

About This Task

You can use the **Policy Admin** page to configure the retention settings for the execution history records of policies created using Policy Designer.

Important: You can access the **Policy Admin** page only if you are a member of the MI Policy Administrator security group.

Procedure

In the **Applications** menu, navigate to **ADMIN** > **Application Settings** > **Policy Designer**. The **Policy Admin** page appears, displaying the **Execution History Settings** workspace.

Configure Execution History Retention Settings

About This Task

The Policy execution log can grow quickly and significantly impact the size of the APM database. You can control the size of the execution log by minimizing the time that the execution history is retained in the APM system and also by selecting an appropriate execution history setting for each policy.

By default, the Policy Execution History records are never deleted. This topic describes how to set a retention period and the time interval of an automated job that deletes these records after the retention period is over.

The following conditions apply when you set the retention period:

- These settings are applicable to all types of Policies and Policy Execution History records: Errors, Warnings (Action Taken), Warnings (No Action Taken), Success (Action Taken), and Success (No Action Taken).
- The most recent execution history of each instance associated with a Policy is retained even if it exceeds the specified retention duration.
- Depending on the number of old Policy Execution History records that exist when the retention settings are configured, it may take multiple iterations of the automated job to delete all the old execution history records.

Procedure

1. Access the **Policy Admin** page.
2. For each execution result type for which you want to change the retention period, perform the following steps:
   a) In the **Execution History Settings** workspace, in the **Retention Period** section, select **Duration**. The **Duration** and **Every** fields appear.
   b) In the **Duration** box, enter the duration in months for which you want to retain the records.
3. In the **Schedule for background cleanup job** box, enter the following detail:
   - REPEAT INTERVAL: The repeat interval at which the automated job must run to delete old Policy Execution History records.
   - NEXT OCCURRENCE: The next occurrence date for the selected Repeat Interval.
   - START DATE: The start date to enable the job.

- END REPEAT: The end date for the job.
- TIMEZONE: The timezone that will be considered for running the job.

By default, the time interval that you enter is defined in hours. However, you can select the required unit from the radio button to specify the interval in other units of time in REPEAT INTERVAL.

4. Select **Save**.
   The execution history settings are configured.

About API Node Credential Records

API Node Credential records are used to store the user names, authentication types, and display names for user accounts that will be used by the system when executing API nodes in Policy Designer.

Note: Authentication type is set to APM by default.

When you create or update an API Node Credential, you must enter the user's current password. The password not stored; it is only used to authorize permission to use that user account as an API Node Credential. If the API user's password is changed you do not need to re-enter it through the Policy Admin page.

The user accounts that you specify in API Node Credential records must have the appropriate permissions to execute the APIs that you want to use in the API node. For information on supported APIs, see the API Node documentation.

Each API Node Credential must contain a unique value for display name. This is the name that will be displayed to Policy Designer users and should be descriptive to enable them to identify the correct user for the API they want to use. Policy Designer users will only see the display name and authentication type; they will not have access to the username or password.

You can set up any number of API Node Credentials as required. For example, you might want to set up different API Node Credentials for access to different APIs.

Add API Node Credential Records

Procedure

1. Access the Policy Admin Page on page 67
2. Select the **API Node Credentials** section.
3. Select ➕
   The **API Node User Details** window appears.
4. Enter the Display Name, Username and Password.
5. Select **Save**.

Results

The credential is saved and displayed in the list of available credentials in the API Node the next time you open a new Policy Designer page.

Update API Node Credential Records

Procedure

1.
2. Select the **API Node Credentials** section.
3. In the grid, select the credential that you want to edit and then select ✎
   The API Node User Details window appears.
4. Update the credential information as required.
5. Re-enter the Password.
6. Select **Save**.

Results

The credential is updated.

Delete API Node Credential Records

Procedure

1.
2. Select the **API Node Credentials** section.
3. In the grid, select the credential that you want to delete and then select 🗑
   A Confirmation Dialog Box appears.
4. Select **OK**.

Results

The credential is deleted. Any policies containing API nodes configured to use the credential will no longer execute successfully.

Chapter

11

Data Loader

Topics:

About the Policy Instance Data Loader

Using the Policy Instance Data Loader, you can add or update large numbers of policy instances for a policy in APM.

To import data using the Policy Instance Data Loader, APM provides an Excel template, `Policy Instance.xlsx`.

Note: In this documentation, the Excel template is referred to as the data loader workbook.

About the Policy Instance Data Loader Requirements

Before you use the Policy Instance Data Loader, ensure that the Policy Name and the node details specified in the data loader workbook exist in the Policy Designer module in APM.

Security Settings

The user who loads data must be associated with the following Security Groups and Security Roles:

- MI Data Loader User Security Role or MI Data Loader Admin Security Role
- MI Policy Administrator, MI Policy Designer, or MI Policy User Security Group; or a Security Role that is associated with the MI Policy Administrator, MI Policy Designer, or MI Policy User Security Group

About the Policy Instance Data Loader Data Model

A Policy Instance record has a foreign-key relationship to the Policy record that defines the instance.

The following data model illustrates the Policy Instance Data Loader:
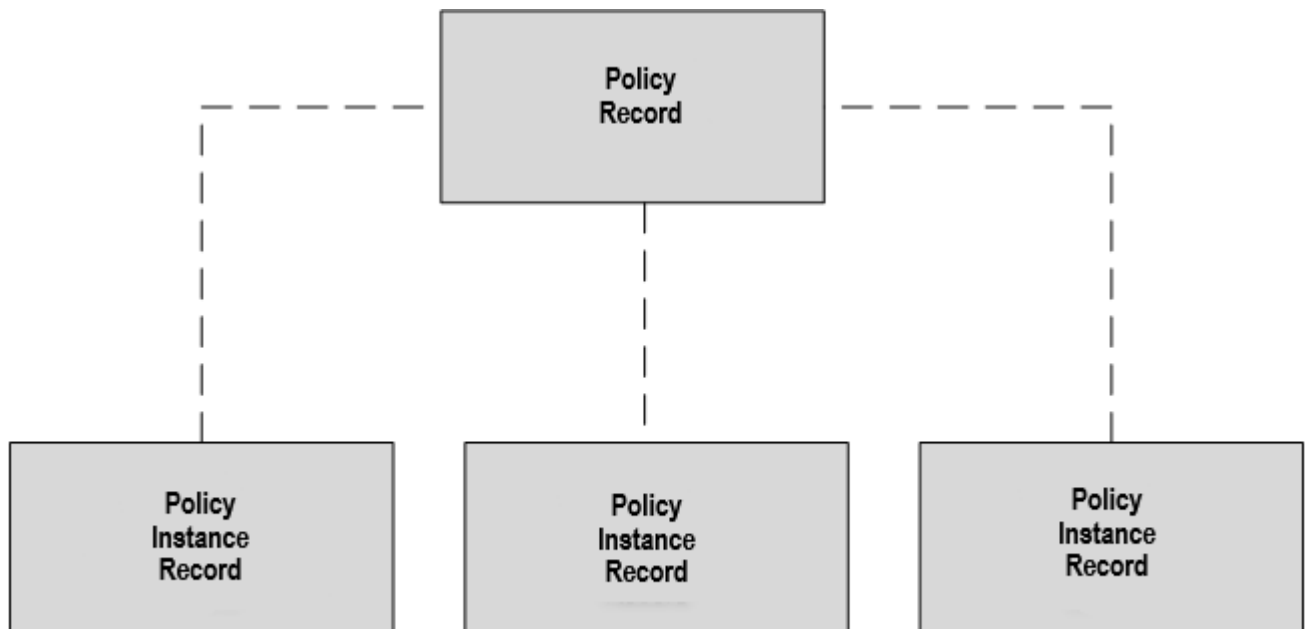
About the Policy Instance Data Loader General Loading Strategy

This section describes the prerequisites for loading the data and the order in which the data will be loaded.

Prerequisites

- The policy for which you want to import the instances must already exist in the APM database.
- Identify the Input nodes contained in the policy or policies for which you want to create instances. The Input nodes determine the columns required in the MI_POLICY_INST sheet of the data loader workbook.
- All input records that will be used in the policy instances must already exist in the APM database.
- Extract the data required to populate the data loader workbook from APM. You can do this by querying the database to retrieve the entity keys or entity IDs to use as inputs for the policy instances. You can then export the results of the query to an Excel spreadsheet to update the Policy Instance Data Loader workbook.

Important: In a data loader workbook, you must define the instances for only one policy. If you define instances for multiple policies in the same workbook, the data loader will not be imported.

Load Sequence

1. Download the Policy Instance Data Loader workbook provided by APM.
2. Modify the worksheet with the data extracted from APM.
3. Load the data loader workbook.
4. Monitor the status of the data load operation.
5. Access the data loader log to check for errors and warnings.
6. Conduct tests in APM to make sure that the imported data is correctly loaded.

About the Policy Instance Data Loader Workbook Layout and Use

To load data using the Policy Instance Data Loader, APM provides an Excel workbook, `Policy Instance.xlsx`, which supports baseline policy instances in APM.

Tip: You can export the data loader for a specific policy, including the data for current instances.

The following table lists the worksheets that are included in the Policy Instance Data Loader workbook:

Worksheet	Description
Configuration	This worksheet is used to specify whether policy instances will be updated.
MI_POLICY_INST	This worksheet is used to specify policy instances per policy.

Configuration Worksheet

In the Configuration worksheet, you must specify the settings for the import.

Field Caption	Field ID	Data Type	Comments
Batch ID	BATCH_ID	Numeric	Enter a serial number (for example, 1). **Note:** Records with different Batch ID values are processed as separate batches by the data loader.
Replace Existing Instance?	OPTION_REPLACE_INSTANCE	Character	Enter one of the following values: • FALSE: The existing policy instances that may be specified in the MI_POLICY_INST worksheet will not be updated. • TRUE: The existing policy instances that may be specified in the MI_POLICY_INST worksheet will be updated. **Note:** If you do not enter a value, the value FALSE will be considered and existing policy instances will not be updated.

MI_POLICY_INST Worksheet

In the MI_POLICY_INST worksheet, you must specify the existing or new instances to be imported for a policy. You must use the entity key or the entity ID of a record to map it to a node. By default, the MI_POLICY_INST worksheet is configured to use the entity keys to identify the records. If you want to specify entity IDs instead of entity keys, you must replace the text 'ENTY_KEY' with 'ENTY_ID' in the field ID of the required columns of the worksheet, and then enter the relevant entity IDs in the subsequent rows. For example, if you want to specify entity IDs instead of entity keys for a column with the field ID n0|ENTY_KEY, you must change the field ID to n0|ENTY_ID before entering the values in the subsequent rows.

Important: A record specified in the worksheet is not identified under the following conditions:
- If the entity ID is associated with multiple records and there is no primary record specified for the instance.
- If the entity ID is associated with multiple records that are linked to the primary record specified for the instance.
- If the entity ID or entity key does not exist in the specified entity family.

Under these conditions, the information associated with the relevant record is not imported, though it will not affect the rest of the information in the workbook, and a warning message appears in the data loader log, indicating the record that could not be imported. However, if an entity ID is associated with multiple records, among which, only one record is linked to the primary record specified for the instance, the record that is linked to the primary record is identified for the entity ID.

Field Caption	Field ID	Data Type	Comments
Batch ID	BATCH_ID	Numeric	Enter a serial number (for example, 1). **Note:** Records with different Batch ID values are processed as separate batches by the data loader.
Policy Name	MI_POLICY_INST_POLICY_GUID	Text	Enter the name of an existing policy.

Field Caption	Field ID	Data Type	Comments
Policy Instance ID	MI_POLICY_INST_ID_C	Text	Enter the name of a policy instance to associate with the specified policy.
			This value must be unique within a policy.
			Note: If the worksheet contains duplicate Policy Instance IDs within a policy, only the last row associated with the Policy Instance ID is imported, and a warning message appears in the data loader log, indicating the instances that could not be imported due to the duplicate IDs.
Active?	MI_POLICY_INST_ENABLED_F	Character	Enter one of the following values:
			• FALSE: Deactivates the policy instance.
			• TRUE: Activates the policy instance only if all the required information is available. Otherwise, it will remain inactive.
			Note: If you do not enter a value, the value FALSE will be considered for a new policy instance, whereas for an existing policy instance, its status will be retained.

Field Caption	Field ID	Data Type	Comments
Primary Record (ignored if there is a Primary Node)	FAMILY_ID|ENTY_KEY	Text	Enter the family ID and entity key of the record that you want to specify as the primary record in the policy (for example, MI_EQUIP000|64262245939). **Note:** If the policy contains a primary node, this field will be ignored.

<Input node name>	One of the following:
	
	 • <Input node ID>|ENTY_KEY
	 • <Input node ID>| FAMILY_ID| ENTY_KEY|FIELD_ID
	 • <Input node ID>|CONSTANT
		Text	For each Input node that you want to specify, you must create a new <Input node name> column.
			
			 Specify the name of the Input node (for example, Oil Level HI) as the field caption.
			
			 Enter the ID of the Input node and one of the following values as the field description:
			
			 • ENTY_KEY: Used for an Input node other than the Point Value node (for example, n0|ENTY_KEY). This value indicates that the subsequent rows contain the entity keys of the records that you want to map to this node.
			
			     Note: For a User node, this value indicates that the subsequent rows contain the entity keys of the Security User records that you want to map to the node.
			 • FAMILY_ID|ENTY_KEY|FIELD_ID: Used for a Point Value node that represents a value from a field (for example, n0| FAMILY_ID|ENTY_KEY|FIELD_ID). This value indicates that the subsequent rows contain sets of family ID, entity key, and field ID of the records that you want to map to this node.
			 • CONSTANT: Used for a Point Value node that represents a constant value (for example, n0| CONSTANT). This value indicates that the subsequent rows contain the constant values that you want to use for this node.
			
			 In the subsequent rows:
			
			 • For <Input node ID>|ENTY_KEY, enter the entity key (for example, 64251832824).
			 • For <Input node ID>| FAMILY_ID|ENTY_KEY|FIELD_ID, enter the following values:
			     ◦ Family ID
			     ◦ Entity key
			     ◦ Field ID
			     (For example, MI_MEAS_LOC|64254041888| Checkpoint ID)
			 • For <Input node ID>|CONSTANT, enter the Constant value (for example, 5).

Example Policy Instance Data Loader Workbook

This topic provides a sample of the worksheets in the Policy Instance Data Loader workbook to illustrate the process of adding and updating policy instances.

The following Configuration worksheet is populated to update the policy instance that is specified in the MI_POLICY_INST worksheet:

Replace Existing Instance?
OPTION_REPLACE_INSTANCE
TRUE

The following MI_POLICY_INST worksheet is populated to add and update policy instances for a policy named Equipment Lubrication Policy:

Batch ID	Policy Name	Policy Instance ID	Active?	Primary Record (ignored if there is a Primary Node)	Oil Level HI	Oil Check Measurement	Oil Level Limit
BATCH_ID	MI_POLICY_INST_POLICY_GUID	MI_POLICY_INST_ID_C	MI_POLICY_INST_ENABLED_F	FAMILY_ID|ENTY_KEY	n0|ENTY_KEY	n1|FAMILY_ID|ENTY_KEY|FIELD_ID	n2|CONSTANT
1	Equipment Lubrication Policy	Instance 1	TRUE	MI_EQUIP000|64262245939	64251832824		5
2	Equipment Lubrication Policy	Instance 2	FALSE		64251959674	MI_MEAS_LOC|64254041888|Checkpoint ID	

Instance 1 is an existing policy instance that you want to assign to the following Input nodes of Equipment Lubrication Policy:

- n0, an Input node other than the Point Value node whose entity key is 64251832824.
- n2, a Point Value node that represents a constant value and whose constant value is 5.

Instance 2 is a new policy instance that you want to assign to the following Input nodes of Equipment Lubrication Policy:

- n0, an Input node other than the Point Value node whose entity key is 64251959674.
- n2, a Point Value node that represents a value from a field and whose family ID is MI_MEAS_LOC, entity key is 64254041888, and field ID is Checkpoint ID.

When this Policy Instance data loader workbook is successfully imported into APM:

- Instance 1 will be updated in Equipment Lubrication Policy according to the respective node details defined in the worksheet.
- Instance 2, with the respective node details defined in the worksheet, will be added to Equipment Lubrication Policy; however, it will not be activated because the Active column had the value FALSE.

About the Policy Instance Data Loader Load Verification

After the data loader completes, this is how you verify the load was successful.

Procedure

- In the **Policy Designer Overview** page, in the **Policies** section, review the list of relevant policy or policies.

  These list shows the number of instances associated with each policy, allowing you to confirm that the expected number of instances were created.
- In Policy Designer, access the relevant policy or policies and run a sample of new or updated instances to confirm that the policy instances perform as you expect.

- After the policy is active, review the execution history regularly to identify any errors that may be related to incorrectly configured new or updated policy instances.

Chapter

12

Deployment and Upgrade

Topics:

- Deployment
- Upgrade

Deployment

Deployment

Refer to the deployment information here Deploy Policy Designer for the First Time.

Upgrade

Upgrade

Refer to the upgrade information here Upgrade Policy Designer.

Chapter

13

Reference

Topics:

General Reference

Policy Designer Data Model

The basic, underlying data structure of the Policy Designer module contains Policy and Policy Instance records.

- Policy records store identifying information about policies, such as the name, description, and execution dates. Policy records also store code that defines the logic represented by a *policy model*.
- Policy Instance records store identifying information about *policy instances* and store code that maps nodes in a policy model to values from specific APM records.
- Policy instance records are linked to Policy records by the Policy has Policy Instance relationship.

When you work with policies in Policy Designer, values in these records are updated automatically.

Important: Family Policies and VB Rules are not triggered when Policies or Policy Instances are deleted. Therefore, do not configure Family Policies or custom VB Rules to be triggered when the Policy has Policy Instance or the Policy Event Has Events relationship family records are deleted.

API Node Credential records store the user credentials that will be used when executing the API node in a policy. When you add or modify user credentials in the **Policy Admin** page, the values in these records are updated automatically.

Depending on the specific nodes used in a policy model, additional records may be created or modified when the policy is executed. Specifically:

- Records in the Policy Recommendation and Has Recommendations families may be created when you include a Create Recommendation node.
- Records in the Policy Event, Policy Instance has Policy Event and Has Events families may be created when you include a Create Event node and modified when you include a Close Event node.
- Records in families not listed above may be created, updated, or deleted when you include other action nodes in policies. For more information, refer to About Action Nodes in Policies on page 200.

Records in the Has Superseded Recommendation family may be created when you consolidate Policy Recommendations in Action Management.

The following image illustrates these families and their relationships.

81
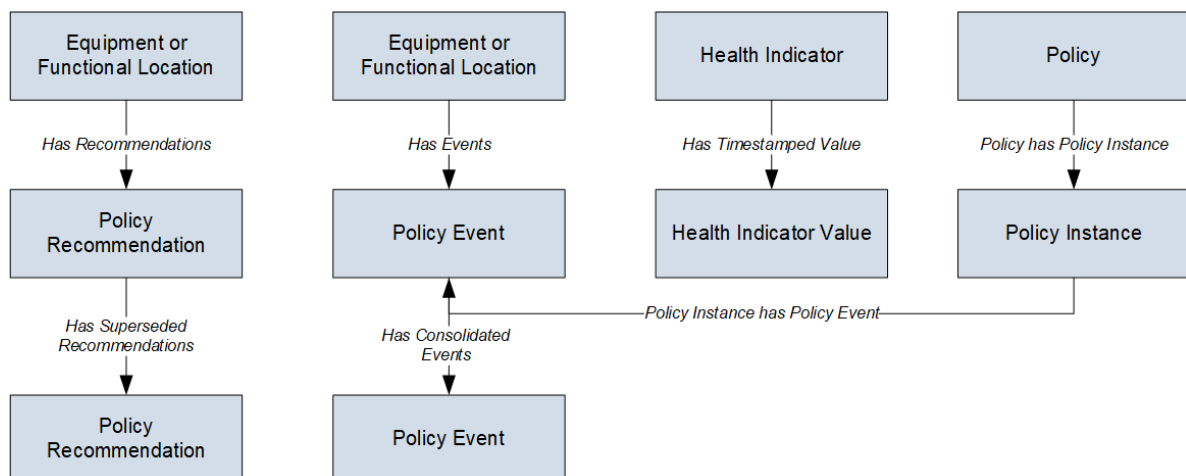
Policy Designer Security Groups

The following table lists the baseline Security Groups available for users within this module, as well as the baseline Roles to which those Security Groups are assigned.

Important: Assigning a Security User to a Role grants that user the privileges associated with all of the Security Groups that are assigned to that Role. To avoid granting a Security User unintended privileges, before assigning a Security User to a Role, be sure to review all of the privileges associated with the Security Groups assigned to that Role. Also, be aware that additional Roles, as well as Security Groups assigned to existing Roles, can be added via Security Manager.

Security Group	Roles
MI Policy Administrator	MI Health Admin MI Policy Administrator
MI Policy Designer	MI Health Power MI Health Admin MI Policy Designer MI Policy Administrator
MI Policy User	MI Health User MI Policy User
MI Policy Viewer	MI Policy Viewer

The baseline family-level privileges that exist for these Security Groups are summarized in the following table.

Family	MI Policy Administrator	MI Policy Designer	MI Policy User	MI Policy Viewer
Entity Families				
API Node Credential	View, Update, Insert, Delete	View	View	View
Health Indicator Value	View, Update, Insert, Delete	View, Update, Insert, Delete	None	View
Policy	View, Update, Insert, Delete	View, Update, Insert, Delete	View	View
Policy Event	View, Update, Insert, Delete	View, Update, Insert, Delete	View, Update	View
Policy Instance	View, Update, Insert, Delete	View, Update, Insert, Delete	View, Update, Insert, Delete	View
Policy Recommendation	View, Update, Insert, Delete	View, Update, Insert, Delete	View, Update	View
Relationship Families				

Family	MI Policy Administrator	MI Policy Designer	MI Policy User	MI Policy Viewer
Policy Has Policy Instance	View, Update, Insert, Delete	View, Update, Insert, Delete	View	View
Policy Instance Has Policy Event	View, Update, Insert, Delete	View, Update, Insert, Delete	View	View
Has Event	View, Update, Insert, Delete	View, Update, Insert, Delete	View, Update	View

Policy Designer URLs

There is one URL route associated with Policy Designer: `policy`. The following table describes the various paths that build on the route, and the values that you can specify for each element in the path.

Tip: For more information, refer to the URLs section of the documentation.

Element	Description	Accepted Value(s)	Notes
`policy`: Opens a new policy in the **Details** workspace.			
`policy/overview`: Opens the **Policy Designer Overview** page.			
`policy/<EntityKey>/<WorkspaceName>`: Opens a new or existing policy in the **Policy Designer** page.			
<EntityKey>	Specifies the policy that you want to open.	Any numeric Entity Key that corresponds to an existing policy.	Opens the specified policy in the **Details** workspace. This value is required to open an existing policy from a URL.
		0	Opens a new policy in the **Details** workspace.
<WorkspaceName>	Specifies the workspace in which you want to open the policy.	details	The specified policy will appear in the **Details** workspace.
		design	The specified policy will appear in the **Design** workspace.
`policy/instance/<InstanceEntityKey>`: Opens a policy instance in the **Instances** pane.			
<Instance EntityKey>	Specifies the policy instance that you want to open.	Any numeric Entity Key that corresponds to an existing policy instance.	The specified policy instance will appear in the **Instances** pane in the **Design** workspace for the policy that contains the instance.
`policy/Execute/<PolicyName>`: Executes all active instances associated with a policy.			
<PolicyName>	Specifies the name of the policy that you want to execute.	Name of a policy.	None.
`policy/Execute/<PolicyName>/<InstanceName1>/<InstanceName2>/...../<InstanceNameN>`: Executes specific instances associated with a policy.			

Element	Description	Accepted Value(s)	Notes
<PolicyName>	Specifies the name of the policy that you want to execute.	Name of a policy.	None.
<InstanceName>	Specifies the name of the policy instance that you want to execute.	Name of an instance associated with the specified policy.	You can specify multiple instance names associated with the specified policy.

Example URLs

Example URL	Destination
#policy -or- #policy/0/details	The **Details** workspace for a new policy.
#policy/64253414514 -or- #policy/64253414514/details	The **Details** workspace for the policy with the Entity Key 64253414514.
#policy/64253414514/design	The **Design** workspace for the policy with the Entity Key 64253414514.
#policy/instance/64253414515	The **Design** workspace for the policy that contains the policy instance with the Entity Key 64253414515. The specified policy instance is selected in the **Instances** pane.
#policy/Execute/Trigger Notification	A new tab displaying a message, indicating successful execution of all active instances associated with the Trigger Notification policy.
#policy/Execute/Trigger Notification/Instance for Notifying Technician/ Instance for Notifying Manager	A new tab displaying a message, indicating successful execution of the Instance for Notifying Technician and Instance for Notifying Manager instances associated with the Trigger Notification policy.

Policy Designer Site Filtering

The Policy and Policy Instances families are not enabled for site filtering. This means that any user who has the required security and license permissions can access and modify these records. However, as policies and policy instances interact with other records that are enabled for site filtering, it is important to understand what modifications a user may or may not make when working with policies that manipulate records across multiple sites.

Overview Page

On the **Policy Designer Overview** page, any user can view all policies and related information, such as execution results and policy instance information. In addition, any user may open any policy and modify the policy details, activate or deactivate the policy, or modify the policy model.

Instances

Any user may create new policy instances; however, users may only select records for the sites to which they have access when assigning records to a policy instance. When viewing existing policy instances, a user can modify (for example, save, activate, or deactivate) instances that contain records to which they do not have access (that is, records that were already associated with the policy instance by another user with access to different sites). However, users cannot validate an instance containing records to which they do not have access.

Validation

If a user attempts to validate an instance containing records to which they do not have access, the validation results will include errors indicating that the user does not have permission to view certain entities. However, if an instance indirectly references records to which users do not have access (for example, via the Query node), the instance can be validated and the validation results will respect site filtering based on the access of the signed in user.

So, for example, if a policy contains a Query node and a user validates an instance, the results of the query include only the records for the sites to which that user has access.

Ad Hoc validation operates in the same way, with the addition that values entered by the user in Ad Hoc validation are passed directly to validation. For example, assume that a policy uses the Entity Key of an asset record as the input parameter to a query to find information about the asset. The user enters a value in the **Entity Key** input box in Ad Hoc validation and runs validation. If the record with that entity key belongs to a site to which the user does not have access, validation will run and no validation error specific to site filtering will be generated, however, the Query node will return no results. This behavior is similar to the result if a user enters a value in the **Entity Key** input box which does not match any record in the database.

Execution

Policies are executed by a Super User and therefore has no site restrictions. Because of this, the actual results of a policy execution may differ from the validation results seen by a specific user.

To restrict the execution results based on Site, any queries used in a policy must be designed to return appropriate results. For example, you could specify an asset entity key as an input parameter to find related entities, which will generally be site filtered.

Execution Results

All users can view all execution history summary and details for all sites. However, if an execution result contains a hyperlink to a record for which a user does not have access, that user cannot access the record (that is, an error message will appear).

> Consider an organization that has three sites, Site X, Site Y, and Site Z. A policy is created with the following instances:
>
> • Instance 1: References a piece of equipment that is assigned to Site X.
> • Instance 2: References a piece of equipment that is assigned to Site Y.
> • Instance 3: References a piece of equipment that is assigned to Site Z.

Scenario	Description
Scenario 1: User assigned to only Site X	This user may create new policy instances, but he or she may only select records for Site X when assigning records to a policy instance. The user can modify and validate Instance 1. The user can modify Instance 2 and Instance 3, but cannot validate these instances. If an execution result contains a hyperlink to a record from Site Y or Site Z, the user cannot access the record (that is, an error message will appear).
Scenario 2: User assigned to both Site X and Site Y	This user may create new policy instances and select records for Site X and Site Y when assigning records to a policy instance. The user can modify and validate Instance 1 and Instance 2. The user can modify Instance 3, but cannot validate these instances. If an execution result contains a hyperlink to a record from Site Z, the user cannot access the record (that is, an error message will appear).
Scenario 3: Super User	This user may create new policy instances and select records for Site X, Site Y, or Site Z when assigning records to a policy instance. The user can modify and validate any instance.

About Policy Security and Ownership

Policy Designer Security Groups

The following table shows the Policy Designer activities that are accessible to members of the baseline Policy Designer Security Groups.

Activity	MI Policy Administrator	MI Policy Designer	MI Policy User	MI Policy Viewer
Configure policy execution history retention settings	✓	None	None	None
Create a new policy	✓	✓	None	None
Make changes to an existing policy, including the policy model, execution settings, and security settings	✓	✓	None	None
Save a copy of a policy	✓	✓	None	None
Delete a policy or policy instances	✓	✓	None	None
Revert a policy to baseline	✓	✓	None	None
Take ownership of a policy	✓	✓	None	None
Add or edit policy instances	✓	✓	✓	None
Validate a policy	✓	✓	✓	✓
View execution history	✓	✓	✓	✓

Note: Super Users receive the same privileges as members of the MI Policy Administrator group.

Individual Policy Security

If no specific security settings are configured for an individual policy, users can access or modify the policy based on the access level granted by their Policy Designer Security Group, as described in the previous section. However, if you are creating a policy to which only certain users should have certain levels of access, you can optionally configure more restrictive security settings for that individual policy.

Once you grant specific policy permissions to at least one user or Security Group, no other user has access to the policy regardless of their membership in the MI Policy Administrator, MI Policy Designer, MI Policy User, or MI Policy Viewer Security Group.

Note: Super Users continue to have Designer access to all policies unless the Super User is specifically given a lower level of permission in the individual policy security settings. In that case, the specified policy permission applies even to the Super User.

When you configure security for an individual policy, you can select specific users and/or Security Groups to grant Designer, User, or Viewer permissions for the individual policy. The following table shows the Policy Designer activities that are accessible for each type of permission.

Activity	Designer	User	Viewer
Create a new policy	✔	None	None
Make changes to an existing policy, including the policy model, execution settings, and security settings.	✔	None	None
Save a copy of a policy	✔	None	None
Delete a policy or policy instances	✔	None	None
Revert a policy to baseline	✔	None	None
Take ownership of a policy	✔	None	None
Add or edit policy instances	✔	✔	None
Validate a policy	✔	✔	✔
View execution history	✔	✔	✔

When you configure individual policy security, the most restrictive security level applies to the user, based on their membership in the MI Policy Administrator, MI Policy Designer, MI Policy User, or MI Policy Viewer Security Group and security specified for the individual policy. This concept is best explained through examples:

- Example 1: A member of the MI Policy Designer Security Group has been given Viewer permission to a policy. Because the individual policy security is the most restrictive, this user can only view the policy.
- Example 2: A member of the MI Policy Viewer Security Group has been given Designer permission to a policy. Because the Security Group access is most restrictive, this user can only view the policy.
- Example 3: A Super User has been given User permission to a policy. Because the individual policy security is the most restrictive, this user can only use the policy.

Policy Ownership

Each policy is owned by a user who is responsible for the policy and is granted exclusive privileges to modify the policy model and execution settings. The user who creates a policy is automatically assigned as the owner. If a different user wants to modify the policy, he or she must first take ownership of the policy.

In order to be a policy owner, a user must be a Super User or a member of the MI Policy Designer Security Group. If any individual policy security is applied to the policy, the user must also have Designer permission.

About The Notification Bar

The notification bar that appears at the top of the page in Policy Designer provides real-time validation feedback on the configuration of a policy. The following image shows an example of the notification bar.

Notification 1 of 6 ⟩ The Create Event node specifies no duration, so the Close Event node will never get executed.   Go to Source
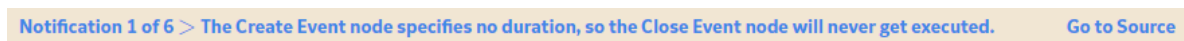
The notification bar appears either red or yellow, depending on the nature of the message:

- Red: The notification bar appears red for notifications that identify missing or invalid data in fields that are required for the policy to be fully configured. To avoid the execution of an invalid policy, you cannot save an active policy while red notifications appear. You must either complete the configuration of the policy or deactivate it prior to saving it.
- Yellow: The notification bar appears yellow to identify optional data that is missing or warnings about potentially incorrect configurations. While you can save policies with yellow notifications, you should confirm that the corresponding configurations are correct as you intended them.

Notifications appear at the top of the screen until the requirements described in the message are fulfilled. If multiple notifications are displayed at once, you can scroll through them using the **<** and **>** buttons.

Depending upon the notification, the bar may include a **View Details** link, which you can select for more information, and a **Go To Source** link, which you can select to go to the source of the notification.

About Execution Result Summaries

Each time a policy instance is executed, the execution results are recorded in the execution log. You can view a summary of each execution in the **Execution History** pane. Summaries include items such as warning messages, errors, returned values, and actions. For example, you might see the following summary information:

No Action Taken

The policy was executed but no actions were triggered. For example, this might occur if the policy's input values did not meet the conditions defined in the policy logic.

START TIME	END TIME	SUMMARY
01/06/2020 18:10:23	01/06/2020 18:10:23	No action taken.

<Action Taken>

The policy was executed and resulted in actions. The actions are listed in the Summary column of the grid.

START TIME	END TIME	SUMMARY
01/06/2020 18:09:26	01/06/2020 18:09:26	Email sent to test@ge.com. Entity record (Equipment ~ ~ Equipment Test Record-64263064232) is updated. Created value on health indicator. [Value: 70000]

Errors Occurred

There was an error in the policy logic, and no action was taken. For example, this might occur if a node's **Properties** window did not contain a value in a required field.

START TIME	END TIME	SUMMARY
01/06/2020 18:10:57	01/06/2020 18:10:57	Errors occurred. No action taken.

When you select an execution summary in the **Execution History** pane, detailed execution results are displayed on the model canvas. However, if changes have been made to the policy model since the selected execution occurred, you will not be able to view that details of that execution on the canvas.

Note:

- If the Execution History Log Setting selected for a policy is Errors Only, you will see only the executions that resulted in an error.
- If the policy model has been updated since the policy execution occurred, you may be unable to access the detailed execution results.

About Validation and Execution Details

After you validate or execute a policy, detailed results of the validation or execution can be viewed on the model canvas. Validation details appear on the model canvas automatically when you run the validation process. Execution details appear when you select a past execution on the **Execution History** pane.

Node Color-Coding

To indicate the results of an execution, the nodes in the policy model are color-coded as follows:

Green

Indicates that the node was executed successfully (i.e., the node was configured correctly, the source value was valid, and the execution produced a valid result).

Pale yellow

Indicates that the node was executed successfully, but raised a warning that the policy designer should review and address if needed.

Red

Indicates that the node was not executed successfully due to an error. When a node's execution encounters an error, subsequent nodes in the model will not be executed.

Gray

Indicates that the node was not executed. This may be expected (i.e., due to the policy logic) or unexpected (i.e., due to errors in the execution of preceding nodes).

The following example diagram shows nodes in each of these states.
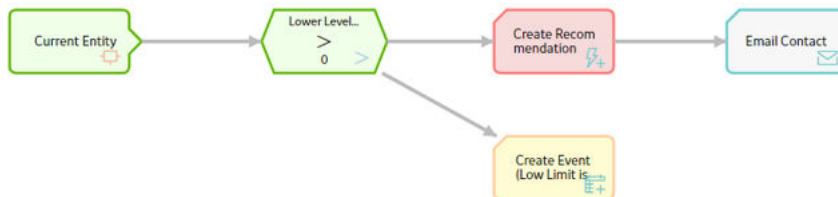
Node Execution Details Window

After you validate or execute a policy, you can select a node in the policy model to view the execution details of the node in the **Node Execution Details** window. If the node was successfully executed, the inputs and outputs (for example, certain values, logical results, or actions) of the node appear in the execution details of the node. If the execution of the node resulted in a warning or error, the window provides additional details about the cause of the warning or error. In addition to the execution details of the node, the **Node Execution Details** window for the Sub Policy node contains the **View Execution Details** link. You can use the link to view the model and execution details of the sub policy that is mapped to the node.

The following image is an example of the **Node Execution Details** window for a Condition node that was executed successfully with a logical result of Yes.

Node Execution Details

950

≥

900

Logical Result:

Yes
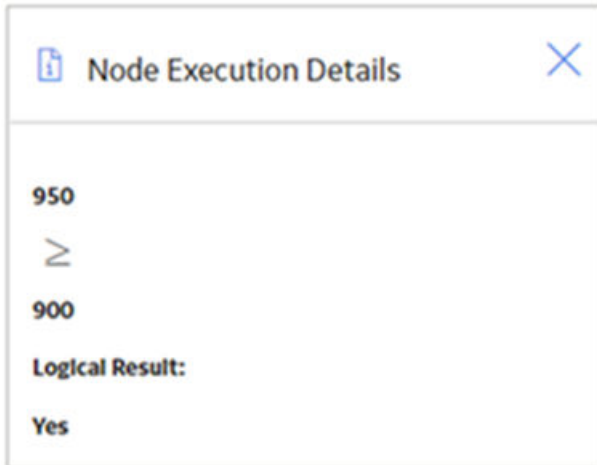
About Specifying Dates and Times to Evaluate in Policy Designer

In the **Properties** window for some nodes, you can specify a date and time that should be used when evaluating values. For example, consider the following **Properties** window for a Collection Filter node that is configured to filter Measurement Location reading values to only those that were recorded in the past 7 days.

n2 - Properties

Name
Within Last Week

Collection
Measurement Location 1
Readings

Column
Timestamp   >

Condition
now - 7d
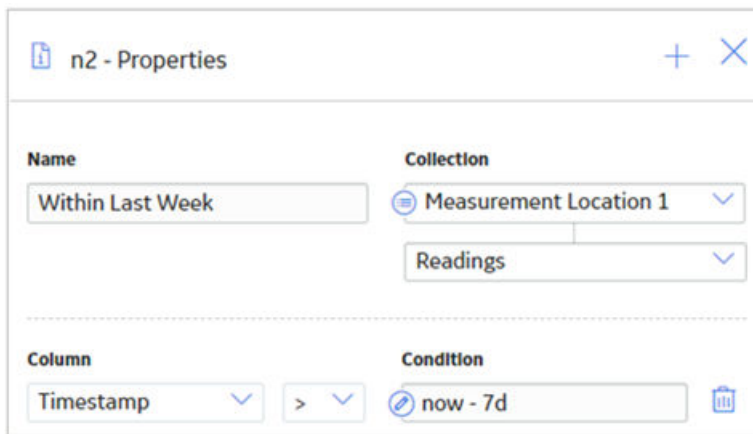
You can define specific date and time values in the standard format YYYY-MM-DD HH: mm: ss, where the time values use a 24-hour time format.

Alternatively, you can use a combination of acceptable values to specify dates and times that are relative to the time at which the policy is executed.

Specifying Relative Dates and Times

When using relative dates and times, the actual date and time that is evaluated depends on the date and time at which the policy is executed. For example, if you specify a relative date of Sunday, the APM system will use the most recent Sunday (at 12:00:00 A.M. in the policy's time zone) relative to the date and time that the policy is being executed. So, if a policy is executed on Friday, July 13, the evaluated date would be Sunday, July 8, as illustrated in the image below, where:

- The yellow star identifies the policy execution date.
- The red box indicates the evaluated date relative to the policy execution date.
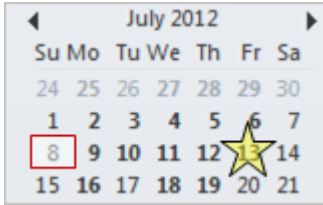
You can adjust relative date and time values by adding operators and variables to the relative date:

Operator

Specifies whether or not time should be added to or subtracted from the relative date and time. You can use the + (plus sign) or - (minus sign) operators.

Variable

Specifies the amount of time to add to or subtract from the relative date and time.

For example, if you specify Sunday + 4 hours, the evaluated date will be 4:00:00 A.M. on the most recent Sunday.

Acceptable Format for Relative Date and Time Values

The following table describes the constants that you can use to specify relative dates and times. Note that:

- Relative dates and times are determined in respect to the policy's time zone.
- These values are not localized, so you must enter them in English.

Constant	Meaning
Constants for Days	
*, start	The day and time that the policy execution began.
now	The day and time that the specific node is executed. For example, policy execution might begin at 1:00:00 A.M., but a subsequent node may not be executed until 1:00:25 A.M. If now is specified in the subsequent node, the APM system will use the time 1:00:25 A.M.
t, today	12:00 A.M. of the current day.
y, yesterday	12:00 A.M. of the previous day.
Sun, Sunday	12:00 A.M. of the most recent Sunday.
Mon, Monday	12:00 A.M. of the most recent Monday.
Tue, Tuesday	12:00 A.M. of the most recent Tuesday.
Wed, Wednesday	12:00 A.M. of the most recent Wednesday.
Thu, Thursday	12:00 A.M. of the most recent Thursday.
Fri, Friday	12:00 A.M. of the most recent Friday.
Sat, Saturday	12:00 A.M. of the most recent Saturday.
Constants for Months	

Constant	Meaning
Jan, January	12:00 A.M. of the most recent January 1st.
Feb, February	12:00 A.M. of the most recent February 1st.
Mar, March	12:00 A.M. of the most recent March 1st.
Apr, April	12:00 A.M. of the most recent April 1st.
May	12:00 A.M. of the most recent May 1st.
Jun, June	12:00 A.M. of the most recent June 1st.
Jul, July	12:00 A.M. of the most recent July 1st.
Aug, August	12:00 A.M. of the most recent August 1st.
Sep, September	12:00 A.M. of the most recent September 1st.
Oct, October	12:00 A.M. of the most recent October 1st.
Nov, November	12:00 A.M. of the most recent November 1st.
Dec, December	12:00 A.M. of the most recent December 1st.

Acceptable Format for Variables

The following table describes the variables you can use with relative dates. Note that these values are not localized, so you must enter them in English.

Variable	Description	Valid Number Type	Example
s, sec, second, seconds	Adjusts the time by the specified number of seconds.	Integer or decimal	7 sec
m, min, minute, minutes	Adjusts the time by the specified number of minutes.	Integer or decimal	10 minutes
h, hour, hours	Adjusts the time by the specified number of hours.	Integer or decimal	6.5 h
d, day, days	Adjusts the time by the specified number of days. A day is interpreted as 24 hours and does not account for Daylight Savings Time.	Integer	1 day
w, week, weeks	Adjusts the time by the specified number of weeks.	Integer	3 w
mo, month, months	Adjusts the time by the specified number of months.	Integer	6 months

Variable	Description	Valid Number Type	Example
y, year, years	Adjusts the time by the specified number of years.	Integer	2 years
[ddd].HH:MM:SS	Adjusts the time by the specified time period.	N/A	15.12:15:35

How the Policy Time Zone Affects Relative Dates and Times

When you specify a date and time that is relative to 12:00:00 A.M. of a specific day, the date range criteria and the policy time zone work together to determine the range of data that is included in the evaluation.

For example, suppose that you set up two policies to evaluate the exact same data, but one policy uses Eastern Time (UTC - 5) and the other uses Pacific Time (UTC - 8). If you specify that the policy should evaluate reading values starting at the relative time of today + 4 hours, the start time will be interpreted as 4:00:00 A.M., in the policy's time zone, of the current day. Because 4:00 A.M. Eastern Time occurs 3 hours earlier than 4:00 A.M. Pacific Time, each of those policies will end up evaluating a different set of data.

The following image illustrates this difference, where:

- The start and end time for the policy using Eastern Time is represented by the red shaded region.
- The start and end time for the policy using Pacific Time is represented by the green shaded region.
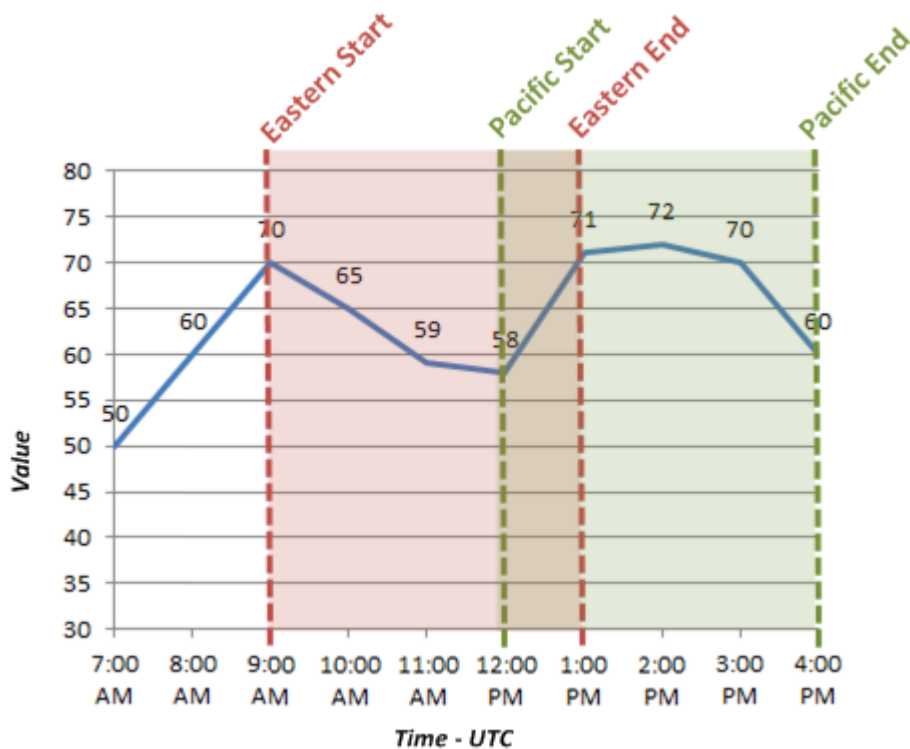
About Specifying Amounts of Time to Evaluate in Policy Designer

In the **Properties** window for some nodes, you can specify an amount of time (i.e., a time span) that should be used when evaluating values. For example, consider the following **Properties** window for a

Comparison node that is configured to determine whether or not the Accumulated Time output of a Threshold Statistics node is greater than 3 days.

Acceptable Format for Amounts of Time

The following table describes the values that you can use to specify an amount of time to evaluate. Note that:

- These values are not localized, so you must enter them in English.
- You can use negative numbers.
- You cannot specify a number of months or years because some months and years have a different number of days. Instead, you should specify a number of days (e.g., 30 days or 365 days).

Value	Description	Valid Number Type	Example
s, sec, second, seconds	Specifies a number of seconds.	Integer or decimal	15 seconds
m, min, minute, minutes	Specifies a number of minutes.	Integer or decimal	10.5 min
h, hour, hours	Specifies a number of hours.	Integer or decimal	1 hour
d, day, days	Specifies a number of days.A day is interpreted as 24 hours and does not account for Daylight Savings Time.	Integer	5 d
w, week, weeks	Specifies a number of weeks.	Integer	7 weeks
[ddd].HH:MM:SS	Specifies a number of days, hours, minutes, and seconds.	N/A	300.23:13:22

About Constants for Specific Values

To streamline the creation of calculations in policies, you can use constants to specify certain values. These constants can be specified in a Constant or Point Value node with an appropriate data type, and then used as an input to any other node in your policy.

The following table describes the supported constants.

Note: These values are not localized, so you must enter them in English.

Constant	Meaning	Data Type
Pi, pi	Pi, rounded to 14 decimal places (i.e., 3.14159265358979)	Decimal
E, e	Euler's number, rounded to 14 decimal places (i.e., 2.71828182845904)	Decimal
Null, null	Null (i.e., no value)	Any data type other than string.

About Units of Measure in Policies

During validation and execution of policies involving numeric values, Units of Measure (UOMs) are handled in the following ways for conversion and display of the values:

- If numeric values with different base UOMs are calculated or compared, the policy must be configured to apply the required UOM conversions before performing any action on the values or updating any of the values in other records.
- If a numeric value is required to be displayed in the execution summary of a policy or in a notification triggered by the policy, the value appears in the base UOM and not in the UOM Conversion Set associated with the user.
- Any value that you specify to be updated in a record through a policy is considered to be in the base UOM of the corresponding field.

> **Values with Different Base UOMs**
>
> Suppose that values from two temperature related fields with different UOMs are compared in a policy, which is configured to send an email notification if one temperature exceeds the other. The first temperature has a base unit of Kelvin and the second has a base unit of Fahrenheit. In this scenario, the policy must be configured to convert the first temperature value from Kelvin to Fahrenheit, and then make the comparison.

> **Precedence of Base UOM over User UOM Conversion Set**
>
> Suppose that the UOM associated with a temperature related field is Fahrenheit and the UOM Conversion Set associated with a user is configured to display all values of temperature related fields in Kelvin. Now, a policy is configured such that in a certain condition, the numeric value of the temperature related field is sent in an email notification to the user. In this scenario, the temperature value in the email notification appears in Fahrenheit even though the UOM Conversion Set associated with the user is configured to display temperature values in Kelvin.

Family Field Descriptions

API Node Credential Records

API Node Credential records store information about user credentials that can be used by the policy execution service when executing an API. The following table provides an alphabetical list and description of the fields that exist in the API Node Credential family. The information in the table reflects the baseline state and behavior of these fields.

This family is enabled for site filtering, which means that records in this family can be assigned to a specific site and will only be accessible to users who are assigned to the same site and have the appropriate license and family privileges. For more information, refer to the Sites section of the documentation.

By default, this family is configured to be excluded from global search and to use Rules.

Field	Data Type	Description	Behavior and Usage
Authentication Type	Character	The authentication method to be used when executing an API node.	
Display Name	Character	The display name of the user credential.	This value is displayed in the **API Node Settings** window in Policy Designer. A value is required and must be unique.
GUID	Character	The unvarying unique identifier for the record. It is used to identify the API user in the policy model.	This field is populated automatically.
User ID	Character	The user ID to be used when executing an API Node.	

Policy Records

Policy records store basic information about policies. The following table provides an alphabetical list and description of the fields that exist in the Policy family and that are displayed on the baseline Policy datasheet (unless otherwise noted). The information in the table reflects the baseline state and behavior of these fields. This list is not comprehensive.

This family is not enabled for site filtering, which means that records in this family can be accessed by any user with the appropriate license and family privileges. See the Site Filtering section of the documentation for more information.

By default, this family is configured to be included in global search and to use Rules.

Note: The values in Policy records cannot be modified in Record Manager.

Field	Data Type	Description	Field Behavior and Usage
Active	Boolean	Indicates whether the policy is active or inactive.	Reflects the selection of the activate/deactivate policy button in the **Details** workspace.
Automatic Evaluation	Boolean	Indicates whether the policy logic will be executed automatically when records belonging to the policy instance are modified.	Reflects the selection in the **Automatic Execution** check box in the **Details** workspace.

Field	Data Type	Description	Field Behavior and Usage
Description	Text	A brief summary of the policy.	Contains the value that you enter in the **Description** box on the **Details** workspace.
Execution History Setting	Character	Defines how the execution history is recorded for the policy.	Reflects the selection in the Execution History option in the **Details** workspace. The field is populated with the value Normal, by default.
Last Policy Model Change Date	Date and Time	Date and time when the policy model was last updated.	This field is populated automatically with the current date and time when you save the policy only if the policy model has been substantively modified.

If you modify the layout of the nodes on the canvas but do not make any other changes to the model, or you modify the name, description, schedule, or other settings of the policy, this field is not updated. |
| Model | Text | Code defining the logic that is represented by the policy model. | This field is populated automatically and does not appear on the Policy datasheet. |
| Module Workflow Policy | Character | Module Identifier | This field is used to determine whether the policy is a Module Workflow Policy for purposes of the **Policy Designer Overview** page.

The value in this field indicates the product module in which the policy is used. It does not appear on the Policy datasheet. |
| Last Scheduled Execution | Date and Time | Last scheduled execution date and time | If the policy is scheduled, this field contains the date and time at which the policy was last scheduled to execute. |
| Name | Character | The name of the policy. | Contains the value that you specify in the **Name** box in the **Details** workspace.

This field is required, and the value must be unique. |

Field	Data Type	Description	Field Behavior and Usage
Next Scheduled Execution		Next scheduled execution date and time	If the policy is scheduled, this field contains the date and time at which the policy is next scheduled to execute.
Owner	Character	The owner of the policy.	Contains the policy owner that appears at the top of the **Details** workspace. Only the owner of the policy can save changes to the policy. This field is required. This field is configured to track revision history by default. You can look at the revision history for this field to determine the ownership history for a particular policy.
Schedule	Character	Code defining the policy's schedule.	This field is populated automatically and does not appear on the Policy datasheet.
Schedule Executed	Boolean	Indicates whether the currently configured schedule has executed at least once.	This field is populated automatically and does not appear on the Policy datasheet.
Scheduled	Boolean	Indicates whether the policy has a defined execution schedule.	Populated automatically.

Field	Data Type	Description	Field Behavior and Usage
Time Zone	Character	The time zone associated with the policy.	Contains the value that you specify in the **Time Zone** list in the **Details** workspace. This field contains the value (UTC) Coordinated Universal Time by default.
Upgrade Log	Text	The log of upgrade steps that are applied to the policy model.	If your APM database has been upgraded from an earlier version to V5.0.0.0.0 or later versions, this field contains a record of the upgrade steps that were applied to the policy model to accommodate data model changes in V5.0.0.0.0.or a later release. This field does not appear on the Policy datasheet. Review the upgrade log information to identify any issues that need to be corrected manually once the upgrade is complete.

Policy Instance Records

Policy Instance records store information about policy instances that are associated with a policy. The following table provides an alphabetical list and description of the fields that exist in the Policy Instance family. The information in the table reflects the baseline state and behavior of these fields.

This family is not enabled for site filtering, which means that records in this family can be accessed by any user with the appropriate license and family privileges. See the Site Filtering section of the documentation for more information.

By default, this family is configured to be excluded from global search and to use Rules.

Note: The values in Policy Instance records cannot be modified in Record Manager. You must update the instance in the Policy Designer module in order to change values in the record.

Field	Data Type	Description	Behavior and Usage
Active	Logical	Indicates whether the instance is active or inactive.	Reflects the selection of the activate/deactivate policy instance button in the **Instances** pane.
ID	Character	The name of the instance.	Populated automatically with the value that you enter in the **Instance** box in the **Instances** pane.

Field	Data Type	Description	Behavior and Usage
Mapping	Text	Contains code that maps values from records belonging to the policy instance to the nodes in the associated policy model.	This field is disabled and does not appear in the Policy Instance datasheet.
Policy Key	Character	The entity key of the associated policy record.	This field is disabled and does not appear in the Policy Instance datasheet.

Policy Event Records

Policy Event records store information about events that are associated with Equipment or Functional Location records that are monitored by a policy. The following table provides an alphabetical list and description of the fields that exist in the Policy Event family and that are displayed on the baseline Policy Event datasheet (unless otherwise noted). The information in the table reflects the baseline state and behavior of these fields. This list is not comprehensive.

This family is enabled for site filtering, which means that records in this family can be assigned to a specific site and will then only be accessible to users who are assigned to the same site and have the appropriate license and family privileges. See the Site Filtering section of the documentation for more information on using this feature.

By default, this family is configured to be excluded from global search and to use Rules.

Field	Data Type	Description	Behavior and Usage
Close Description	Text	A description of why the event was closed.	Populated automatically based on the properties defined for the associated Create Event or Close Event node.
Description	Text	A description of the event.	Populated automatically based on the properties defined for the associated Create Event node.
End Time	Date	The date on which the event ended.	Populated automatically based on the properties defined for the associated Create Event or Close Event node. This field is used only when the Has Duration field is set to True.
Event Type	Character	The type of event.	Populated automatically based on the properties defined for the associated Create Event node.

Field	Data Type	Description	Behavior and Usage
Has Duration	Logical	Indicates whether or not there is an end time associated with the event.	Populated automatically based on the properties defined for the associated Create Event node.
Name	Character	The name of the event.	Populated automatically based on the properties defined for the associated Create Event node. In the **Execution History** pane, this value will appear as a hyperlink that you can select to access the **Events** section in Asset Health Manager.
Policy Instance Key	Character	The entity key of the associated policy instance record.	In the **Policy Event** datasheet, this field is labeled **Link to Policy** and contains a link to the policy that created the Policy Event record.
Severity	Character	The severity of the event.	Populated automatically based on the properties defined for the associated Create Event node.
Start Time	Date	The date on which the event started.	Populated automatically based on the properties defined for the associated Create Event node.
Time Line Reset	Logical	This field is not currently used.	N/A

Policy Recommendation Records

Policy Recommendation records store basic information about recommendations that have been created as a result of a policy. The following table provides an alphabetical list and description of some of the fields that exist in the Policy Recommendation family. The information in the table reflects the baseline state and behavior of these fields. This list is not comprehensive.

This family is enabled for site filtering, which means that records in this family can be assigned to a specific site and will only be accessible to users who are assigned to the same site and have the appropriate license and family privileges. For more information, refer to the Sites section of the documentation.

By default, this family is configured to be included in global search and to use Rules.

Note: Email notifications are not provided for Policy Recommendations that are overdue or require re-evaluation.

Field	Data Type	Description	Behavior and Usage
Associated Reference	Character	The Reference ID of the event or any other entity that originated the recommendation.	Populated automatically based on the properties defined for the associated Create Recommendation node.
Completed Date	Date	The date on which the recommended action was completed.	You can use the Calendar feature to select the date on which the recommended action was completed.
Completion Comments	Text	Details about the completed recommendation.	You can enter a value manually.
Create Work Request?	Boolean	Specifies whether a work request for the EAM system that you have configured in APM will be created from the Policy Recommendation record.	Populated automatically based on the properties defined for the associated Create Recommendation node.
Equipment ID	Character	The Record ID of the Equipment record to which the Policy Recommendation record is linked.	Populated automatically based on the properties defined for the associated Create Recommendation node. -or- Populated automatically with the Record ID of the Equipment record that is linked to the Functional Location record identified by the value in the **Functional Location ID** field.
Functional Location ID	Character	The Record ID of the Functional Location record to which the Policy Recommendation record is linked.	Populated automatically based on the properties defined for the associated Create Recommendation node. -or- Populated automatically with the Record ID of the Functional Location record that is linked to the Equipment record identified by the value in the **Equipment ID** field.

Field	Data Type	Description	Behavior and Usage
Recommendation Basis	Character	The policy that created the recommendation.	Populated automatically with the name of the Policy record to which the Policy Recommendation record is linked. This field does not appear on the Policy Recommendation datasheet.
Recommendation Description	Text	Information about the policy logic that caused the Policy Recommendation record to be created.	Populated automatically.
Recommendation Headline	Character	A short description of the recommended action.	Populated automatically based on the properties defined for the associated Create Recommendation node.
Recommendation ID	Character	A unique value that identifies the Policy Recommendation record.	Populated automatically when the recommendation is created.
Recommendation Priority	Character	The priority value used to rank the importance of the recommendation.	Populated automatically based on the properties defined for the associated Create Recommendation node.
Recommendation Type	Character	The type of recommendation record.	Populated automatically with the value Policy (PCY). This field does not appear on the Policy Recommendation datasheet.
Target Completion Date	Date	The date by which the recommended action should be completed.	Populated automatically based on the properties defined for the associated Create Recommendation node. This field is required.
Work Request Equipment	Character	The ID of the EAM system Equipment that is associated with the work request that was created from this Policy Recommendation record.	Populated automatically when the work request is created.

Field	Data Type	Description	Behavior and Usage
Work Request Functional Location	Character	The ID of the EAM system Functional Location that is associated with the work request that was created from this Policy Recommendation record.	Populated automatically when the work request is created.
Work Request Reference	Character	The ID of the EAM system work request that was created from this Policy Recommendation record.	Populated automatically when the work request is created.

Catalog Items

Policy Manager Folder

The Catalog folder `\\Public\Meridium\Modules\Policy Manager` contains the following items used by the Policy Designer module.

Item	Item Type	Behavior and Usage
Policy Overview	Dashboard	Displays the **Policy Designer Overview** page.
Policy Overview - Execution	Graph	Displays a summary of the execution results for the 20 most active policies within the past 30 days. This graph is based on the query Policy Overview- Execution and is displayed on the **Policy Designer Overview** page.
Recent Policy Execution Results	Query	Returns execution results of selected policies within the selected time range (in minutes).
Recent Policy Executions	Dashboard	Displays the table of recent policy execution results and the graph of time in queue.

Queries Folder

The Catalog folder `\\Public\Meridium\Modules\Policy Manager\Queries` contains the following items.

Query	Behavior and Usage
Email Address Query	Returns values in the First Name, Last Name, and Email Address fields in Human Resource records whose Email Address field contains a value. This query is used to display results in the grid on the **Choose User** window.
Events Query	Returns information about the Severity, Name, Description, Type, Start Time, End Time, and Duration (in seconds) of Policy Events. This query can be used with the Query node in the policy model to bring in policy events related to equipment or functional locations. The Events Query includes an Asset Key prompt, which can be passed to the Query node via the Entity Key system field of a family-based predecessor node. By default, the results of the Events Query are sorted in ascending order by start time. This query works with an SQL database. See Events Query (Oracle) for use with an Oracle database.
Events Query (Oracle)	Returns the same values as the Events query. This query, however, works in an Oracle database.
Family Policies with Upgrade Issues	Returns a list of family policies that may not be automatically upgraded when the APM database is upgraded to V5.0.0.0.0. **Note:** Failure to modify policies that were not successfully upgraded may result in unexpected system behavior when inserting, updating, or deleting records in the relevant families. Contact GE Vernova support in case of any difficulty with family policy modifications.
Policy Overview - Execution	Returns a summary of the execution results for the 20 most active policies within the past 30 days. The results of this query are used by the Policy Overview-Execution graph which is displayed on the **Policy Designer Overview** page.
Policy Overview – Module Workflow Policies Tile	Returns a list of all Module Workflow Policies in the APM database. The information displayed includes the policy name, whether the policy is active, last scheduled execution date, whether the policy is locked (security settings are applied), and the number of policy instances. The list also displays any policies that were not successfully upgraded to V5.0.0.0.0. Review the upgrade logs for any policy that may not be successfully upgraded and modify the policy model as appropriate.
Policy Overview - Overdue Recommendations Tile	Returns the Policy Recommendations assigned to any user for which the Target Completion Date has passed. The results of this query are displayed in the **Overdue Policy Recommendations** list in the **Policy Designer Overview** page.

Query	Behavior and Usage
Policy Overview - Policies	Returns a list of all Policies, other than Module Workflow policies in the APM database. The information displayed shows the policy name, whether the policy is active, last updated date, whether the policy is locked (security settings are applied) and the number of policy instances. The list also shows any policies that were not successfully upgraded to V5.0.0.0.0. Review the upgrade logs for any policy that could not be successfully upgraded and modify the policy model as appropriate. The results of this query are displayed in the **Policies** list in the **Policy Designer Overview** page.
Policy Overview - Policies Alternate Query	This query provides the same information as the **Policy Overview - Policies** query and a summary of the most recent policy execution result. This query may time out if your APM database contains a large number of policy execution history records. Therefore, it is not selected as the default content for the **Policies** list in the **Policy Designer Overview** page. To use this query for the **Policies** list, rename the **Policy Overview - Policies** query, and then update the name of this query to **Policy Overview - Policies**.
Policy Overview - Recommendations Tile	Returns the Policy Recommendations that are open and assigned to the current user. The results of this query are displayed in the **My Policy Recommendations** list in the **Policy Designer Overview** page.
Source Tags	Returns a list of all Source Tags, including the Tag ID, Description, Source (i.e. process historian), Type (e.g. String), UOM (i.e. Unit of Measure, as defined in the process historian), and Historical Readings Available (i.e. whether the process historian for this tag support querying for historical readings). This query is used to display results in the grid on the **Search** window when selecting a record to associate with an OT Connect Tag node.
Time in Queue	Returns a summary of the **Time in Queue**, that is, the time taken from the Execution Start time to the Trigger Sent time for 15-minute intervals in the past four hours.

Graphs Folder

The Catalog folder `\\Public\Meridium\Modules\Policy Manager\Dashboards\Graphs` contains the following items.

Graph	Behavior and Usage
Average and Max Wait Times in Queue for 15 min Intervals in Last 4 Hours	Returns a summary of the **Time in Queue**, that is, the time taken from the Execution Start time to the Trigger Sent time for 15-minute intervals in the past four hours. This graph displays in the **Policy Designer Overview** page and the **Recent Policy Executions** dashboard.

Note: Additional catalog items, for example, sub-queries, deprecated content, and so on, may exist, which are not documented in this topic.

Policy Examples

Policy Model Basic Principles

A policy model is made up of nodes and connections that define the policy logic. In order to build a functioning policy model, you must understand several basic principles.

Configuring a Policy Model

The following principles apply to working with a policy model:

- Policy models do not reference specific records. Rather, they contain nodes that reference families. You must use *policy instances* to identify the individual records whose values are evaluated when the policy is executed.
- The initial nodes in a policy model, that is, nodes with no predecessors, must be Input nodes other than Query nodes.
- A family policy must contain one Current Entity node for entity families, or one Current Relationship node for relationship families.
- A node can use an input from any predecessor node in the same logic path, even if the nodes are not directly connected.
- Any number of nodes can use an input from the same predecessor node.
- With the exception of the Or node and the Case node, a node will be executed only when all necessary preceding nodes have been successfully executed.
- A policy model can often be arranged in various configurations without impacting the execution results. You may want to arrange the policy model in the configuration that provides the best visual representation of the policy.
- Connections that start at a Condition or Logic node can be configured to create separate logic paths in a policy model. Specifically:
  - If the connection property is Yes, the corresponding path will be followed when the logical result of the Condition of Logic node is yes. If you do not configure a logic path for a connection, a Yes path is assumed but does not appear on the model.
  - If the connection property is No, the corresponding path will be followed when the logical result of the Condition or Logic node is no.
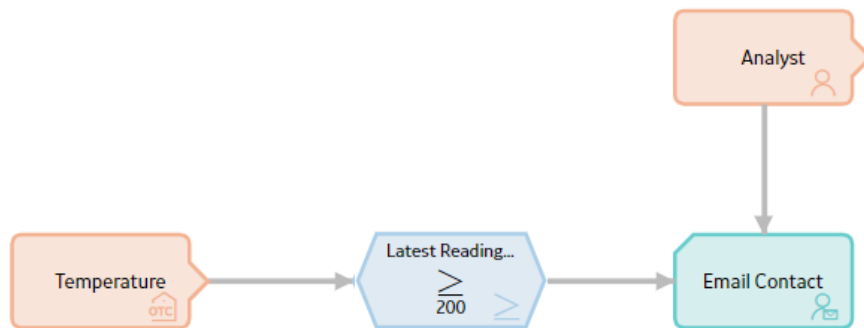
Configuring Node Properties

- Most nodes have outputs that successor nodes can use as inputs. You must specify inputs for each successor node using the **Properties** window that appears when you select the node in the policy model.

Note: Outputs and inputs may represent either a single value or a *collection* of values. The types of outputs that each node generates and the types of inputs that each node accepts is different for each node. When building a policy model, you must use corresponding input and output types.

- Any numeric values entered in Calculation nodes should be entered in the format matching the user's culture setting. For example, a user with German culture would enter `4,5` to represent four and a half, whereas a user with US-English culture would enter `4.5`.
- There are specific formats in which you can enter dates and times and amounts of time (that is, time spans). Refer to the respective topics for details.
- When using a policy node to update values in a record:
  - If a field has complex behavior defined by field-level rules (for example, rules for valid values) and field-level behaviors, this behavior will not be reflected in the **Properties** window or detected by policy validation. Therefore, you are responsible for ensuring that the values you specify are valid according to any baseline or custom field-level rules for the corresponding field.
  - If a field value is defined by a system code, the value that you specify in the corresponding section must be the system code, not the value that is displayed to the end user.

Policy Model Principles Illustrated

The principles for working with a policy model can be illustrated through the following example model:

In this example model, the node named Temperature is an OT Connect Tag node which represents a process historian tag. When this policy is executed, if the Latest Reading Value that is associated with the process historian tag is greater than or equal to 200, the APM system will send an email message to the email address that is specified in the Human Resource record that is associated with the Analyst User node.

The following table describes each of the policy model basic principles in the context of this example:

Policy Model Principle	Example
Policy models do not reference specific records. Rather, they contain nodes that reference families. You must use *policy instances* to identify the individual records whose values are evaluated when the policy is executed.	The Temperature and Analyst nodes represent families. The specific records whose values are evaluated are determined by policy instances.
The initial nodes in a policy model, that is, nodes with no predecessors, must be Input nodes other than Query nodes. A family policy must contain one Current Entity node for entity families, or one Current Relationship node for relationship families.	The Reading in Error node is a Current Entity node, which is a type of Input node. The Current Entity node is required for an entity family policy.
Most nodes have outputs that successor nodes can use as inputs. You must specify inputs for each successor node using the **Properties** window that appears when you select the node in the policy model. **Note:** Outputs and inputs may represent either a single value or a collection of values. The types of outputs that each node generates and the types of inputs that each node accepts is different for each node. When building a policy model, you must use corresponding input and output types.	The output Latest Reading Value from the Temperature node is used as an input to the Condition node. This output represents a single value, which corresponds to the type of input that the Condition node accepts. The value 200 is used as the second input to the Condition node, but it is not an output from another node. Instead, it is a constant value that is specified directly in the **Properties** window for the Condition node.
A node can use an input from any predecessor node in the same logic path, even if the nodes are not directly connected.	The Email Contact node can use an input from the Temperature node even though the two nodes are not directly connected.
Any number of nodes can use an input from the same predecessor node.	The Email Contact node and the Condition node can both use inputs from the Temperature node.

Policy Model Principle	Example
With the exception of the Or node and the Case node, a node will be executed only when all necessary preceding nodes have been successfully executed.	The Email Contact node will be executed only when all of the nodes preceding it have been successfully executed. If, for example, the condition defined in the Condition node was not met or if an error occurred when executing the Analyst node, the Email Contact node would not be executed.
A policy model can often be arranged in various configurations without impacting the execution results. You may want to arrange the policy model in the configuration that provides the best visual representation of the policy.	The execution results of the policy would be identical even if the Analyst node were connected to the Temperature node or the Condition node. The current configuration, however, provides a clear visual representation of the policy because the Email Contact node is the only node in the model that uses an input value from the Analyst node.
Connections that start at a Condition or Logic node can be configured to create separate logic paths in a policy model. Specifically: • If the connection property is Yes, the corresponding path will be followed when the logical result of the Condition of Logic node is yes. If you do not configure a logic path for a connection, a Yes path is assumed but does not appear on the model. • If the connection property is No, the corresponding path will be followed when the logical result of the Condition or Logic node is no.	A property is not defined for the connection between the Condition node and the Email Contact node, therefore, a value of Yes is assumed. This means that an email message is sent only if the preceding condition is true. If the condition is false, policy execution will not continue.

Policy Examples

You can use Policy Designer to accomplish tasks in a wide variety of scenarios. These examples illustrate a few of these scenarios.

Basic Policy and Policy Instance

Suppose that a certain motor is known to fail when it exceeds a temperature of 200 degrees. Part of the strategy for maintaining the motor might be to create a policy that continually monitors the motor's temperature.

In this example, temperature readings are gathered as readings associated with a process historian tag, which is represented in a policy by an OT Connect Tag node.

Policy Model
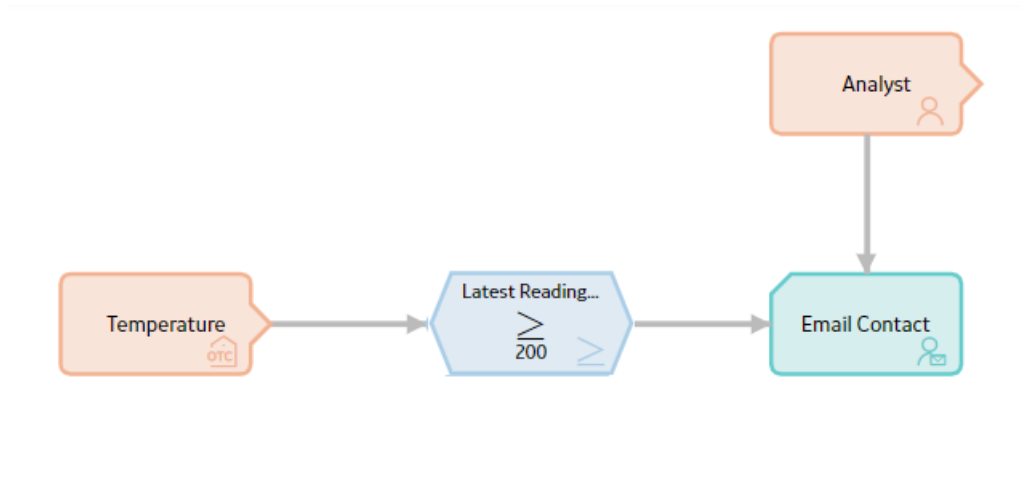
Policy Logic Summary

If the temperature of the motor exceeds 200 degrees, an email message will be sent to the responsible analyst. This policy would likely be configured to execute automatically when records belonging to a policy instance are updated.

Individual Node Descriptions

Node Name	Node Type	Description
Temperature	OT Connect Tag	Represents the process historian tag whose associated reading values are inputs to the policy.
		The specific process historian tags that this policy monitors are identified by the policy instances belonging to this policy.
Analyst	User	Represents the system user whose information is used by the policy.
		The specific user records used by this policy are identified by the individual policy instances belonging to this policy.
		Note: User nodes retrieve data from records in the Security User and Human Resource families.

Node Name	Node Type	Description
Latest Reading Value > 200	Comparison	Defines the logic that will be applied to values associated with the process historian tag that the OT Connect Tag node represents. Specifically, this Condition node specifies that the policy logic will continue (i.e., an email message will be sent) only when the Latest Reading Value is greater than or equal to 200.
Email Contact	Email Contact	Represents an action to send an email message. The email message will be sent to the email address defined in the specific user record that is included in the corresponding policy instance.

Policy Instance

A policy instance that is associated with this policy might look like this:
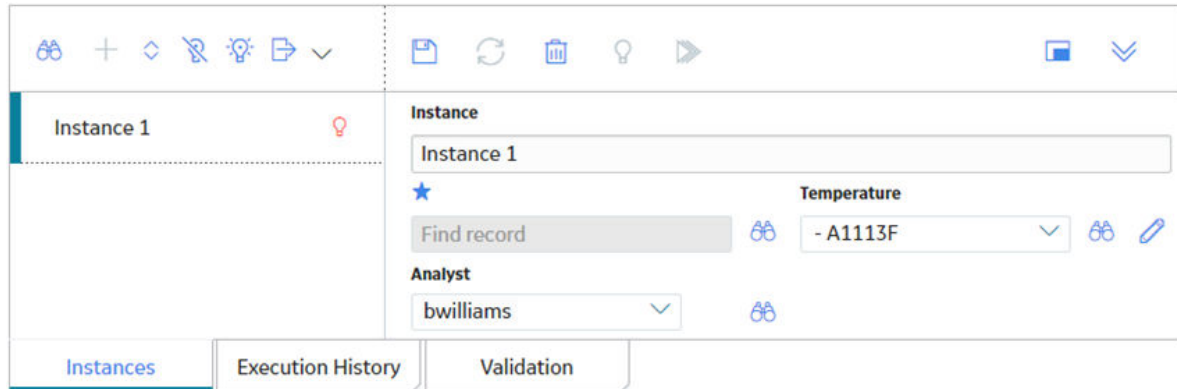
...where:

- The specific process historian tag is A1113F.
- The User ID associated with the specific user record is bwilliams.

When this policy is executed, if the process historian tag A1113F has a Latest Reading Value that is greater than or equal to 200, the APM system will send an email message to the user with the User ID bwilliams.

If you want to apply the same logic to monitor the temperatures of additional motors, you can define an additional policy instance for each specific process historian tag whose related Latest Reading Values contain the temperature readings for the motors that you want to monitor.

Monitoring a Single Value

Suppose a factory contains a heat exchanger shell, which can fail as a result of hot hydrogen attack. Part of the mitigation strategy for the heat exchanger shell, therefore, is to monitor the temperature of the process stream exposed to the shell component to ensure that it does not exceed 600 degrees Fahrenheit.

Context

The following table outlines how this policy is used within a larger workflow that also includes other APM modules or features.

Step	Module	Task
1	RBI	Assess the risk of loss of containment for a heat exchanger shell, which is associated with the damage mechanism Hot Hydrogen Attack.
2	OT Connect	Create a Source Tag record representing the process historian tag that record the process stream temperature of the heat exchanger shell.
3	AHM	Optionally create a Health Indicator record based on the Source Tag record.
4	Policy Designer	Create a policy to monitor the temperature values retrieved from the process historian.

Policy Model
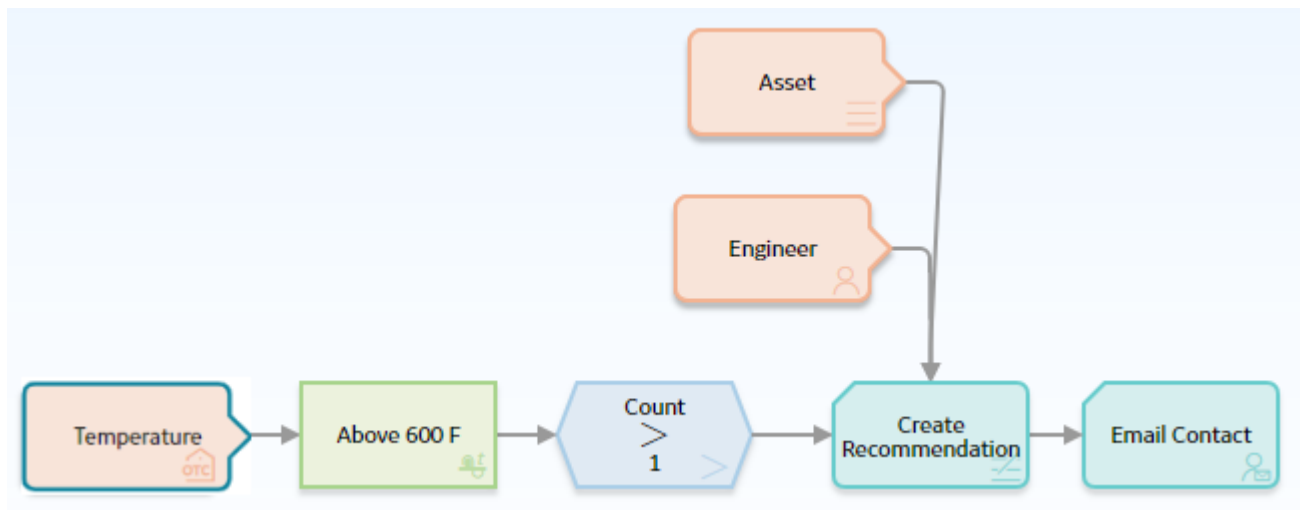
Policy Logic Summary

If the temperature of the heat exchanger shell rises above 600 degrees Fahrenheit more than once during a specified interval:

• A Policy Recommendation record will be created and configured to create a work request for immediate corrective action.
• An email message will be sent to a process engineer.

This policy should be configured with a scheduled execution frequency of once a week.

Individual Node Descriptions

Node Name	Node Type	Description
Temperature	OPC Tag	Provides reading values that are passed to the Above 600 F node.
Last Week	Collection Filter	Defines the time range for which historical reading values will be retrieved from the process historian. Note that if no time range is defined, all the available readings for the tag will be retrieved, subject to a maximum of 1,000,000 readings per retrieval.
Above 600 F	Threshold Statistics	Compares the collection of reading values returned by the Temperature node to the threshold value 600, and counts number of times that reading values exceed the threshold value.
Count > 1	Comparison	Compares 1 to the number of times that the reading values exceeded the threshold.
Create Recommendation	Create Recommendation	Creates a Policy Recommendation record linked to the Asset and assigned to the User.
Email Contact	Email Contact	Sends an email message.
Engineer	User	Supplies a User ID to the Create Recommendation node and an email address to the Email Contact node.
Asset	Entity	Represents the asset to which the Policy Recommendation record will be linked.

Monitoring Multiple Values

Suppose a factory contains a highly critical pump, which can pose risk if one of its bearings fails. Part of the mitigation strategy for the pump, therefore, is to perform a vibration analysis and monitor multiple variables in order to create a composite view of the pump's condition.

Context

The following table outlines how this policy is used within a larger workflow that also includes other APM modules or features.

Step	Module	Task
1	ASM	Assess the risk of bearing failure for the pump.
2	OT Connect	Create a Source Tag record representing the process historian tag that records the process stream temperature of the heat exchanger shell.
3	AHM	Create Health Indicator records based on the process historian tag.
4	Policy Designer	Create a policy to monitor the values that are stored in the process historian.

Policy Model
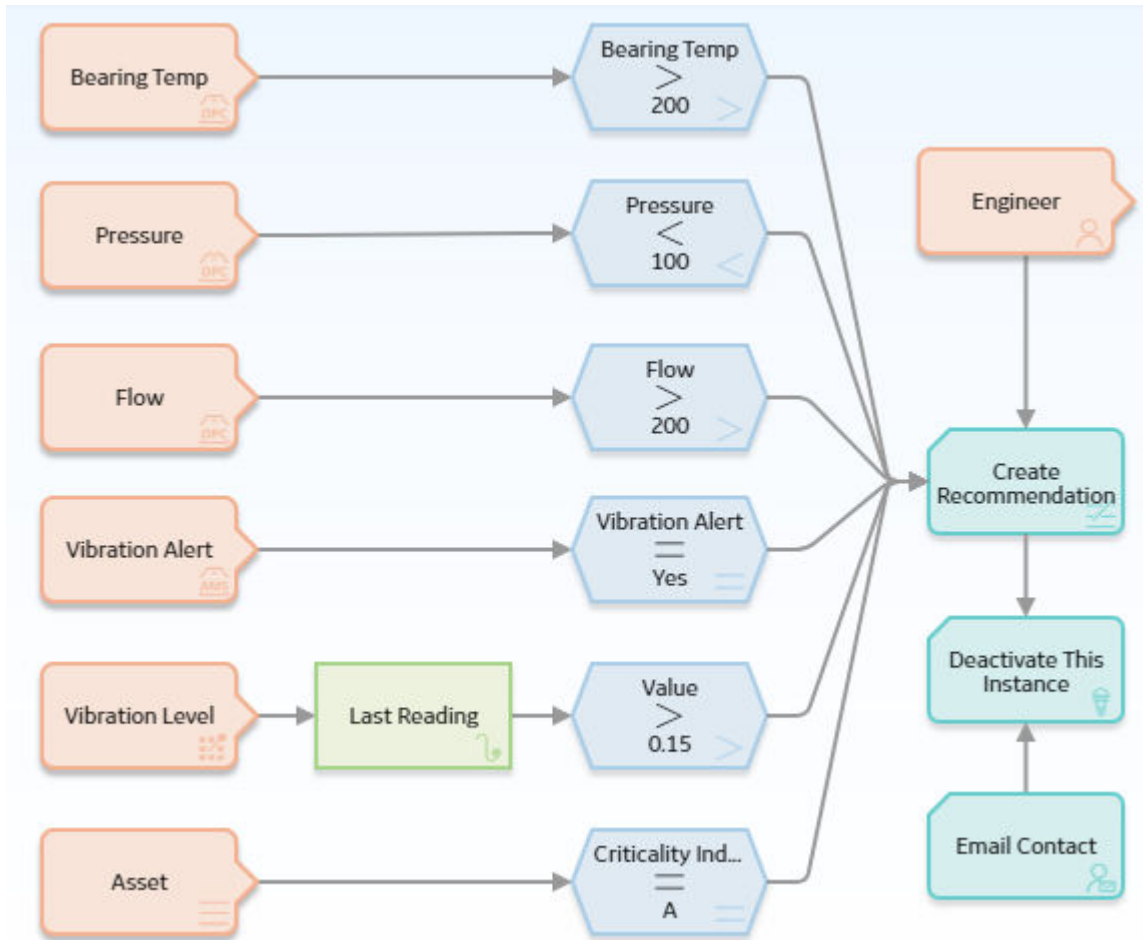
Policy Logic Summary

If the pump conditions indicate a pending failure (i.e., all of the conditions specified in the policy logic are true):

- A Policy Recommendation record will be created and configured to create a work request for immediate corrective action.

- An email message will be sent to a reliability engineer.
- The corresponding policy instance will be deactivated so that only one recommendation is created and only one email is sent for the same failure condition.

Individual Node Descriptions

Node Name	Node Type	Description
Bearing Temp	OPC Tag	Provides Latest Reading Values that are evaluated by the Bearing Temp > 200 node.
Pressure	OPC Tag	Provides Latest Reading Values that are evaluated by the Pressure < 100 node.
Flow	OPC Tag	Provides Latest Reading Values that are evaluated by the Flow > 200 node.
Vibration Level	Measurement Location	Provides Measurement Location readings that are evaluated by the Last Reading and Value > .15 nodes.
Asset	Entity	Represents the Equipment family and supplies a criticality indicator to the Condition node Criticality Indicator = A.
Pressure < 100	Comparison	Evaluates the process historian pressure readings to determine if any readings are less than 100.
Bearing Temp > 200	Comparison	Evaluates the process historian temperature readings to determine if any readings are greater than 200.
Flow > 200	Comparison	Evaluates the process historian flow readings to determine if any readings are greater than 200.
Last Reading	Last	Evaluates the collection of readings associated with the Vibration Level node and returns the last row in the collection, which, in this case, contains the most recent reading.
Value > .15	Comparison	Evaluates the value in the Value field of the Measurement Location reading returned by the Last Reading node to determine if it is greater than .15.
Criticality Indicator = A	Comparison	Evaluates the value in the Criticality Indicator field in the Equipment record to determine if it contains the value A.
Engineer	User	Supplies user information to the Create Recommendation and Email Contact nodes.

Node Name	Node Type	Description
Create Recommendation	Create Recommendation	Creates a Policy Recommendation record if all of the preceding conditions are met.
Deactivate This Instance	Deactivate This Instance	Deactivates automatically the policy instance whose values caused the Deactivate This Instance node to be executed.
Email Contact	Email Contact	Sends an email message.

Creating Health Indicator Value Records

The policy in this example illustrates how you can manipulate multiple readings to create health indicator readings that represent the overall health of an asset.

Suppose you are using Asset Health Manger to monitor the health of a pump using the following indicators, which are tracked using Health Indicator records whose primary source records are Measurement Location records:

- Flow Rate
- Pressure
- Lubrication
- General operation
- Vibration

You can use Asset Health Manager to evaluate the health of the pump in terms of these individual indicators by viewing gauges and trends for them on an individual basis. But you might also want to evaluate the overall health of the pump as a composite view of all of these underlying factors. If so, you can create a policy that facilitates this.

Context

The following table outlines how this policy is used within a larger workflow that also includes other APM modules or features.

Step	Module	Task
1	Rounds	Create the Measurement Location records that will be used to measure the pump's conditions. Complete the Rounds workflow to yield reading values for those checkpoints.
2	Asset Health Manager	Create Health Indicator records based on the Measurement Location records.
3	Policy Designer	Create a policy to monitor the values that are stored in the Health Indicator records and create a new health indicator that indicates the overall health of the pump.

Policy Model
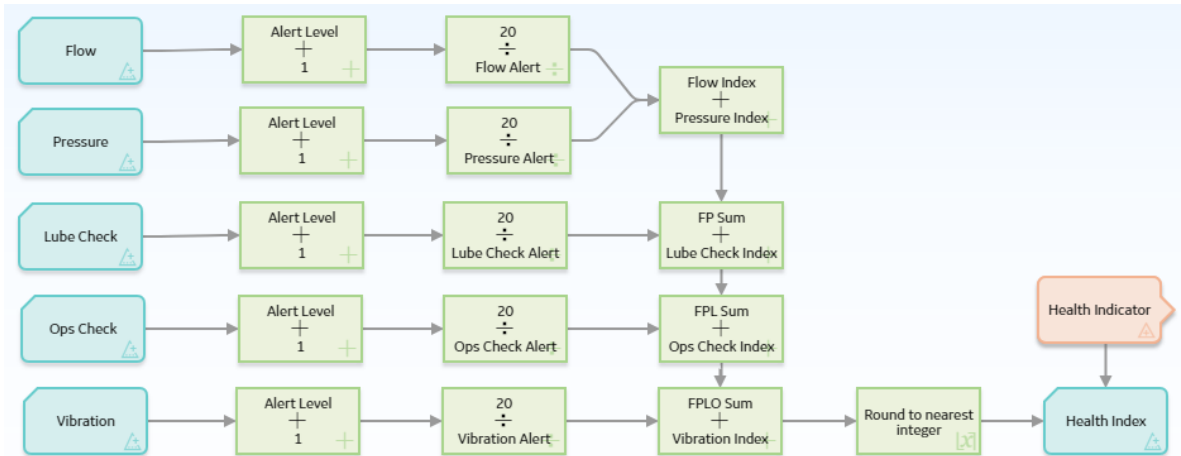
Policy Logic Summary

The values in the existing Health Indicator records will be used in calculations that determine an overall, single value that provides a composite view of the pump's health. This calculated value is stored in a Health Indicator Value record that gets created automatically each time the policy is executed and is linked to a different Health Indicator record that will be used to evaluate the overall health of the pump.

This policy would likely be configured with a scheduled execution of once a day.

Individual Node Explanation

Node Name	Node Type	Description
Flow, Pressure, Lube Check, Ops Check, and Vibration	Health Indicator	Represents various existing Health Indicator records whose primary source records are Measurement Locations. These records provide the values used in successor nodes to determine the overall health of the pump.
Alert Level + 1	Add	Adds 1 to the value in the Alert Level field in the corresponding Health Indicator record. The resulting value (n+1) is used as the denominator in the successor Divide nodes.
20 / <Health Indicator name> Alert	Divide	Divides 20 by the result of the corresponding Add node. This value represents the Health Index contribution for each node. 20 is used as the numerator to ensure that each individual Health Indicator has equal weight, (i.e., since there are five Health Indicators, 20 is one-fifth of 100).

Node Name	Node Type	Description
Various	Add	Combines the results from the Division nodes. This sum represents the overall health index of the pump on a numeric scale between 1 and 100, where a higher value is better than a lower value.
Round to nearest integer	Round	Rounds the value of the overall health index to the nearest integer.
Health Index	Add Value to Health Indicator	Adds the rounded value of the overall health index to a new Health Indicator Value record and links the record to a Health Indicator record.
Health Indicator	Health Indicator	Represents the Health Indicator record to which the Health Indicator Value record will be linked when the policy is executed.

Creating and Updating a Policy Event Record

Suppose a factory contains a heat exchanger, which can fail as a result of hot hydrogen attack. Part of the mitigation strategy for the heat exchanger shell, therefore, is to monitor the temperature of the process stream exposed to the shell component to ensure that it does not exceed 600 degrees Fahrenheit. Specifically, the organizational policy dictates that all operation above 550 degrees Fahrenheit will be recorded so that mechanical integrity experts can evaluate the risk related to exceeding a safe operating temperature.

Context

The following table outlines how this policy is used within a larger workflow that also includes other APM modules or features.

Step	Module	Task
1	RBI	Assess the risk of loss of containment for a heat exchanger shell, which is associated with the damage mechanism Hot Hydrogen Attack.
3	Policy Designer	Create a policy that monitors the temperature values that are stored in the process historian and creates and updates a Policy Event record.

Policy Model
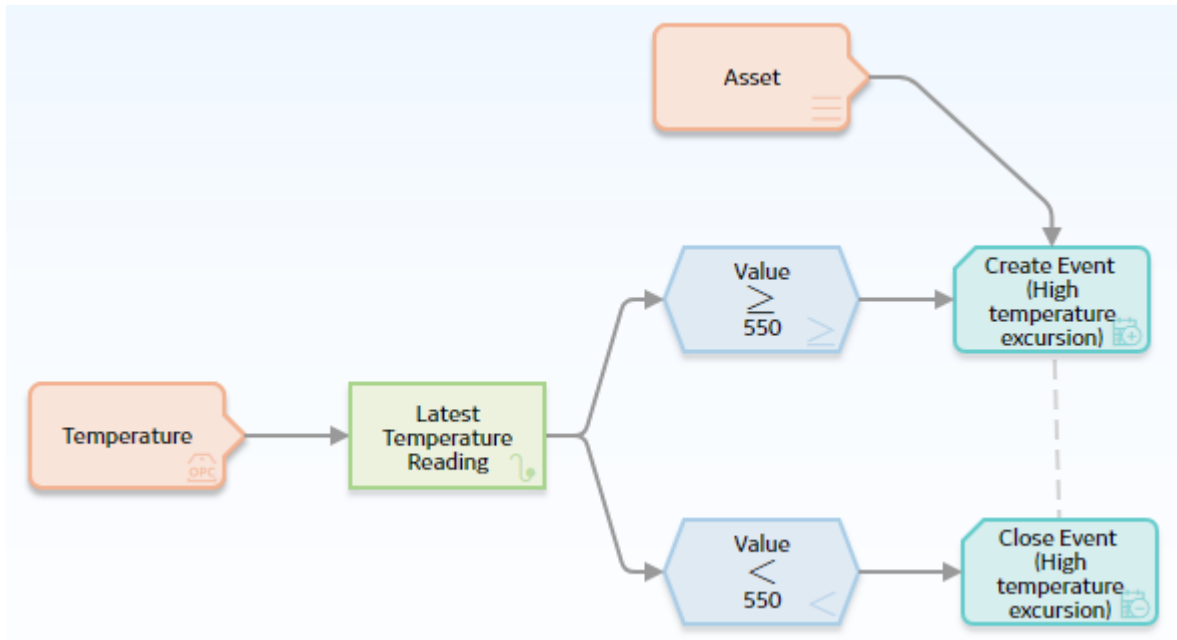
Policy Logic Summary

If the temperature of the process stream exposed to the exchanger shell:

- Rises to or above 550 degrees Fahrenheit, a Policy Event record will be created.
- Falls back below 550 degrees Fahrenheit (after rising to or above 550 degrees Fahrenheit), the Policy Event record will be updated to indicate that the event has closed.

This policy would likely be configured to execute automatically when records belonging to a policy instance are updated.

Individual Node Descriptions

Node Name	Node Type	Description
Temperature	OPC Tag	Provides reading values that are passed to the Latest Temperature Reading node.
Latest Temperature Reading	Last	Provides reading values that are passed to the Value ≥ 550 and Value < 550 nodes for evaluation.
Value ≥ 550	Comparison	Compares the reading value returned by the Latest Temperature Reading node. If the reading value is greater than or equal to 550, the Create Event node is triggered.
Value < 550	Comparison	Compares the reading value returned by the Latest Temperature Reading node. If the reading value is less than 550, the Close Event node is triggered.

Node Name	Node Type	Description
Asset	Entity	Represents the asset that is associated with the Policy Event record that is created by the Create Event node.
Create Event (High temperature excursion)	Create Event	Creates a Policy Event record.
Close Event (High temperature excursion)	Close Event	If there is an open Policy Event record for this instance of the policy, updates the record to indicate that the event is now closed.

Creating an Entity Record and Linking it to Another Record

The policy in this example illustrates how you can use a policy to monitor certain conditions and, in a specific scenario, create new entity and relationship records in the APM database.

Suppose that you are a Rounds user and you know that when a reading value exceeds a certain threshold, a recommendation should be created for a mobile Rounds user to address the situation. You can configure a policy to monitor these conditions and then create an OPR Recommendation that mobile Rounds users can then view on a handheld device as they complete their routes.

Context

The following table outlines how this policy is used within a larger workflow that also includes other APM modules or features.

Step	Module	Task
1	Rounds Designer	Create a Route with Measurement Locations and define the threshold values for readings.
2	Rounds Data Collection	Enter new readings for Measurement Locations on a Route.
3	Policy Designer	Create a policy to monitor the reading values and create a new OPR Recommendations if the threshold limits are exceeded and an OPR recommendation has not been added since the reading value was taken.
4	Rounds Data Collection or Rounds Designer	View the recommendations on a handheld device or in Route Manager and take any action necessary.

Policy Model
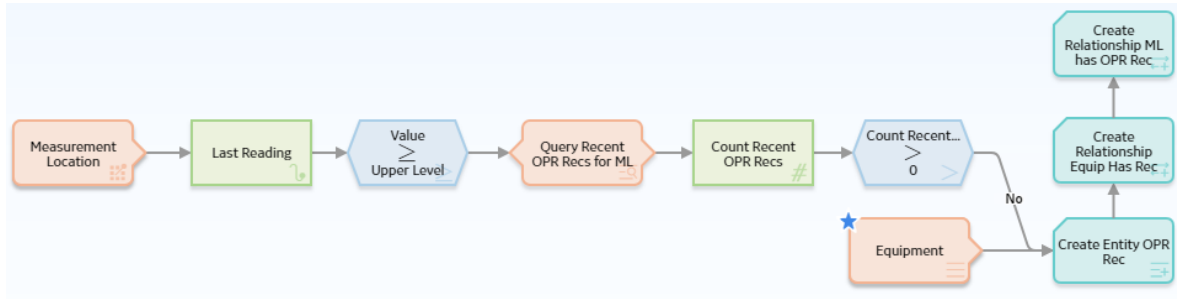
Policy Logic Summary

If a reading value is recorded that exceeds the Upper Level 2 threshold, an OPR Recommendation will be created and linked to the associated Equipment and Measurement Location. There is also a check to see if an OPR Recommendation was already created since the reading was taken, in which case a Recommendation will not be created.

This policy would likely be configured to execute automatically when records belonging to a policy instance are updated.

Individual Node Descriptions

Node Name	Node Type	Description
Measurement Location	Measurement Location	Provides reading values that are evaluated by the policy.
Last Reading	Last	Represents the most recent reading associated with the Measurement Location.
≥ Upper Level 2	Greater Than or Equal To	Compares the most recent reading with the Upper Level 2 threshold value defined for the Measurement Location. If the reading is greater than or equal to the threshold value, the successor nodes are executed.
Query Recent OPR Recs for ML	Query	Gets any OPR Recommendations that are linked to the Measurement Location, where the created date is after the date and time of the most recent reading.
Count Recent OPR Recs	Count	Calculates the number of results returned by the Query node.
Count Recent Recs > 0	Greater Than	Determines whether any OPR Recommendations linked to the Measurement Location were created after the date and time of the most recent reading. If there were not, the successor nodes are executed.

Node Name	Node Type	Description
Create Entity OPR Rec	Create Entity	Creates a new OPR Recommendation record using various data from the Measurement Location, Equipment, and the Last Reading nodes.
Equipment	Entity	Represents the piece of equipment related to the Measurement Location.
Create Relationship Equip Has Recommendations	Create Relationship	Creates a relationship between the OPR Recommendation record being created and the Equipment record represented by the Equipment node.
Create Relationship ML has OPR Recommendations	Create Relationship	Creates a relationship between the OPR Recommendation record being created and the Measurement Location record represented by the Measurement Location node.

Creating a Production Event Record

The policy in this example illustrates how you can analyze relevant readings from external systems (for example, process historians), and, when certain conditions are met, create a Production Event record to trigger the Production Loss Analysis workflow.

Context

The following table outlines how this policy is used within a larger workflow that also includes other APM modules or features.

Step	Module	Task
1	Policy Designer	Define a policy and set up instances to analyze the data received from the process historian that will identify conditions indicating a process interruption and create a Production Event record if this occurs.
2	Production Loss Analysis	Investigate the production event and update the Production Event record with the findings, including the root cause of the event and the related losses. This information will then be used in a production reliability analysis.

Policy Model
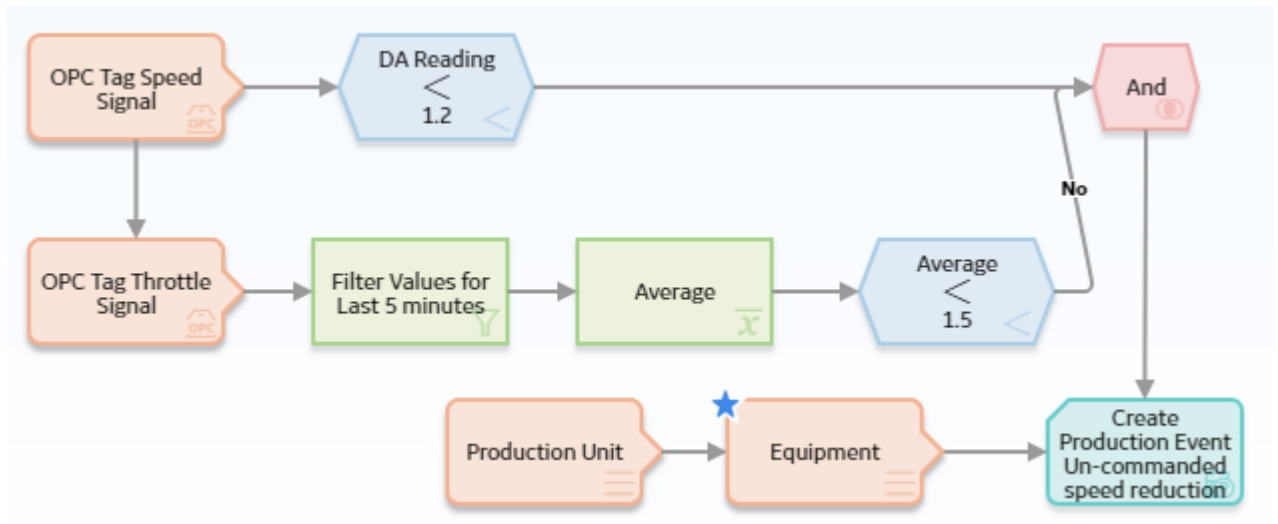
Policy Logic Summary

If the speed of a conveyor falls below the minimum operating speed, and the speed reduction was not commanded, a Production Event record is created. The policy could be expanded to also send an email message is sent to the responsible engineer.

This policy would likely be configured to execute automatically when the values associated with the OPC Tag Speed Signal node are updated.

Individual Node Descriptions

Node Name	Node Type	Description
OPC Tag Speed Signal	OPC Tag	Represents the speed of a conveyor. Changes in values associated with this node trigger the policy execution.
DA Reading < 1.2	Less Than	Compares the speed of the conveyor with the minimum operating speed of 1.2 ft/sec.
OPC Tag Throttle Signal	OPC Tag	Represents the throttle signal of a conveyor.
Filter Values for Last 5 minutes	Collection Filter	Filters the throttle signal values to only those from the last 5 minutes (that is, the 5 minutes prior to the speed of the conveyor falling below 1.2 ft/sec).
Average	Average	Determines the average throttle signal for the last 5 minutes.
Average < 1.5	Less Than	Compares the average throttle signal with 1.5 ft/sec to determine whether the speed reduction was commanded.

Node Name	Node Type	Description
And	And	Evaluates the preceding logic branches and allows policy execution to continue only when the speed of the conveyor falls below 1.2 ft/sec and the average throttle signal for the last 5 minute was not less than 1.5 ft/sec.
Production Unit	Entity	Represents a Production Unit record.
Equipment	Entity	Represents the conveyor being monitored.
Create Production Event Un-commanded speed reduction	Create Production Event	When all prior conditions are met (that is, policy logic indicates that the speed reduction was not commanded), creates a new Production Event record. The properties for the Create Production Event node are configured such that the new record is linked to the related Production Unit record and the Equipment record representing the conveyor is defined as the causing asset.

Input Nodes

About Input Nodes in Policies

Input nodes represent various items and values that you can use as inputs to the policy logic. Primarily, Input nodes represent the records whose values are evaluated by a policy.

With the exception of Query, Query Entity, and Constant nodes, you must use policy instances to assign specific records to each Input node in a policy. For a Point Value node, you can also use policy instances to assign specific values to the node.

Because Input nodes provide successor nodes with values to evaluate, the first node in a policy model must be an Input node.

Input Nodes

- AMS Asset
- Constant
- Entity
- GE Tag
- Health Indicator
- Measurement Location
- OT Connect Tag
- Point Value
- Query
- Query Entity
- User

Note: You must have the appropriate licenses and family-level privileges to view and use nodes that are associated with specific families. For example, to view a Measurement Location node, the Operator Rounds license must be active and you must have View permissions on the Measurement Location family. Otherwise, the button for the Measurement Location node will be disabled.

AMS Asset Nodes in Policy Designer

Note: You can access the AMS Asset node only if the AMS Analytics license is active in your APM system.

Note: The AMS Asset node was designed to work with the integration between APM and Emerson AMS. This integration is no longer supported.

An AMS Asset node is an Input node that represents the AMS Asset family. You can use the AMS Asset node to access records in the AMS Asset family and information about related alerts.

An AMS Asset node generates the following outputs:

- Has Active Alerts, which represents a value of yes or no depending on whether or not the AMS Asset has any active alerts (i.e. the AMS Asset record is linked to an AMS Asset Alert record with the value True in the Active field).
- Any field in the AMS Asset family.
- The following system fields for the AMS Asset family: Entity Key, Family Key, Entity ID, and Site Key.

Node Properties

The **Properties** window for an AMS Asset node contains the items that are described in the following table:

Item	Description
Primary Record	Specifies whether the node represents a primary record.
Trigger	Specifies whether changes to the record that is associated with the node will result in policy execution.

The following example illustrates how you can use the AMS Asset node with successor nodes to trigger actions based on whether or not an AMS Asset has any active alerts. Consider the following nodes and connections.
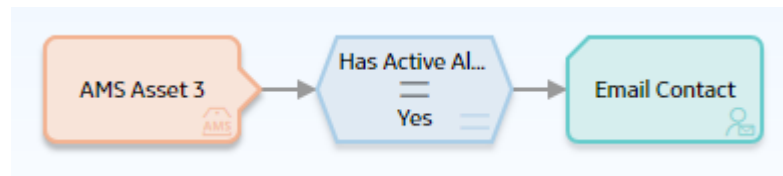
As shown in the following image, you can use the **Properties** window for the Equal node to select the AMS Asset node and the Has Active Alerts option. When this option is selected, the output of the AMS Asset node is either yes or no, depending on whether or not the AMS Asset is in an alert state. Together, the AMS Asset node and the Equal node indicate that an email message should be sent if the corresponding AMS Asset record has any active alerts.
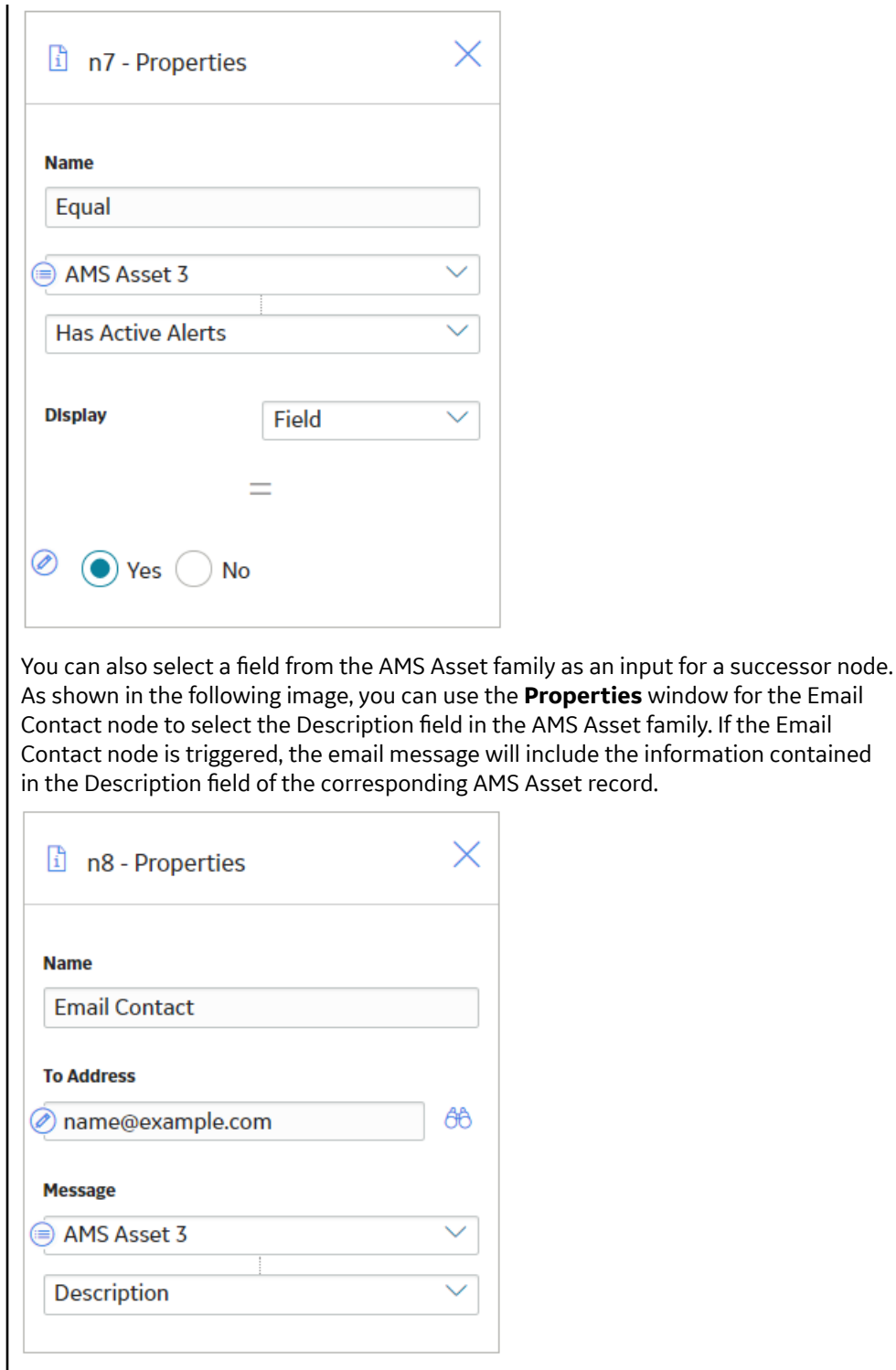
You can also select a field from the AMS Asset family as an input for a successor node. As shown in the following image, you can use the **Properties** window for the Email Contact node to select the Description field in the AMS Asset family. If the Email Contact node is triggered, the email message will include the information contained in the Description field of the corresponding AMS Asset record.

Constant Nodes in Policy Designer

A Constant node is an Input node that represents a specific value that does not change from instance to instance. You can use a Constant node for input values that are the same for all instances and are used in multiple places in the policy model.

The output of a Constant node is the value that you specify in the **Value** section of the node's **Properties** window.

Node Properties

The **Properties** window for a Constant node contains the items that are described in the following table.

Item	Description	Notes
Data Type list	Specifies the type of data that the node represents.	This property is not required, but it is highly recommended to select a value. This minimizes the chance of the Constant node result either being misinterpreted during the policy execution and the validation, or being displayed in an incorrect format while viewing the execution and the validation results.
Value box	Specifies the value that the node represents.	If the data type for the Constant node is a Data Frame, then, on the **Properties** window, **<DATAFRAME>** appears in the **Value** box. You can select ⚬⚬⚬ to access the **Edit Data Frame** window, in which you can view the Data Frame or configure its input values.

Consider the following nodes and connections.
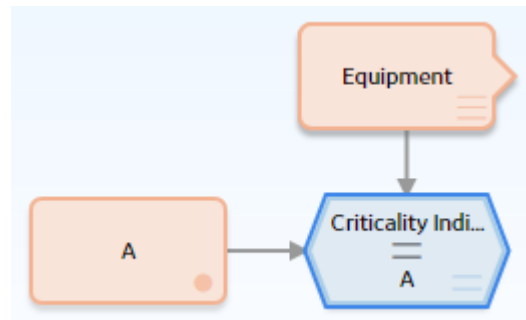
As shown in the following image, you can use the **Properties** window for the Condition node to select the A Constant node and its value, which is A. With these options selected, the Condition node evaluates whether or not the value in the Criticality Indicator field of a specified Equipment record is A.

Regardless of the Equipment records specified via policy instances, the Condition node will always compare the value in the Criticality Indicator field to A.

Entity Nodes in Policy Designer

An Entity node is an Input node that represents any APM entity family. You can use the Entity node to access information that is stored in a record belonging to any APM entity family.

Note: The Entity node displays only the entity families for which your APM system has active licenses, and for which you have access permissions.

An Entity node generates the following outputs:

- Any field in the family that the Entity node represents.
- The following system fields that the Entity node represents: Entity Key, Entity ID, Family Key, and Site Key.
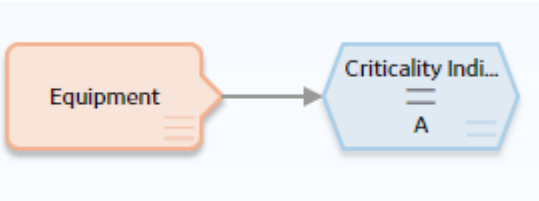
Node Properties

The **Properties** window for an Entity node contains the items that are described in the following table.

| Item	Description	Notes
Primary Record	Specifies whether the node represents a primary record.	None
Trigger	Specifies whether changes to the record that is associated with the node will result in policy execution.	None
Family ID list	Specifies the family that the Entity node represents.	The **Family ID** list contains all of the APM entity families from which you can access record information. This value is required. **Important:** Ensure that the family you select has not be excluded from the global search. If it has, you will not be able to assign records from that family to a policy instance.

Entity nodes are often the starting point in policy models because they provide successor nodes with fields to evaluate. For example, consider the following nodes and connection.

In this example, the Equipment Entity node provides the Condition node with a field, Criticality Indicator, to evaluate.

GE Tag Nodes in Policy Designer

Note: You can access the GE Tag node only if the GE Analytics license is active in your APM system.

Note: The GE Tag node was designed to work with the integration between APM and SmartSignal or System 1. This integration is no longer supported.

A GE Tag node is an Input node that represents the GE Tag family. You can use the GE Tag node to access records in the GE Tag family and information about related events.

A GE Tag node generates the following outputs:

- GE Tag Events, which represents the collection of GE Tag Event records that are associated with a specified GE Tag record. This output contains the start time, severity level, and type of each event in the collection and can be used only with successor nodes capable of handling collections.
- Any field in the GE Tag family.
- The following system fields in the GE Tag family: Entity Key, Entity ID, Family Key, and Site Key.

Node Properties

The **Properties** window for a GE Tag node contains the items that are described in the following table.

Item	Description
Primary Record	Specifies whether the node represents a primary record.
Trigger	Specifies whether changes to the record that is associated with the node will result in policy execution.

The following example illustrates how you can use the GE Tag node with successor nodes to evaluate a related GE Tag Event record. Consider the following nodes and connections.
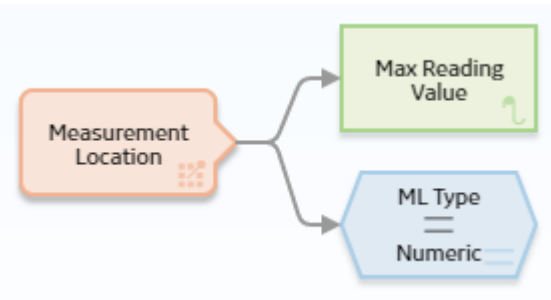

In this example, the GE Tag node is connected to a Last node and to a Greater Than Condition node. As shown in the following image, you can use the **Properties** window for the Last node to select the GE Tag node and the GE Tag Events option. When this option is selected, the Last node evaluates the events that are associated with the GE Tag record and returns the most recent event.
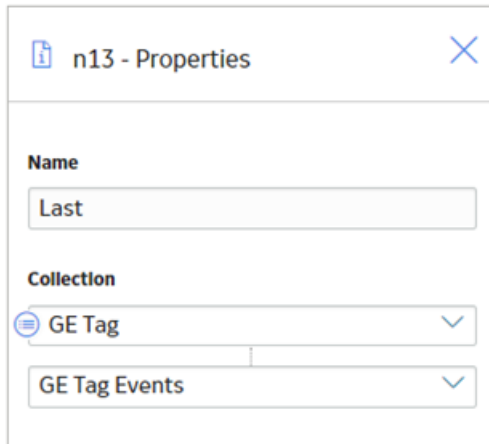
You can then use subsequent nodes to evaluate information related to the most recent event. As shown in the following image, you can use the Greater Than node to evaluate whether or not the most recent event has a severity greater than 0.

Health Indicator Nodes in Policy Designer

A Health Indicator node is an Input node that represents the Health Indicator family. You can use the Health Indicator node to access records in the Health Indicator family and information about the readings associated with a Health Indicator's source record.

Note: You can access the Health Indicator node only if the Asset Health license is active in your APM system.

When you associate a policy instance with a Health Indicator node, you can assign an existing Health Indicator record to the node or create a new Health Indicator record without a source and assign it to the node. For Health Indicator records without sources, you can use an Add Value to Health Indicator node to create Health Indicator Value records and link them to the related Health Indicator record.

A Health Indicator node generates the following outputs:

- Readings, which represents the collection of all readings associated with the Health Indicator's source record. This output contains a value and timestamp for each reading in the collection and can be used only with successor nodes capable of handling collections.
- Any field in the Health Indicator family.
- The following system fields in the Health Indicator family: Entity Key, Entity ID, Family Key, and Site Key.

Note: When you configure a policy with a Health Indicator node for Automatic Execution, the Asset Health Indicator service must be running in order for new readings to trigger a policy execution. If a new reading is added with a timestamp earlier than a reading that was previously processed by the health indicator service, the policy will not be triggered, because the Asset Health Indicator service does not update the health indicator record in this case.

Node Properties

The **Properties** window for a Health Indicator node contains the items that are described in the following table.

| Item	Description
Primary Record	Specifies whether the node represents a primary record.
Trigger	Specifies whether changes to the record that is associated with the node will result in policy execution.

The following example illustrates how you can use the Health Indicator node with successor nodes to evaluate fields in a Health Indicator record and the associated collection of readings. Consider the following nodes and connections.
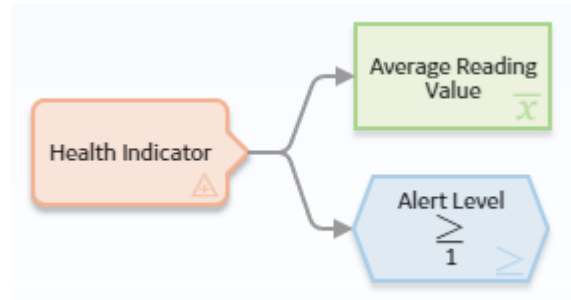
In this example, the Health Indicator node is connected to the Average Reading Value Average node and to a Greater Than or Equal Condition node.

As shown in the following image, you can use the **Properties** window for the Average node to select the Health Indicator node and the Reading option. When these options are selected, this node evaluates the readings that are associated with the Health Indicator record and returns the average reading value.
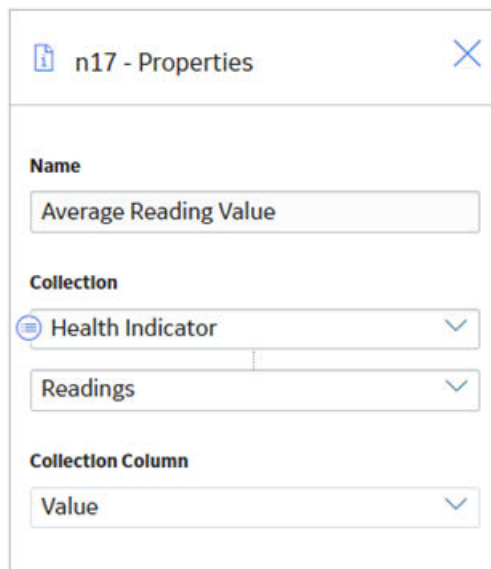
You can also select a field from the Health Indicator family as an input for a successor node. As shown in the following image, you can use the **Properties** window for the Greater Than or Equal node to select the Alert Level field in the Health Indicator family. This node, therefore, evaluates whether or not the value in the Alert Level field of the corresponding Health Indicator record is greater than or equal to 1, which would indicate that the Health Indicator record meets the Warning or Alert status criteria.
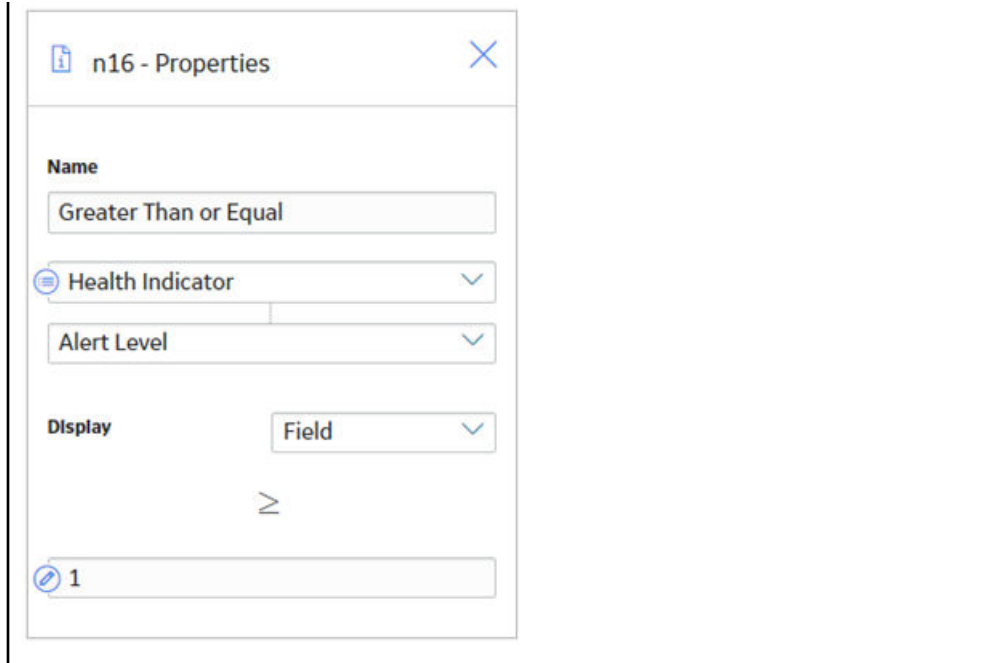
Measurement Location Nodes in Policy Designer

A Measurement Location node is an Input node that represents the Measurement Location family. You can use the Measurement Location node to access records in the Measurement Location family and information about related readings.

A Measurement Location node generates the following outputs:

- Readings, which represents the collection of all measurement readings that are associated with a specified Measurement Location record. This output contains a value and timestamp for each reading in the collection and can be used only with successor nodes capable of handling collections.

  Note: If the policy is triggered by a new or updated reading, the Readings collection includes only readings with timestamps less than or equal to the timestamp of the triggering reading.
- Any field in the Measurement Location family.
- The following system fields in the Measurement Location family: Entity Key, Entity ID, Family Key, and Site Key.

Node Properties

The **Properties** window for a Measurement Location node contains the items that are described in the following table.

Item	Description
Primary Record	Specifies whether the node represents a primary record.
Trigger	Specifies whether changes to the record that is associated with the node will result in policy execution.

The following example illustrates how you can use the Measurement Location node with successor nodes to evaluate fields in a Measurement Location record and the collection of readings associated with a Measurement Location record. Consider the following nodes and connections.

In this example, the Measurement Location node is connected to a Max Calculation node and an Equal Condition node.

As shown in the following image, you can use the **Properties** window for the Max node to select the Measurement Location node and the Readings option. When this option is selected, the Max node evaluates the readings that are associated with the Measurement Location record and returns the reading with the highest value.

You can also select a field from the Measurement Location family as an input for a successor node. As shown in the following image, you can use the **Properties** window for the Equal node to select the ML Type field in the Measurement Location family. This node evaluates whether or not the ML Type field contains the exact value Numeric.
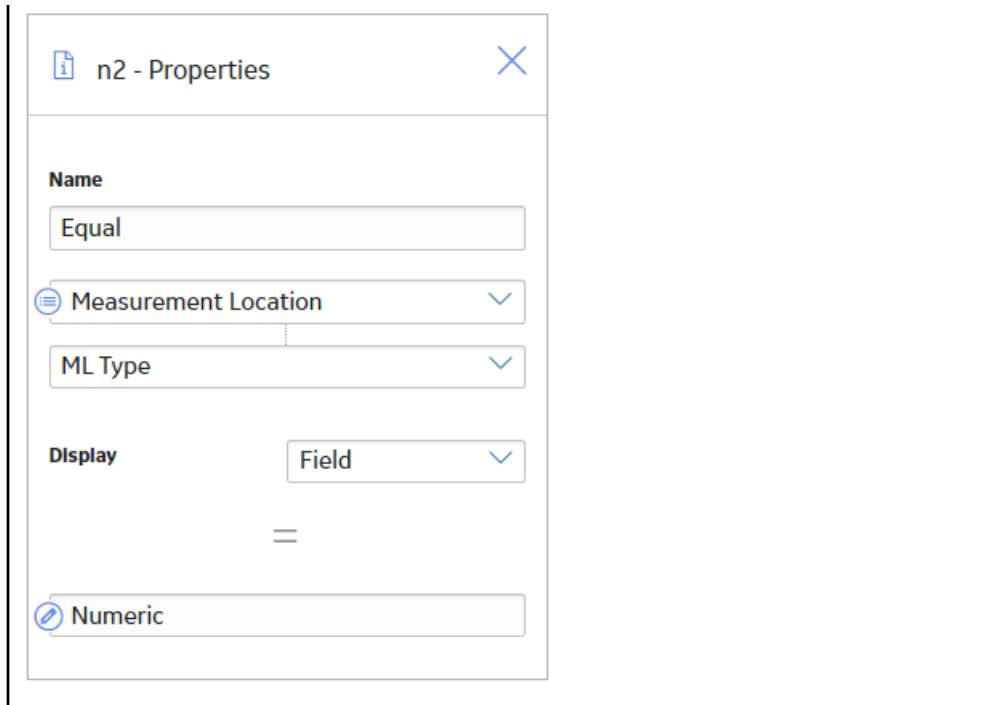
Measurement Step Nodes in Policy Designer

A Measurement step node generates the following outputs:

- Readings, which represents the collection of all measurement readings that are associated with a specified Measurement step record. This output contains a value and timestamp for each reading in the collection and can be used only with successor nodes capable of handling collections.
- Any field in the Measurement step family.
- The following system fields in the Measurement step family: Entity Key, Entity ID, Family Key, and Site Key.

Note: In this documentation, the Excel template is referred to as the data loader workbook.

Node Properties

The **Properties** window for a Measurement step node contains the items that are described in the following table:

Item	Description
Primary Record	Specifies whether the node represents a primary record.
Trigger	Specifies whether changes to the record that is associated with the node will result in policy execution.

The following example illustrates how you can use the Measurement step node with successor nodes to evaluate fields in a Measurement step record and the collection of readings associated with a Measurement step record. Consider the following nodes and connections.
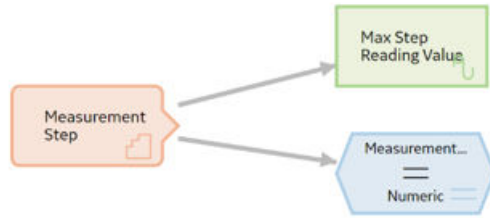
In this example, the Measurement step node is connected to a Max Calculation node and an Equal Condition node.

As shown in the following image, you can use the **Properties** window for the Max node to select the Measurement step node and the Readings option. When this option is selected, the Max node evaluates the readings that are associated with the Measurement step record and returns the reading with the highest value.
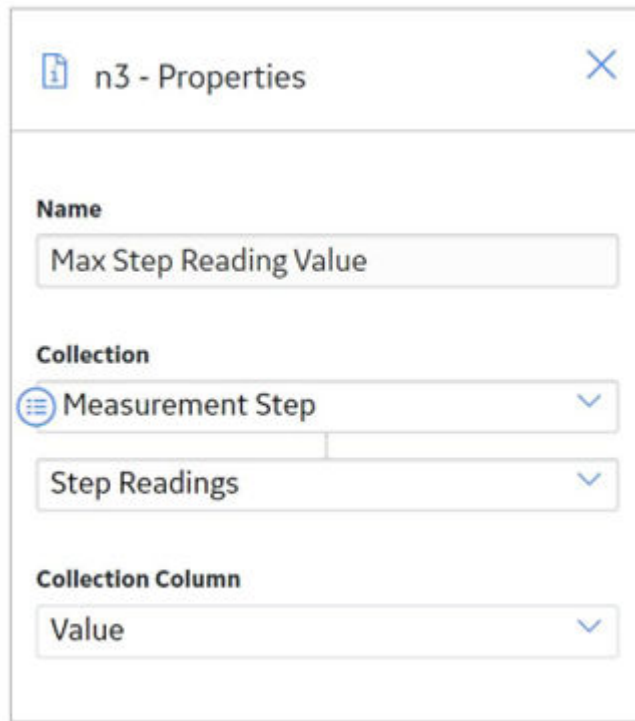
You can also select a field from the Measurement step family as an input for a successor node. As shown in the following image, you can use the **Properties** window for the Equal node to select the ML Type field in the Measurement step family. This node evaluates whether or not the ML Type field contains the exact value Numeric.

OT Connect Tag Nodes in Policy Designer

An OT Connect Tag node is an Input node that represents a tag in a process historian accessed by the OT Connect service. You can use the OT Connect Tag node to access records in the Source Tag family, limit values defined for policy instances, and the related reading data that is accessed through the OT Connect service from the process historian.

Note: You can access the OT Connect Tag node only if the APM OT Connect license is active in your APM system.

An OT Connect Tag node generates the following outputs:

- Latest Reading Quality Is Good, which indicates whether the quality of the most recent reading available from the process historian is good. You can use this output to control the policy logic based on the most recent reading quality.
- Latest Reading Value, which represents the most recent reading available from the process historian.
- Latest Reading Timestamp, which represents the date and time that the most recent reading was recorded in the process historian.
- Historical Readings, which represents the collection of historical readings taken over a period in the process historian. This output contains a value and time stamp for each reading in the collection and can be used only with successor nodes capable of handling collections.

    Note:

    ◦ You should use a Collection Filter node to limit the readings to only those in the time span that you want to evaluate. This way, only the required readings will be retrieved during policy execution. If you do not use a collection Filter, 2 years of historical readings will be retrieved, subject to a maximum retrieval of 1,000,000 readings. The default time range for retrieving historical readings can be modified in the APM Server configuration. For information on configuring the limits, refer to the Configure the Default Historical Readings Time Range for the OT Connect Tag node.
    ◦ If the policy is triggered by a new or updated reading, the Historical Readings collection includes only readings with timestamps less than or equal to the timestamp of the triggering reading.
    ◦ The Last Reading Quality is Good, Last Reading Value, and Last Reading Timestamp outputs are not supported for Source Tags related to OT Sources which support only OPC HDA (for example, GE Historian). An error will occur in policy execution or validation if you attempt to use these node outputs with Source Tags from these OT Sources.
- The following fields from the Source Tag family:

    ◦ Comments
    ◦ Data Type
    ◦ Deleted from Source
    ◦ Description
    ◦ Historical Readings

- ◦ Id
- ◦ Last Reading Quality is Good
- ◦ Last Reading Timestamp
- ◦ Last Reading Value
- ◦ Name
- ◦ Raw UoM
- ◦ Source
- ◦ UoM
- The following system fields in the Source Tag family: Entity Key, Family Key, Entity ID, and Site Key.
- Standard limit fields, i.e. Numeric Upper Level 3, Numeric Upper Level 2, Numeric Upper Level 1, NumericLower Level 1, Numeric Lower Level 2, Numeric Lower Level 3, Character Upper Level 3, Character Upper Level 2, Character Upper Level 1, Character Lower Level 1, Character Lower Level 2, and Character Lower Level 3.

Note:

- When a Source Tag record is assigned to an OT Connect Tag node in a policy instance, you can configure the limit values to be evaluated during policy execution. These limit values are associated with the use of the selected Source Tag only in this specific policy instance and are stored in a Content Map record associated with the policy instance and the Source Tag. This means that you can use different limits in each policy instance where a Source Tag is used, depending on the purpose of the policy. For information on configuring the limits, see Configure OT Connect Tag Limits from the Instances Pane.
- If your APM database has been upgraded from V4.3.1.0.0 or earlier version, the OT Connect Tag node may display Obsolete fields. The policy must be reconfigured to remove any references to the Obsolete fields. For more information, contact the GE Vernova Support for advice.

Node Properties

The **Properties** window for an OT Connect Tag node contains the items described in the following table.

Item	Description
Historical Readings Required	Specifies whether the Source Tag must represent a process historian tag that provides access to historical readings. The Historical Readings output is available in the policy design when this setting is selected. This setting is set to Yes by default.
Primary Record	Specifies whether the node represents a primary record.
Trigger	Specifies whether changes to the record that is associated with the node will result in policy execution.

OT Connect Tag node

The following is an example of how you can use the OT Connect Tag node with successor nodes to evaluate different types of data related to the OPC Tag family. Consider the following nodes and connections.
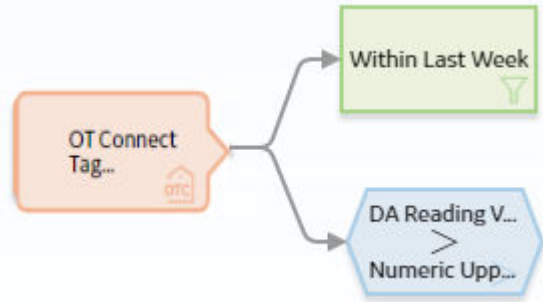
In this example, the OT Connect Tag node is connected to a Collection Filter node and to a Greater Than Condition node.

As shown in the following image, you can use the **Properties** window for the OT Connect Tag node to specify whether the Source Tag requires Historical Readings. In this example, the Require Historical Readings option is selected. This enables access to the Historical Readings output from the OT Connect Tag node in successor nodes.

As shown in the following image, you can use the **Properties** window for the Collection Filter node to select the OT Connect Tag node and the Historical Readings option. When this option is selected, the Collection Filter node evaluates the historical readings retrieved from the process historian and filters the collection to only the readings that have a time stamp within the last seven days.

You can also use the OT Connect Tag node to evaluate the most recent reading associated with an OT Connect Tag record in addition to the values contained in fields of an OT Connect Tag record. As shown in the following image, you can configure the Greater Than node to evaluate whether the most recent reading is greater than the Numeric Upper Level 2 limit value defined for the Policy Instance.

Point Value Nodes in Policy Designer

A Point Value node is an Input node that represents a value that you specify using a policy instance. You can use the Point Value node to represent:

- Constant values that are used more than once in a policy model but are not the same value for each policy instance.
- Values from fields in different families for each policy instance.

The output of a Point Value node is the value that you specify in each policy instance.

Node Properties

The **Properties** window for a Point Value node contains the items that are described in the following table.

Item	Description	Notes
Trigger check box	Specifies whether or not changes to the record that is associated with the node will result in policy execution.	None.
Data Type list	Specifies the type of data that the node represents.	This property is not required, but it is highly recommended to select a value. This minimizes the chance of the Point Value node result either being misinterpreted during the policy execution and the validation, or being displayed in an incorrect format while viewing the execution and the validation results. **Note:** The Data Frame option exists only for use by certain module workflow policies.

Point Value node

The following example illustrates how you can use the Point Value node to represent values from fields in different families for each policy instance. Consider the following nodes and connection.
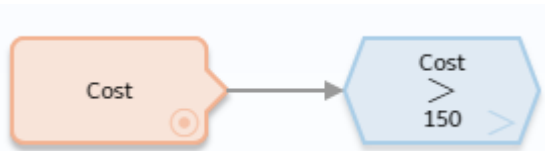
In this example, the Point Value node Cost represents cost values of various types. This model evaluates each cost value to determine whether or not it is greater than 150.

The particular record and field containing each cost value are defined by policy instances. For example, you might create two policy instances, where:

- One policy instance maps the Cost field from the Action family to the Cost Point Value node.
- The other policy instance maps the Total Cost field from the Work History family to the same Cost Point Value node.

By using the Point Value node in this way, the policy can use one Input node to access values in multiple families and evaluate them against the same criteria.

Query Nodes in Policy Designer

A Query node is an Input node that represents a query that is stored in the APM Catalog. You can use a Query node to access the results of a specific query. The query will run each time the policy is executed so that the latest results are used in the policy execution. If the specified query contains prompts, you must use the node's Properties window to identify the values that should be provided to the prompts.

A Query node generates the following outputs:

- Result Set, which represents the full results of the specified query. This output can be used only with successor nodes that are capable of handling collections.
- Any value in the top row of the specified query.

Note:

- The Result Set is restricted to the first 10000 rows of the query results.

Node Properties

The **Properties** window for a Query node contains the items that are described in the following table.

Item	Description	Notes
Query Path	Specifies the path to the query that will run when the policy is executed.	You can select the **switch to field input** ( ) button to select an input from another node. You can select the **show the constant field input** ( ) button to select an output of a predecessor node in this section. **Note:** When the query path is sourced from a predecessor node, the results collection output from the query node can be used only when you do not need to select a specific column in the collection. You can enter the path manually, or you can browse to the query by selecting . The query that you choose must meet the following criteria: • It must have an ID and a caption. • It must be a Select query (that is, not a Crosstab, Update, Append, or Delete query). • Any parameters within the query must contain a unique ID.
Query section	Provides values to any query prompts.	One **Query** section appears for each prompt in the selected query. The label that appears after **Query:** identifies the prompt caption. You can select to specify the output of a predecessor node in this section.

Query node

The following example illustrates how you can use the Query node with other nodes in a policy model to access and analyze Policy Event records using the Events Query in the baseline APM Catalog.

Consider the following nodes and connections.
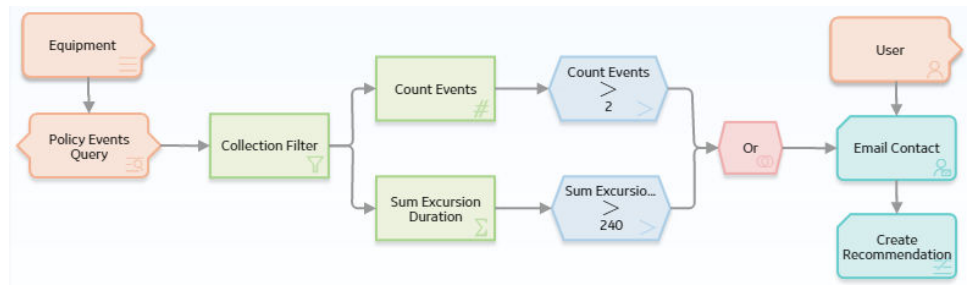
In this example, the Policy Events Query node is connected to the Equipment Entity node and to a Collection Filter node, which is then connected to various additional Condition, Calculation, and Action nodes.

As shown in the following image, you can use the **Properties** window for the Query node to specify the query whose results you want to use in the policy model. You can also see that the specified query contains a prompt, Asset Key, and that a system field from the Equipment Entity node, Entity Key, is used to supply values to the prompt.
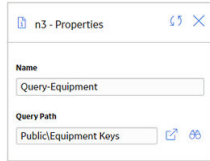
When you connect the Query node to a successor node, in the **Properties** window of the successor node, you can select the Result Set option to evaluate the full query results.

In this example, the Collection Filter node is used to filter the query results. As you can see in the following image of the **Properties** window for the Collection Filter node, the query results are filtered to include only the policy events that lasted longer than 10 seconds, occurred within the last month, and have the name IOW Excursion.
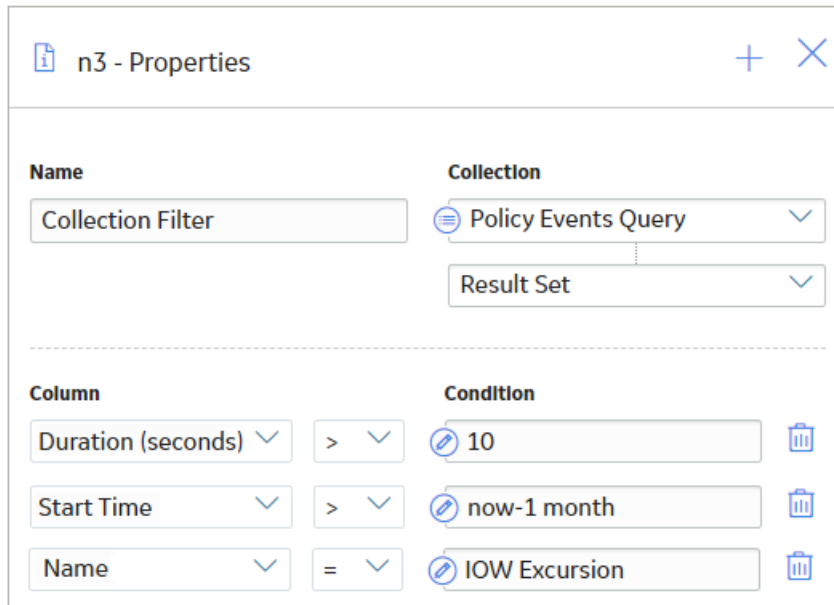
The subsequent Calculation and Condition nodes evaluate the filtered query results and trigger actions when the specified conditions are met. Specifically, the remaining nodes in this policy work together to send an email message and to create a recommendation if the filtered query contains more than two results or if the total duration of the query results exceeds 4 minutes.

Query Entity Nodes in Policy Designer

A Query Entity node is an input node that represents any entity family in APM. You can use the Query Entity node to access information that is stored in a record belonging to an entity family in APM.

Note: The Query Entity node displays only the entity families for which your APM system has active licenses, and for which you have view permissions.

An Entity node generates the following outputs:

- Any field in the entity that the Query Entity node represents.
- The following system fields for the entity that the Query Entity node represents: Entity Key, Entity ID, Family Key, and Site Key.

Node Properties

The **Properties** window for an Entity node contains the items that are described in the following table.

Item	Description	Notes
Entity Key	Specifies the entity key of the record that you want to retrieve.	You can select to specify the output of a predecessor node in this section. A value is required.
Family ID list	Specifies the family that the Query Entity node represents.	The **Family ID** field contains all the APM entity families for which you have the view permission. A value is required.

For an example of using this node, refer to the Family Policies documentation.

User Nodes in Policy Designer

A User node is an Input node that represents the Security User and Human Resource families. You can use a User node to access user information that is stored in records belonging to either of these families.

A User node generates the following outputs:

- Any field in the Security User family or Human Resource family.
- The following system fields in the Security User family: Entity Key, Entity ID, and Family Key.

Note: When you assign records to a User node via policy instances, you will assign Security User records only. The Human Resource records that are associated with each Security User record will be mapped automatically.

Node Properties

The **Properties** window for a User node contains the items that are described in the following table.

Item	Description
Primary Record	Specifies whether the node represents a primary record.
Trigger	Specifies whether changes to the record that is associated with the node will result in policy execution.

User node

The following example illustrates how you can use the User node to access the email address, which exists in the Human Resource family, and the User ID, which exists in the Security User family, for the same person. Consider the following nodes and connections.

147
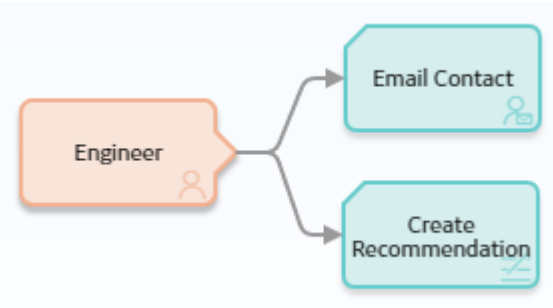
In this example, the Engineer User node is connected to an Email Contact node and a Create Recommendation node.

As shown in the following image, you can use the **Properties** window for the Email Contact node to select the User node and the Email Address field in the Human Resource family.

Additionally, as shown in the following image, you can use the **Properties** window for the Create Recommendation node to select the User node and, in this case, a field in the Security User family, User ID.

Condition, Logic, and Calculation Nodes

Condition, Logic, and Calculation Nodes in Policies

You can use the following nodes to apply a variety of conditions, calculations, and logics to the values represented by Input nodes in the policy model.

Condition Nodes

- Comparison nodes
- Case nodes

Logic Nodes

- And
- Or

Calculation Nodes

The following Calculation nodes perform calculations on single values:

- Add
- Convert Type
- Divide
- Exponent

- Is Null
- JSON Parser
- Math
- Multiply
- R Script
- Remainder
- Round
- Subtract
- Text

The following Calculation nodes perform calculations on a *collection* of data:

- Average
- Collection Filter
- Count
- Last
- Max
- Min
- R Script
- Sum
- Threshold Statistics

Add, Subtract, Multiply, Divide, Exponent, and Remainder Nodes in Policy Designer

The following Calculation nodes represent basic mathematical calculations:

Node	Description
Add	Adds one value to another value.
Subtract	Subtracts one value from another value.
Multiply	Multiplies one value by another value.
Divide	Divides one value by another value.
Exponent	Determines the number of times a value is multiplied by itself.
Remainder	Determines the remainder after a division operation.

Each math node has two inputs. Input requirements differ depending on the type of math node.

- Input for Add, Subtract, Multiply, Divide, and Remainder nodes may be numeric values or certain time-based values.
- Inputs for Exponent nodes must be numeric values.

A math node has only one output. The output of a math node is the result of the mathematical calculation.

Node Properties

The **Properties** window for each math node contains the items that are described in the following table.

Item	Description	Notes
First value	The first input value that will be used in the calculation.	None.
Calculation symbol	The symbol that corresponds with the mathematical calculation that is performed by the node.	None.
Second value	The second input value that will be used in the calculation.	None.
Display list	Determines the label that appears on the node in the policy model.	This list does not appear if you enter a constant in the corresponding section.

Add Node

The following example illustrates how you can use an Add node to add a constant value to a value that is defined by a predecessor node. Consider the following nodes and connection.

In this example, the Add node adds 1 to the value in the Alert Level field of a Health Indicator record. The following image shows the **Properties** window for the Add node.

Remainder and Subtract Nodes

Suppose that you want to add reading values to a Health Indicator, but you know that the accuracy of the reading values is +/-10%. Before adding the reading values to a Health Indicator, use a policy to round down the reading values.

Note: You can use the Round node to round values to the nearest number. This example describes how you can consistently round down to a nearest number.
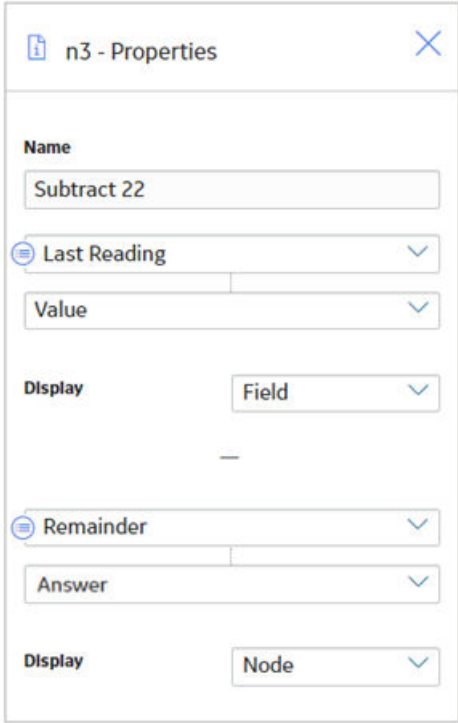
Consider the following nodes and connections. In the policy shown in the following image, the Remainder node is used in conjunction with a Subtract node to round down to the nearest ten the last reading value associated with a Measurement Location.
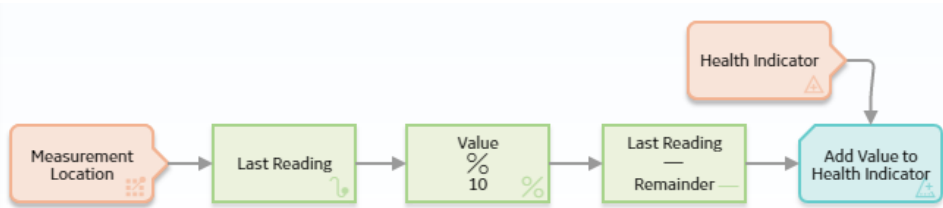
The following image shows the **Properties** window of a Remainder node. You can use the Remainder node to calculate the amount remaining after the Last Reading value is divided by 10. For example, if the last reading value is 87, the result of the remainder node is 7 (87/10=8 with a remainder of 7).
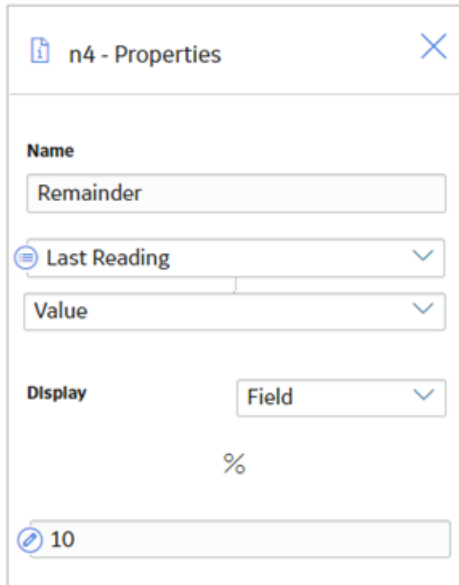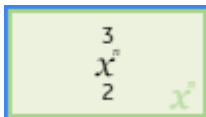
The following image shows the **Properties** window of a Subtract node. You can then subtract the remainder from the reading value. The result of this calculation is the reading value rounded down to the nearest ten. Continuing with the previous example, the result of this node is 80 (87-7=80).

Finally, the result of the Subtract node (80) is added to a Health Indicator Value record that is linked to a Health Indicator.

How Input Values Correspond to Calculations

The following table illustrates how the input values that you define for each node correspond to the mathematical calculations performed by the node. This table includes the mathematical calculation and output for each type of node when the first input value is 3 and the second input value is 2, as shown in the following image:

Node Type	Mathematical Calculation	Output Value
Add	3+2	5
Subtract	3-2	1
Multiply	3x2	6
Divide	3/2	1.5
Exponent	3^2	9
Remainder	3%2	1

Using Time-Based Values

The following tables summarize the possible combinations of time-based input values that you can use with Add, Subtract, Multiply, Divide, and Remainder nodes.

Table 2: Add Node

First Value	Second Value	Output	Example
Time stamp	Time span	Time stamp	1/11/2000 + 10 Days = 1/21/2000
Time stamp	Number	Time stamp	1/11/2000 + 10 = 1/21/2000 00:00:10
Time span	Time stamp	Time stamp	10 Days + 1/11/2000 = 1/21/2000
Time span	Time span	Time span	12 Days + 10 Days = 22 Days
Time span	Number	Time span	12 Days + 10 = 12.00:00:10
Number	Time stamp	Time stamp	10 + 1/11/2000 = 1/21/2000 00:00:10
Number	Time span	Time span	10 + 12 Days = 12.00:00:10
Time stamp	Time stamp	Invalid	N/A

Table 3: Subtract Node

First Value	Second Value	Output	Example
Time stamp	Time span	Time stamp	1/11/2000 - 10 Days = 1/1/2000
Time stamp	Number	Time stamp	1/11/2000 - 10 = 1/10/2000 23:59:50
Time span	Time stamp	Invalid	N/A
Time span	Time span	Time span	12 Days - 10 Days = 2 Days
Time span	Number	Time span	12 Days - 10 = 11.23:59:50
Number	Time span	Time span	10 - 12 Days = 11.23:59:50
Time stamp	Time stamp	Time span	1/11/2000 - 1/1/2000 = 10 Days

Table 4: Multiply Node

First Value	Second Value	Output	Example
Time span	Time span	Invalid	N/A
Time span	Number	Time span	10 Days x 5 = 50 Days
Number	Time span	Time span	5 x 10 Days = 50 Days

Table 5: Divide Node

First Value	Second Value	Output	Example
Time span	Time span	Number	1 day / 8 hours = 3
Time span	Number	Time span	1 day / 2 = 12 hours
Number	Time span	Invalid	N/A

Table 6: Remainder Node

First Value	Second Value	Output	Example
Time span	Time span	Time span	15 Days % 10 Days = 5 Days
Time span	Number	Invalid	N/A
Number	Time span	Invalid	N/A

And and Or Nodes in Policy Designer

And and Or nodes are Logic nodes that you can use in a policy model to specify whether or not policy execution should continue based on the results of the incoming logic paths.

Specifically:

- The And node evaluates whether or not all incoming logic paths result in a value of true passed to the And node.
- The Or node evaluates whether or not at least one incoming logic path results in a value of true passed to the Or node.

    Note: Unlike other nodes which require all immediate predecessor nodes to be executed in order for the node to be, the Or node requires only one immediate predecessor node to be executed in order for the Or node to be executed.

In a policy model, Logic nodes must be preceded immediately by comparison or other Logic nodes. A value of true is passed to the Logic node when the logical result of the preceding node matches the logic path configured for the corresponding connection. For example, if the logical result of an immediately preceding condition node is no and the logic path configured for the corresponding connection is no, a value of true is passed to the Logic node.

The output of a Logic node is the logical result of the node. Specifically, when a Logic node is executed:

- If the Logic node's criteria is met, the output (that is, logical result) of the node will be yes.
- If the Logic node's criteria is not met, the output (that is, logical result) of the node will be no.

The logical results of Logic nodes are used by connections to successor nodes in order to determine if the successor node will be executed. You can use the **Properties** window for a connection starting at a Logic node to configure a logic path for the connection. If you do not configure a logic path for a connection, a Yes path is assumed but does not appear on the model. The APM system will execute only the branches of a policy model where the logical result of the Logic node matches the logic path defined for the corresponding connection.

More Information: Logical Results

The following table summarizes what the result of each Logic node will be for various input combinations.

Input A	Input B	And Node Result	Or Node Result
True	True	Yes	Yes
False	False	No	No
True	False	No	Yes

Note: For Or nodes, any input value that is not true is considered false. This means that if a preceding node is not executed or if errors occur during execution, the input from the corresponding path will be false.

And Node

The following example illustrates how you can use the And node to monitor policy execution. To simplify this example, only constant values are used in the policy model. Consider the following nodes and connections, which are shown after validation has been run.
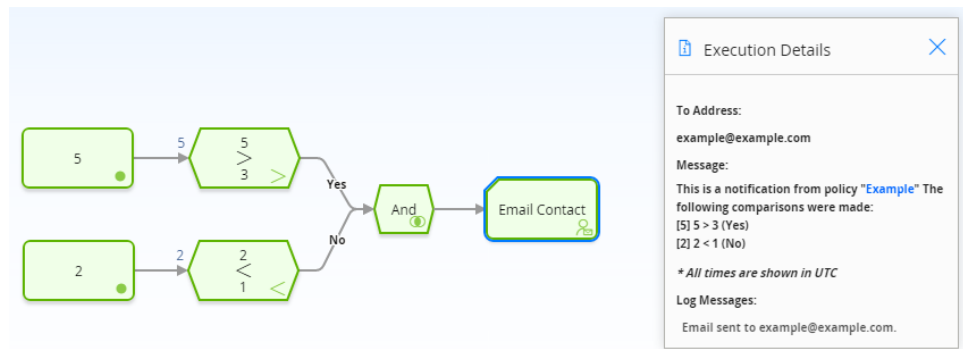
In this example, you can see that each node executed successfully. The logical result of the And node is yes because all incoming logic paths result in a value of true passed to the And node (that is, the logical result of each preceding condition node matches the logic path of the corresponding connection). Therefore, policy execution continues past the And node.

Or Node

The following example illustrates how you can use the Or node to monitor policy execution. To simplify this example, only constant values are used in the policy model. Consider the following nodes and connections, which are shown after validation has been run.
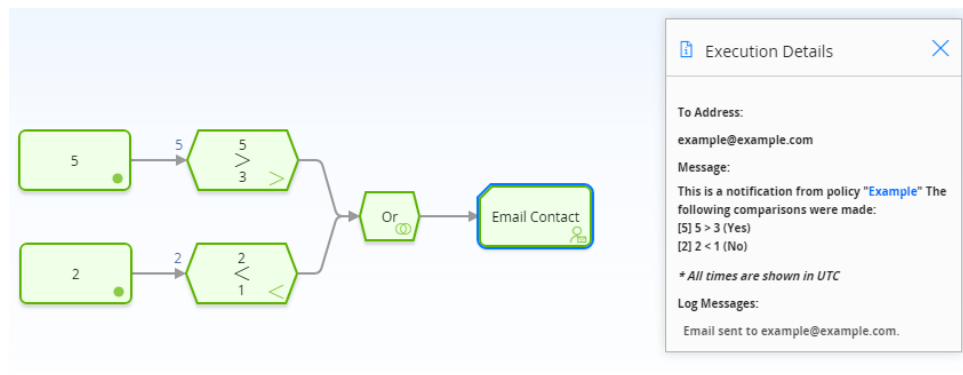
In this example, you can see that each node executed successfully. The logical result of the Or node is yes because the logical result of at least one incoming logic path results in a value of true passed to the Or node (that is, the result of the 5 > 3 Condition node is yes, which matches the logic path of the corresponding connection). Therefore, policy execution continues past the Or node.

Average Nodes in Policy Designer

An Average node is a Calculation node that you can use in a policy model to calculate the average value of data in a specified column of a collection.

The input for an Average node must be a *collection* with a column containing numeric or time-based values. The output of an Average node, Value, contains the average value of data in the specified column.

Node Properties

The **Properties** window for an Average node contains the items that are described in the following table.

Item	Description	Notes
Collection section	Specifies the collection that contains the values that you want to average.	You can select ✏ to specify the output of a predecessor node in this section.
Collection Column section	Specifies the column that contains that values that you want to average.	This list contains the columns that are available in the selected collection. The column that you select must contain numeric or time-based values.

Average node

The following example illustrates how you can use an Average node to evaluate the readings that are associated with a Health Indicator record and return the average reading value. Consider the following nodes and connection.
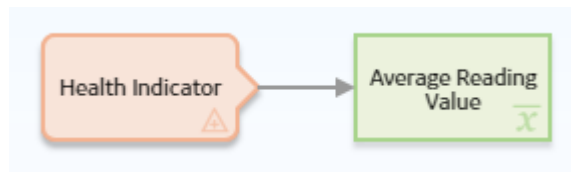
As shown in the following image, you can use the **Properties** window for the Average node to select the Health Indicator node and the Readings option. The Value column of the Readings collection is also selected, indicating that the output of the node is the average of the values in the Value column.
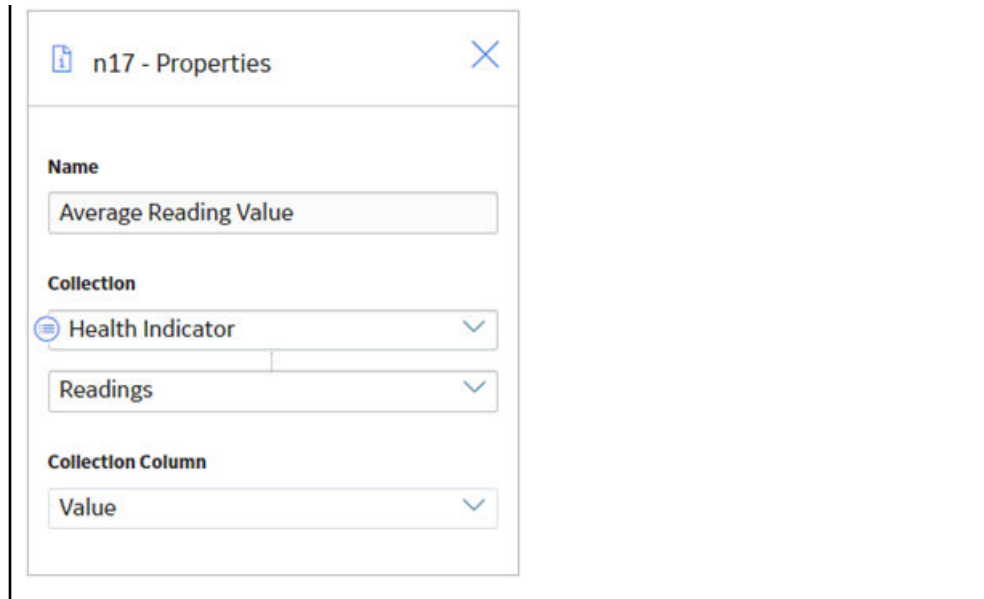
Case Nodes in Policy Designer

A Case node is a Condition node that you can use in the policy model to set up scenarios in which the output values of the node should be changed automatically based on specific input values. Throughout this documentation, we refer to each defined scenario as a Case. Each Case within the Case node has an input value and one or more output values that the APM system will use if the value in a defined input field matches the input value of the Case.

There are two types of Cases in the Case node, which we refer to as the If Case and Else Case throughout this documentation:

If Case

The type of Case that is executed if the value in a defined input field matches a specific input value that you specify. You can define one or more If Cases for a given Case node.

Else Case

The Case that is executed by default if the value in a defined input field does not match a specific input value that you specified in an If Case. There is only one Else Case for a given Case node.

You can use the Case node to:

- Change the output value of a node from one value to another.
- Combine policy logic after a Condition node in the policy model.

The input of a Case node must be a single value or the logical result of a comparison node. The output of a Case node is the value that you define in the **Value** column corresponding to the output option that you select in a successor node. Outputs may be single values or collections.

Node Properties

The **Properties** window for a Case node contains the items that are described in the following table.

| Item	Description	Notes
Input section	Specifies the field whose value you want to compare to each **If Input =** value in order to determine the output(s) of the Case node.	If you select a comparison Condition node as a predecessor node, the logical result of the condition will be used automatically as the input value.
If Case section		
If Input = text box	Specifies the value that must match the value that is defined in the Input section in order for the corresponding If Case to be executed.	You can use the ╋ and 🗑 buttons in the **If Input =** row to add or delete If Cases in the Case node.
Output section	Specifies the output(s) for the corresponding If Case. The output section contains two columns: • **Mapping**: Specifies a name for each output that you define. • **Value**: Specifies the value or collection that will be the output of the Case node if the corresponding Case is executed.	You can use the ╋ and 🗑 buttons in the output section to add or delete output rows. When you add or delete an output row in one Case, corresponding output rows in all cases are added or deleted automatically.
Else Case section		
Else output section	Specifies the output(s) of the Else Case. The Else Case will be executed if the value in the **If Input =** section does not match any value defined in an **If Input =** text box.	This output section contains the same functionality as described above. You cannot delete the Else Case from the Case node.

Change the Output Value of a Node from One Value to Another

You can use the Case node in the policy model to change a node's output values so that they can be used in successor nodes to perform calculations or actions. For example, you can use the Case node to convert character-based values from a Health Indicator reading or an AMS Asset status message to numeric values that you can use in subsequent calculations.

Consider the following policy model in which reading values from a character-based Health Indicator record are converted to numeric values. The values are then added to a Health Indicator record that uses numeric values.
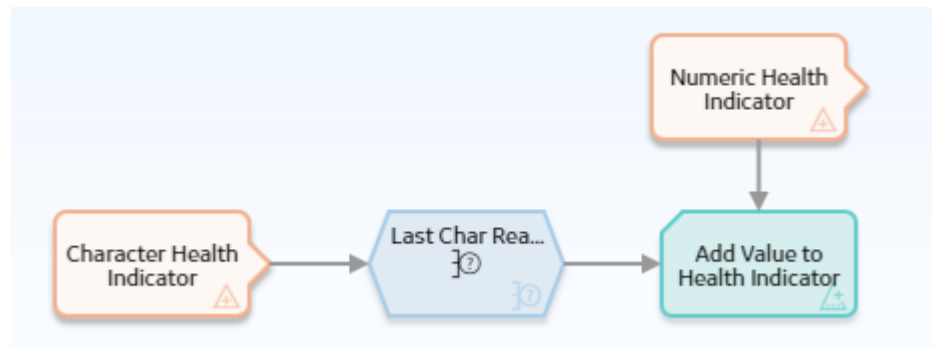
As shown in the following image, you can use the **Properties** window of the Case node to indicate that the output value of the Case node will differ depending on the reading values associated with the Character Health Indicator node.

In this example, one output, HI Value, is defined. For this output, if the value in the Last Char Reading Value field is:

- 1-OK, the output value of the Case node will be 100. This is an If Case.
- 2-Not OK, the output value will be 50. This an another If Case.
- Any other value, the output value will be 0. This is the Else Case.

These values represent the numeric equivalents of the character-based readings associated with the Character Health Indicator node.

As shown in the following image of the **Properties** window of the Add Value to Health Indicator node, you can add the converted values to a numeric-based health indicator by selecting the HI Value output from the Case node.

Combine Policy Logic after a Condition Node in the Policy Model

You can use the Case node to combine policy logic in situations when a Condition node's successor nodes are the same in each logic branch. Rather than creating separate logic paths with duplicate nodes, you can use a Case node to define different outputs for each logical result of the condition. This is especially useful in large policy models with multiple logic branches and duplicate nodes.

The following image shows what a policy that uses Condition nodes looks like without Case nodes in the model.
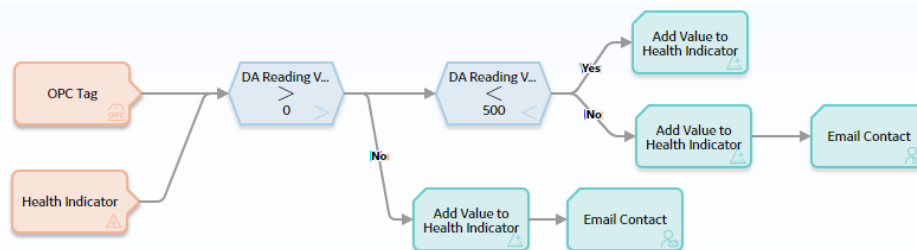
Notice that the Add Value to Health Indicator node appears three times, and the Email Contact node appears twice.

The following image shows what the same policy looks like when you use Case nodes in the model.
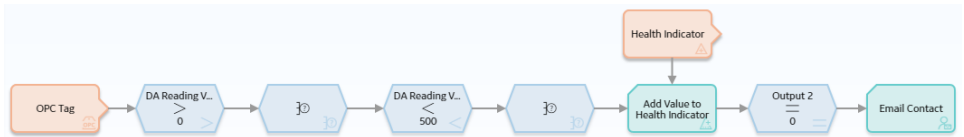
In this model, each Case node takes the place of a logic branch. Rather than splitting the model into separate logic paths, the Case node supplies a specified output corresponding to the logical result of the preceding Condition node. Notice that in this model the Add Value to Health Indicator node and the Email Contact node appear only once.

The following image shows what the **Properties** window looks like for the first Case node in the preceding image.

In this example, if the logical result of the Condition node named Greater Than is:

- Yes, the output of the node will be 50.
- No, the output will be 0.

When Output 1 is selected as the input of a successor node, the APM system will supply the corresponding value to the successor node.

Collection Filter Nodes in Policy Designer

A Collection Filter node is a Calculation node that you can use to apply one or more filters to collections that are represented by nodes in the policy model. Filters may, for example, specify a range of values or dates within which readings must fall in order to be evaluated by successor nodes.

The input of a Collection Filter node must be a collection of data. The output of a Collection Filter node, Filtered Collection, includes only the rows in a collection that pass all filter criteria that are defined in the Collection Filter node's **Properties** window.
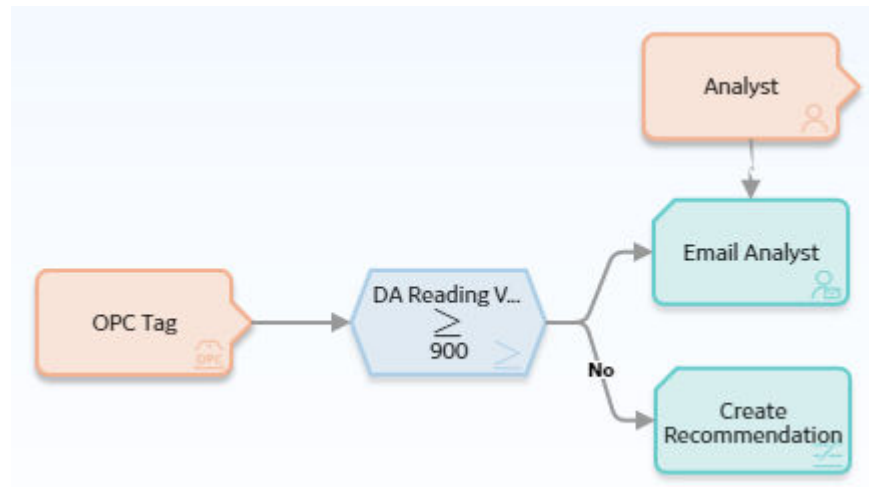
Node Properties

The **Properties** window for a Collection Filter node contains the items that are described in the following table.

Item	Description	Notes
Collection section	Specifies the *collection* that you want to filter.	You can select ✎ to specify the output of a predecessor node in this section.
+ button	Adds a new filter row to the Collection Filter node.	Each filter row consists of a column, operator, condition, and the 🗑 button.
Column list	Specifies the column whose values you want to use in the filter.	This list contains the columns that are available in the selected collection.
Operator list	Specifies the comparison operator that you want to apply to the values in the selected column.	This list contains the following operators: • Greater than (>) • Greater than or equal (>=) • Less than (<) • Less than or equal (<=) • Equal (=) • Does not equal (!=) • Starts with • Contains • Ends with
Condition value	The value that will be compared against the values in the corresponding column to determine the output of the node.	You can select ✎ to specify the output of a predecessor node in this section.
🗑 button	Deletes the corresponding filter row from the Collection Filter node.	None

Collection Filter node

The following example illustrates how you can use the Collection Filter node to filter query results such that only certain results are used in subsequent calculations. Consider the following nodes and connection.
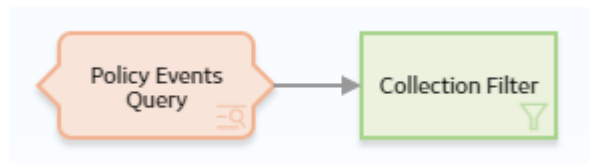
In this example, a Query node, Policy Events Query, is connected to a Collection Filter node.

As shown in the following image, you can use the **Properties** window of the Collection Filter node to define multiple filters that are applied to the full table of results from the query. Specifically, the query results are filtered to include only the policy events that lasted longer than ten seconds, occurred within the last month, and have the name IOW Excursion.

The output of the Collection Filter node (the filtered results of the query), can then be used by successor nodes to perform a variety of calculations.

Comparison Nodes in Policy Designer

Comparison nodes are Condition nodes that you can use to compare two input values using the comparison operator that corresponds to the name of the node. The following comparison nodes are available:

- Equal
- Not Equal
- Greater Than
- Greater Than or Equal
- Less Than
- Less Than or Equal

Each comparison node requires two inputs, which must be single values. For Equal and Not Equal nodes, inputs can be any type of data. For the remaining comparison nodes, inputs must be numeric or time-based values.

The output of a comparison node is the logical result of the comparison (that is, yes or no). The output of a comparison node can be used only as an input to Case or Logic nodes. For all other successor nodes, the output is used by the connection to the successor node in order to determine if the successor node will be executed.

You can use the **Properties** window for a connection starting at a comparison node to configure a logic path for the connection. If you do not configure a logic path for a connection, a Yes path is assumed but does not appear on the model. The APM system will execute only the branches of a policy model where the logical result of the comparison matches the logic path defined for the corresponding connection.

Node Properties

The **Properties** window for each comparison node contains the items that are described in the following table.

Item	Description	Notes
First value	The input value that will be compared to the second input value.	You can select to specify the output of a predecessor node in this section.
Operator symbol	The symbol that corresponds with the comparison operation that is performed by the node.	None.
Second value	The input value that will be compared to the first input value.	You can select to specify the output of a predecessor node in this section.
Display list	Determines the label that appears on the node in the policy model.	This list does not appear if you enter a constant in the corresponding section.

Comparison nodes

The following example illustrates how you can use a comparison node to compare the value in a field of a predecessor node to a constant value. Consider the following nodes and connections.

The following image shows what the **Properties** window looks like for the Greater Than or Equal To node.

Based on the connection properties, the following logic will be applied when the policy is executed:

- If the DA Reading Value is greater than or equal to 900, an email message will be sent. This logic is determined by the connection between the Greater Than or Equal To Condition node and the Email Analyst node. Because a logic path is not defined, a value of Yes is assumed but does not appear on the model.
- If the DA Reading Value is not greater than or equal to 900, a Policy Recommendation record will be created. This logic is determined by the No logic path specified for the connection to the Create Recommendation node.

Convert Type Nodes in Policy Designer

You can use a Convert Type node to convert a single value, or one column in a collection of values, to another data type. The node converts the value or column of values to the corresponding values of the specified data type, and provides the converted value as its output. When the Convert Type node is configured to operate on a collection column, the output collection includes all the columns of the collection, not only the column of converted values.

You can use the Convert Type node to convert a value to one of the following data types:

- Boolean
- Time & Date
- Decimal
- Integer
- String
- Time Span

Node Properties

The **Properties** window for a Convert Type node contains the items described in the following table.

Field	Description	Note
Output Type	Specifies the data type to which you want to convert the value.	None.
Input Value	Specifies the value that you want to convert.	If the data type of the input value is such that it cannot be converted to the data type specified in the **Output Type** field, the node will not be executed during the execution of the policy.

How Values Are Converted

The following table describes how the input values of different data types are converted by the Convert Type node.

| Data Type of Input Value	Data Type Specified for Output Value					
	Boolean	**Decimal**	**Integer**	**String**	**Time & Date**	**Time Span**
Boolean	Retained without conversion.	• If the value is True, it is converted to 1 • If the value is False, it is converted to 0	• If the value is True, it is converted to 1 • If the value is False, it is converted to 0	Converted to the string representation of the value	Cannot be converted, and results in execution failure of the node	Cannot be converted, and results in execution failure of the node
Dataframe	Cannot be converted, and results in execution failure of the node	Cannot be converted, and results in execution failure of the node	Cannot be converted, and results in execution failure of the node	Converted to the string representation of the value	Cannot be converted, and results in execution failure of the node	Cannot be converted, and results in execution failure of the node
Time & Date	Cannot be converted, and results in execution failure of the node	Converted to the number of seconds elapsed between January 1, 1970 and the value	Converted to the number of seconds elapsed between January 1, 1970 and the value	Converted to the string representation of the value	Retained without conversion	Cannot be converted, and results in execution failure of the node
Decimal	• If the value is 0, it is converted to False • Any value other than 0 is converted to True	Retained without conversion	Rounded to the nearest whole number	Converted to the string representation of the value	Converted to date after considering the value as the number of seconds since January 1, 1970	Converted to represent number of seconds

Data Type of Input Value	Data Type Specified for Output Value					
	Boolean	**Decimal**	**Integer**	**String**	**Time & Date**	**Time Span**
GUID	Cannot be converted, and results in execution failure of the node	Cannot be converted, and results in execution failure of the node	Cannot be converted, and results in execution failure of the node	Converted to the string representation of the value	Cannot be converted, and results in execution failure of the node	Cannot be converted, and results in execution failure of the node
Integer	• If the value is 0, it is converted to False • Any value other than 0 is converted to True	Retained without conversion	Retained without conversion	Converted to the string representation of the value	Converted to the corresponding date after considering the value as the number of seconds elapsed since January 1, 1970	Converted to represent number of seconds

Data Type of Input Value	Data Type Specified for Output Value					
	Boolean	**Decimal**	**Integer**	**String**	**Time & Date**	**Time Span**
String	• If the value is Yes, True, No, or False, it is converted to the corresponding Boolean value. • Other values are first converted to the corresponding numeric values. If the numeric value is 0, the value is finally converted to False. Otherwise, the numeric value is converted to True.	Converted to the corresponding numeric value and then rounded to the nearest whole number	Converted to the corresponding numeric value	Retained without conversion	• If the value clearly indicates a date (for example, the strings *today* and *now-3d*), it is converted to the corresponding date • If the value is a numeric value, it is converted to a date after considering the value as the number of seconds elapsed since January 1, 1970 • Other values Cannot be converted, and result in execution failure of the node	• If the value clearly indicates a time span (for example, the strings *3d* and *3 days*), it is converted to the corresponding time span value • If the value is a numeric value, it is converted to represent number of seconds • Other values Cannot be converted, and result in execution failure of the node
Time Span	Cannot be converted, and results in execution failure of the node	Converted to the number of seconds represented by the value	Converted to the number of seconds represented by the value	Converted to the string representation of the value	Cannot be converted, and results in execution failure of the node	Retained without conversion

Convert Type node

The following example illustrates how the Convert Type node can be used to convert the text string output of a Text node to a time span value.
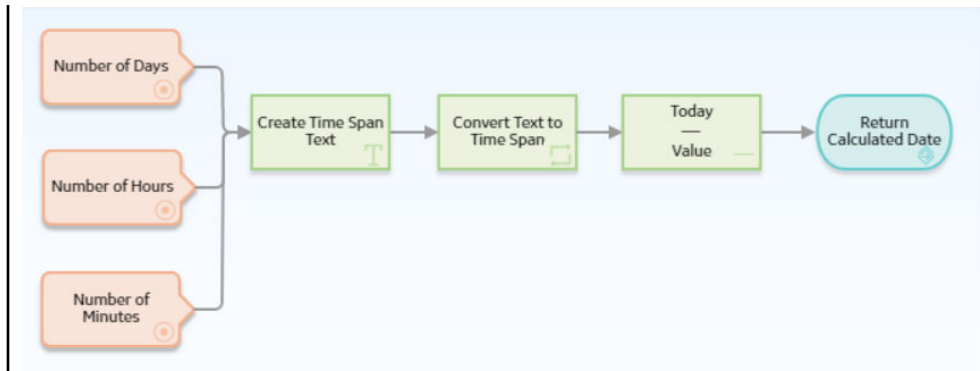
In this example, the Text node is configured to create a text string that indicates a time span, using the values represented by the Point Value nodes. Because the Subtract node cannot perform any calculation on a text string, the Convert Type node converts the string to a value of the Time Span data type. The converted time span value is then used in the Subtract node to calculate the required date.

The following image illustrates the **Properties** window of the Convert Type node described in this example.
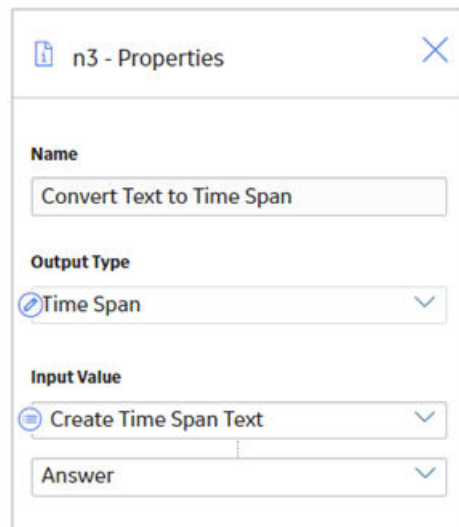
Count Nodes in Policy Designer

A Count node is a Calculation node that you can use in a policy model to calculate the total number of rows in a *collection*.

The input for a Count node must be a collection. The output for a Count node, Value, contains the result of the calculation.

Node Properties

The **Properties** window for a Count node contains the items that are described in the following table.

Item	Description	Notes
Collection section	Specifies the collection for which you want to calculate the total number of rows.	You can select to specify the output of a predecessor node in this section.

Count node

The following example illustrates how you can use a Count node in the policy model to determine if any Measurement Location readings were recorded within the last week. Consider the following nodes and connections.
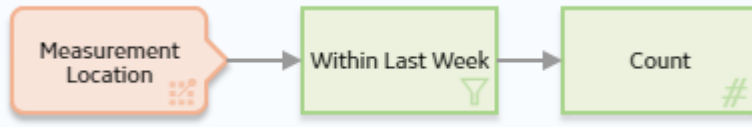
In this example, a Collection Filter node is used to filter the readings associated with the Measurement Location node to only the readings that occurred within the last week. The Count node is then used to calculate the number of readings that occurred. The following image shows what the **Properties** window for the Count node looks like.
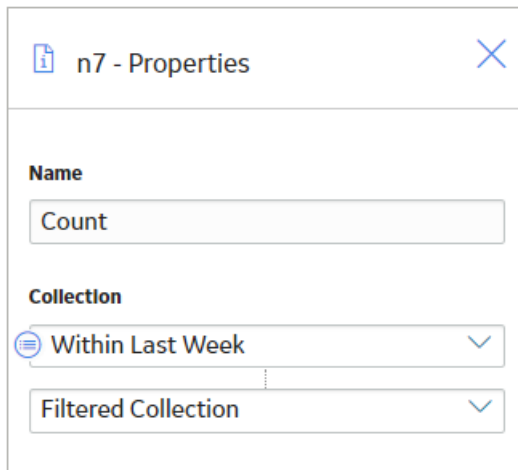
Is Null Nodes in Policy Designer

An Is Null node is a Calculation node that you can use to specify a default value that should be used in subsequent calculations in the event that the input value is null. You can also use this node in combination with a logic node to specify an action to be taken in the event that an input value is null.

The input of an Is Null node must be a single value. The output of an Is Null node, Answer represents either of the following two values:

- If the input value is not null, Answer, represents the same input value.
- If the input value is null, Answer represents the value that you specify in the **Properties** window.

Node Properties

The **Properties** window for an Is Null node contains the items that are described in the following table.

Item	Description	Notes
Input Value section	Specifies the value that you want to determine whether it is null.	You can select ✎ to specify the output of a predecessor node in this section.
Value if Input is Null section	Specifies the output value of the node if the value specified in the **Input Value** section is null.	You can select ✎ to specify the output of a predecessor node in this section.

Assign a default value if a value is null

The following example illustrates how you can use the Is Null node in a policy model to define a default value to be used in subsequent nodes in the scenario where the input value is null.

Consider the following nodes and connections.

As shown in the following image, you can use the Is Null node to check whether or not the Criticality field in the record associated with the AMS Asset node has a value. If there is no value, the Is Null node output will be 0.

Then, as shown in the following image of the **Properties** window for the Case node, you can now use the output of the Is Null node to map the numeric Criticality values to corresponding character values (HIGH, MEDIUM, and LOW), which are used to populate the Recommendation Priority field when the Create Recommendation node is executed.

Using the Is Null node in this way allows you to translate a null value to an actual value that can be used in subsequent calculations and actions.

Trigger an action if a field is null

The following example illustrates how you can use the Is Null node in a policy model to trigger some action if a particular field is null.

Consider the following nodes and connections.

As shown in the following image, you can use the Is Null node to check whether or not the Criticality field in the record associated with the AMS Asset node has a value. If there is no value, the Is Null node output will be -1. The value -1 was chosen in this case because it is a value that is invalid for criticality; therefore, it serves as an indication that the Criticality field does not contain a value.

The Equal condition node in the policy evaluates whether the value from the Is Null node is equal to -1. If it is, this indicates that the Criticality field contains no value and an email alert is sent.

JSON Parser Nodes

A JSON Parser node is a Calculation node that you can use in a policy to transform a JSON object, such as the output from an API node, so that the results can be further evaluated by the policy. You can use the node to transform a JSON object to a collection with column names that you specify, which can then be evaluated by policy nodes designed to operate on collections.

The inputs for a JSON Parser node are a single JSON object and one or more JSON Path query expressions.

The JSON Parser node can operate in two different output modes:

- In the default mode, the outputs from the node include a Collection containing columns with names that you define, and a set of single value outputs. In this mode, all the JSON Path expressions you specify must return lists of values of equal length.
- In the alternate mode, only the single value outputs are provided. In this mode, the JSON Path query expressions are not required to return equal-length lists. To activate the alternate mode, in the **Remove output collection?** box, select **Yes**.

Table 7: Node Properties

Item	Description	Notes
JSON section	Specifies the JSON object that you want to parse.	You can select to specify the output of a predecessor node in this section.
Remove output collection?	Specifies the output mode that you want to use for the node.	By default, this option is set to No.

Item	Description	Notes
➕ button	Adds a new row to the JSON Parser node.	Each row consists of a Name, JSON Path and the 🗑 button.
Name section	Specifies the Name of an output value and/or column.	
JSON Path section	Specifies a JSON Path expression you want to use to query the JSON input.	You can select ✏ to specify the output of a predecessor node in this section. Refer to About Querying using JSON Path Expressions for more information.
🗑 button	Deletes the corresponding row from the JSON Parser node	

For an example showing the use of the JSON Parser node, refer to API Action Nodes in Policy Designer.

About querying using JSON Path expressions

The JSON Parser node uses JSON Path expressions to query the input JSON object.

Note: JSON Path expression queries may return a simple list of values, or a more complex JSON object. If you are using the JSON Parser node to return a collection, all the JSON Path expressions you use should be designed to return lists of values of the same length.

The following JSON Path operators are supported:

JSON Path Operator	Description
$	Root object/element
@	Current object/element
. or []	Child operator
..	Recursive descent
*	Wildcard
[]	Subscript operator
[,]	Union operator
[start:end:step]	Array slice operator
?()	Filter using (script) expression.

The following JSON object is used to demonstrate how a JSON Path query expression operates on a JSON object in the following table of examples:

```
{ "store": {
    "book": [
      { "category": "reference",
        "author": "Nigel Rees",
        "title": "Sayings of the Century",
        "price": 8.95
      },
      { "category": "fiction",
        "author": "Evelyn Waugh",
```

```
            "title": "Sword of Honour",
            "price": 12.99
          },
          { "category": "fiction",
            "author": "Herman Melville",
            "title": "Moby Dick",
            "isbn": "0-553-21311-3",
            "price": 8.99
          },
          { "category": "fiction",
            "author": "J. R. R. Tolkien",
            "title": "The Lord of the Rings",
            "isbn": "0-395-19395-8",
            "price": 22.99
          }
        ],
        "bicycle": {
          "color": "red",
          "price": 19.95
        }
      }
    }
```

Required Output	JSON Path Example	Result
The authors of all books in the store	$.store.book[*].author	["Nigel Rees","Evelyn Waugh","Herman Melville","J. R. R. Tolkien"]
All authors	$..author	["Nigel Rees","Evelyn Waugh","Herman Melville","J. R. R. Tolkien"]
All things in the store, which are some books and a red bicycle	$.store.*	[[{"category":"reference","author":"Nigel Rees","title":"Sayings of the Century","price":8.95}, {"category":"fiction","author":"Evelyn Waugh","title":"Sword of Honour","price":12.99}, {"category":"fiction","author":"Herman Melville","title":"Moby Dick","isbn":"0-553-21311-3","price":8.99}, {"category":"fiction","author":"J. R. R. Tolkien","title":"The Lord of the Rings","isbn":"0-395-19395-8","price":22.99}],{"color":"red","price":19.95}]
The price of everything in the store	$.store..price	[8.95,12.99,8.99,22.99,19.95]
The third book **Note:** The first item in an array has index 0, so the third item has index 2.	$..book[2]	{ "category": "fiction", "author": "Herman Melville", "title": "Moby Dick", "isbn": "0-553-21311-3", "price": 8.99 }
The last book **Note:** The last item in the array is accessed by specifying -1 as the start index.	$..book[-1:]	{ "category": "fiction", "author": "J. R. R. Tolkien", "title": "The Lord of the Rings", "isbn": "0-395-19395-8", "price": 22.99 }

Required Output	JSON Path Example	Result
The first two books	$..book[0,1] -or- $..book[:2]	[{"category":"reference","author":"Nigel Rees","title":"Sayings of the Century","price":8.95}, {"category":"fiction","author":"Evelyn Waugh","title":"Sword of Honour","price":12.99}]
All books with an ISBN number	$..book[?(@.isbn)]	[{"category":"fiction","author":"Herman Melville","title":"Moby Dick","isbn":"0-553-21311-3","price":8.99}, {"category":"fiction","author":"J. R. R. Tolkien","title":"The Lord of the Rings","isbn":"0-395-19395-8","price":22.99}]
All books cheaper than 10	$..book[?(@.price<10)]	[{"category":"reference","author":"Nigel Rees","title":"Sayings of the Century","price":8.95}, {"category":"fiction","author":"Herman Melville","title":"Moby Dick","isbn":"0-553-21311-3","price":8.99}]
All members of JSON structure	$..*	The entire input JSON object shown above is returned.

Last Nodes in Policy Designer

A Last node is a Calculation node that you can use in a policy model to retrieve the last row in a collection, whose fields you can then use in successor nodes to perform various calculations or actions.

Note: For reading collections that are associated with Measurement Location, Health Indicator, or OPC Tag nodes, the last row is the most recent reading. For collections that are associated with Query nodes, the last row depends on the order in which the query is sorted.

The input for a Last node must be a *collection*. The output for the Last node is any field in the row that the Last node retrieves.

Node Properties

The Properties window for a Last node contains the items that are described in the following table.

Item	Description	Notes
Collection section	Specifies the collection from which you want to retrieve the last row.	You can select ✎ to specify the output of a predecessor node in this section.

Last node

The following example illustrates how you can use a Last node to retrieve the most recent reading associated with a Measurement Location record and use this value in a subsequent calculation. Consider the following nodes and connections.
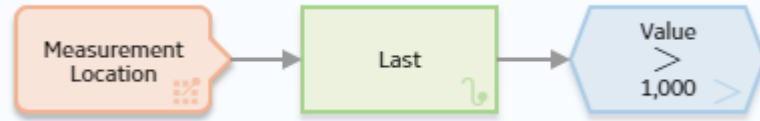
As shown in the following image, you can use the **Properties** window for the Last node to select the Measurement Location node and the Readings option. When this option is selected, the Last node evaluates the readings that are associated with the Measurement Location record and returns the last row in the collection, which contains the most recent reading.
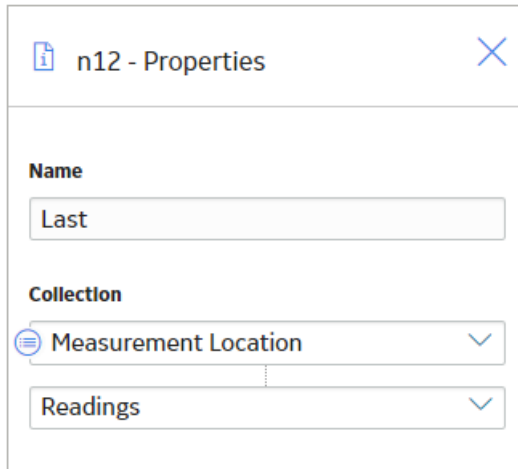
You can use the most recent reading in various subsequent calculations. In this example, the Greater Than node evaluates the value of the most recent reading to determine whether or not it is greater than 1,000.

Math Node

A Math node is a Calculation node that you can use in the policy model to perform mathematical operations on numeric input values. The node performs operations on the values represented by variables that you define for the node. It evaluates the expression that you provide (for example, a^2+b*c) based on the input values, mathematical functions, and operators used. The resulting value of the expression is returned as the output of the Math node, which can be used in another node in the policy model. The variables used in the mathematical expression defined for a Math node can be configured to represent a single value or a collection of numeric values.

Note: User-defined arguments are currently not supported in the Math node.

Comparison with R Script Node

For moderately complex calculations on numeric values, a Math node offers a simpler alternative to the R Script node in that it does not require the knowledge of the R programming language, and it runs faster than the R Script node.

Node Properties

The **Properties** window for a Math node contains items that are described in the following table.

Item	Description	Note
Math expression	The mathematical expression to be evaluated.	• For information on the supported functions and expression syntax, refer to https://github.com/mariuszgromada/MathParser.org-mXparser/blob/master/README.md documentation. • If the expression is invalid, a message appears in the notification bar and you cannot activate the policy. • If the output from a predecessor node that is used as an input to a Math node is an invalid expression, the warning messages will not appear in the notification bar. The associated error occurs during policy validation or execution.
Name	The variable name that you may have used in the Math expression.	• You must define each variable name used in the math expression. • You may define variable names that are not used in the math expression. • The variable name is case-sensitive and must contain only alphanumeric characters without spaces. • You may use the reserved name of a math operation or constant value (for example, cos, pi, Beta, etc.) as a variable name, in which case it will be interpreted as an input variable, instead of the given math operation or value. • If the named variable represents a collection of numeric values, you can reference the individual values in the collection in the math expression by adding a number to the end of the variable name. For example, if the variable is named **List** the individual members of a collection of 3 input values can be referenced in the math expression as **List1**, **List2**, and **List3**.
Value	The value of the named variable.	This must be a numeric value or collection of numeric values.
+	Adds a row for Name and Value.	None
🗑	Deletes the corresponding row.	None.

How Input Values Correspond to Calculations

The following table describes how the input values that you define for a Math node correspond to the mathematical operations performed by the node. Here, 'a' and 'b' are variables whose values are 5 and 8, respectively.

Math expression	Output	Note
2+pi	5.1415926536	pi is a mathematical constant.
a!+b	128	This is a factorial operator.
a=b^2	0, if false; 1, if true.	^ is an exponentiation operator.
if(a>b,100,0)	0, if false; 100, if true.	> is a relational operator.
sqrt(a+b)	3.6055512755	sqrt is a square root operator.

How Input Values Correspond to Calculations using Logical Operators

The following table describes how the input values that you define for a Math node correspond to the mathematical operations performed by the node when the mathematical expression contains logical operators. Here, A, B, and C are variables whose values are 1, 0, and 0 respectively.

Important: The order of precedence of the logical operators (namely AND, Or, and Not) during the evaluation of mathematical expressions has been modified in the Mathparser.org (V4.4.2) library. This can cause unexpected results in your policies. For example, a Math node configured with the mathematical expression A|B&C will now be evaluated as A|(B&C), whereas it was previously evaluated as (A|B)&C.

Math expression	Output	Note
A | B & C	1 (that is, True)	The And (&) operator takes precedence over the Or (|) operator
(A|B) & C	0 (that is, False)	None
A | (B&C)	1 (that is, True)	None

How A Collection of Input Values Represented by a Single Variable Correspond to Calculations

The following table describes how a collection of input values that are represented by a single variable defined for a Math node corresponds to the mathematical operations performed by the node. Here, 'a' is a variable that represents a collection of values 10, 20, 30, and 40, and 'b' is a variable that represents a collection of values 100, 110, 120, and 130.

Math expression	Output	Note
add(a)	100	add is a summation operator.
mean(a,b)	70	mean is an average value operator.
a1+b1	110	You can reference individual values in the collection by adding a number to the end of the variable name.

Math node

The following example illustrates how you can use the Math node to evaluate a Math expression.

Consider the following policy model that is designed to provide an email alert when an equipment has exhausted the threshold percentage of its life expectancy.
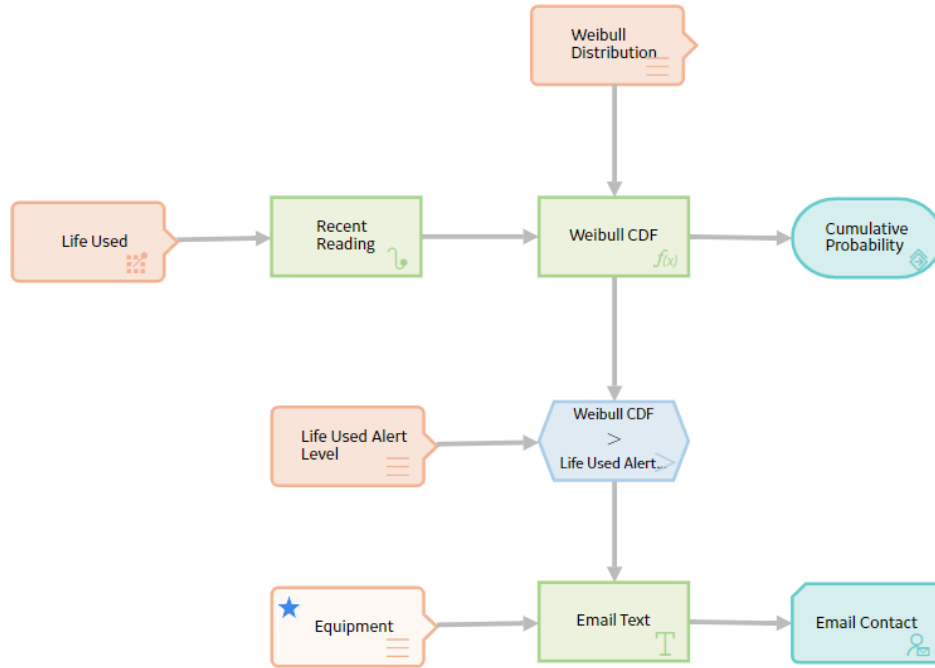
The threshold percentage of life used for the equipment is stored as a Technical Characteristic. The life data of the equipment (for example, hours for turbine, starts for crane, take-offs for aircraft, tonnes moved for mining equipment), is stored as Measurement Location readings.

The percentage of expected life of the equipment is calculated in the Math node using the Cumulative Distribution Function (CDF) of the Weibull Distribution, which is a mathematical function for analyzing life data.

The Math expression used for this calculation is as follows:

100*(1-e^(-((Time/Eta)^Beta)))

where:

- e is a mathematical constant.
- Time is the life used.
- Eta and Beta are Weibull distribution parameters.

Note: Beta is a defined math function name, however, in this example, it is used as the name of an input variable.

The following image shows the **Properties** window of the Math node:

To easily identify the Measurement Location, Technical Characteristic, and Weibull Distribution parameters associated with the equipment, the Equipment Entity input node is specified as the primary node, indicating that the record specified for this node is the primary record to which all the other input records (represented by other Input nodes) will be linked.

The Weibull Distribution Entity input node provides the distribution parameters (that is, Eta and Beta), which were determined in the Reliability Analytics module by analyzing the failure data of a similar equipment.

The Life Used Measurement Location input node provides the readings to the Recent Reading Last calculation node, which retrieves the latest measurement reading (that is, Time).

The resulting value of the Math expression is passed to the Cumulative Probability Return Value action node, which will include the value in the policy execution history.

The Life Used Alert Level Entity input node provides the Technical Characteristic value to a comparison condition node, which compares it with the output of the Weibull CDF Math calculation node.

When the Weibull CDF value exceeds the Technical Characteristic value (that is, the threshold percentage) of the equipment, the message specified in the Email Text calculation node is sent via email to the user specified in the Email Contact action node.

The following image shows the **Execution Details** window of the Math node.

Execution Details

Answer:

98.168436111

Log Messages:

Variable Eta = 3

Variable Beta = 2

Variable Time = 6

Calculating result of 100*(1- e^(- ((Time/Eta)^Beta)))

Note: You can also build this policy model without a Math node, in which case you will need seven calculation nodes to perform the function of a single Math node.

Min and Max Nodes in Policy Designer

Min and Max nodes are Calculation nodes that you can use in a policy model to determine the row of a collection which contains the smallest or largest value, respectively, in a specified column.

The input for a Min or Max node must be a *collection* with a column containing numeric or time-based values. The output of a Min or Max node is any field in the row that the Min or Max node retrieves. If multiple rows contain the minimum or maximum value, then only the first row encountered is returned.

Node Properties

The Properties windows for Min and Max nodes contain the items that are described in the following table.

Item	Description	Notes
Collection section	Specifies the collection for which you want to determine the smallest or largest value in a certain column.	You can select 🖉 to specify the output of a predecessor node in this section.
Collection Column section	Specifies the column that contains the values of which you want to determine the smallest or largest value.	This list contains the columns that are available in the selected collection. The column that you select must contain numeric or time-based values.

Max node

The following example illustrates how you can use a Max node to determine which row of a query has the largest value in certain field and then use a successor node to evaluate a different field in the same row. Consider the following nodes and connections.
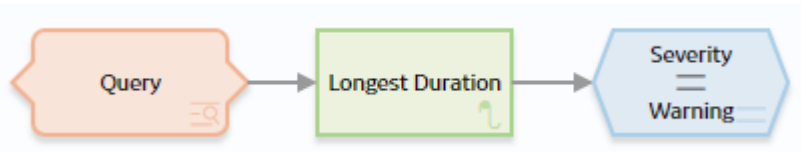
As shown in the following image, you can use the **Properties** window for the Longest Duration Max node to select the Query node and the Result Set option. The Duration (seconds) column of the Result Set is also selected, indicating that the query row with the longest duration will be returned.

As shown in the following image, you can then use the **Properties** window for the Equal node to determine if the *Severity* field in the query row with the longest duration contains the value Warning.
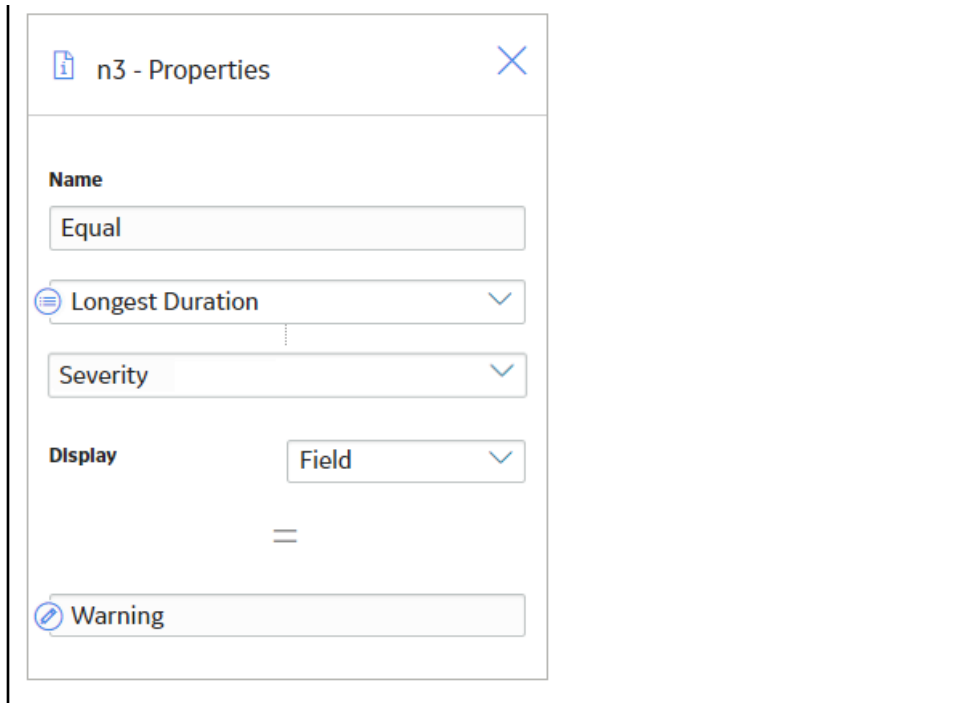
Round Nodes in Policy Designer

A Round node is a Calculation node that you can use in a policy model to perform the following:

- Round a value to a specific number of decimal places. This is called decimal rounding.
- Round a value to a specific number of significant figures. This is called precision rounding.

The input of a Round node must be a single, numeric value. The output of a Round node, Answer, contains the rounded value.

About Decimal and Precision Rounding

When you use decimal rounding, you can specify the number of decimal places to which you want to round the input value. For example, if the input value is 3.51 and you indicate that you want to round to the nearest tenth (by specifying the value 1 in the **Digits** box on the **Properties** window), the input value will be rounded to 3.5.

When you use precision rounding, you can specify the number of digits that you want keep from the input value. For example, if the input value is 7,658,321 and you indicate that you want to keep the first three values (by specifying the value 3 in the **Digits** box on the **Properties** window), the input value will be rounded to 7,660,000 (where the third digit, 5, is rounded to 6, and the remaining digits are changed to 0).

Note: The Round node always rounds values to the nearest number (that is, you can not specify that it always round values up or down). To round values up or down, you can use a Remainder node with an Add or Subtract node.

Node Properties

The Properties window for a Round node contains the items that are described in the following table.

Item	Description	Notes
Value section	Specifies the value that will be rounded.	You can select to specify the output of a predecessor node in this section.
Digits section	The value in this section serves a different purpose depending on the option that you select in the **Mode** section. You can select: • **Decimal rounding**: Specifies the number of decimal places to which you want to round the input value. • **Precision rounding**: Specifies the number of digits that you want keep from the input value.	You can select to specify the output of a predecessor node in this section. If the list in the **Mode** section contains the value: • **Decimal rounding**, you must enter a whole number between 0 and 15 in this box. • **Precision rounding**, you must enter a whole number greater than zero. Regardless of the values that you specify for the Round node, when you validate the policy logic, only two decimal places are displayed. You can see the actual values by accessing the **Properties** window for the appropriate node.
Mode list	Specifies the type of rounding that the node will perform.	This list contains the following options: • Decimal rounding • Precision rounding By default, the **Mode** list contains the value **Decimal rounding** and the **Digits** box is empty (this is the same as typing 0 [zero]). This means that decimal values will be rounded to the nearest whole number.

Round node

The following example illustrates how you can use the Round node in a policy model to round the value of an OPC Tag reading, represented by the Pressure node, to the nearest ten. Consider the following nodes and connection.

As shown in the following image, you can use the **Properties** window for the Round node to specify the type of rounding which you want to use. In this example, Precision rounding is selected as the type of rounding and 1 is specified as the number of digits from the input value that will be kept. This means that if, for example, the input to the node is 68, the output value will be 70.

The Round node in this example could be connected to various successor nodes in order to facilitate calculations. You could, for example, connect the Round node to a Case node, and then define a specific case to be executed based on the rounded input (that is, 10, 20, 30, and so on). Using the Round node in this way facilitates the execution of the Case node by limiting the number of possible inputs.

R Script Nodes in Policy Designer

An R Script node is a Calculation node that you can use in a policy model to return results that are calculated by an R script created outside the policy using the R script editor.

The inputs for an R Script node must correspond to the type of inputs expected by the R Script. The outputs of an R Script node contain the values calculated by the R script.

Node Properties

The **Properties** window for an R Script node contains the items that are described in the following table.

Item	Description	Notes
Path	Specifies the path to the R script that will run when the policy is executed.	You can enter the path manually, or you can browse to the R Script by selecting 👓.
Additional sections corresponding to the inputs expected by the R script.	Specifies the values that will be used in the R script to calculate a result.	The number of sections displayed in the **Properties** window is determined by the number of inputs configured in the specified R script. In each section, you can select 🖊 to display the output of a predecessor node. The value that you specify should correspond to the type of input expected by the R Script (that is, a Vector Matrix or Single Value of a specified type (Numeric, Character, Boolean, or Time & Date), or a Data Frame).

Working with R Scripts

You should account for the following behaviors when working with R scripts in policies:

- The policy execution engine sends date inputs to the R script in UTC. If your R script performs any calculations based on dates, the date output must also be in UTC and use the standard date format yyyy-MM-dd hh:mm:ss.fff.
- For Data Frame inputs:
  - The entire collection you specify is passed to the R script.
  - The supported data types for each column are Time & Date, Numeric, Character, and Boolean.
  - Reading values in readings collections specified directly from an input node such as the OPC Tag, Measurement Location, or Health Indicator node are untyped and therefore sent to the R script as string values.

R Script Using Multiple Inputs from Policy

The following example illustrates how you can use an R Script node to evaluate certain values in other nodes and return a particular value. Consider the following nodes and connections.
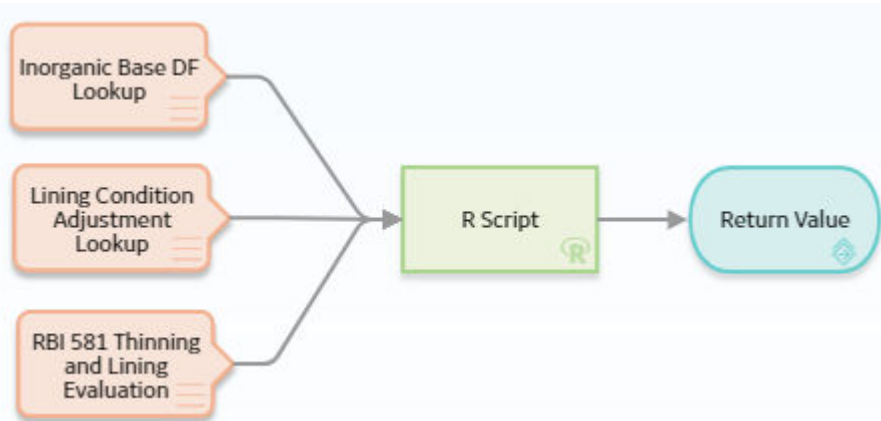
As shown in the following image, you can use the **Properties** window for the R Script node to select a particular R script that was created using the R script editor. The R script in this example requires three input values: LinerBaseDF, LiningConditionAdjustment, and OnlineMonitoringAdjustment. These input values correspond to variables in the specified R script.

You can see that output values from the predecessor Input nodes have been specified to supply the values required by the R script. When this policy is executed, these values will be used to calculate the result of the R script. The Return Value node then returns the value calculated by the R Script node.
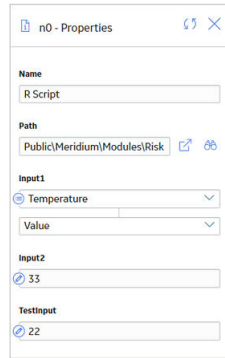
Calculating Average and Standard Deviation of a Reading Collection

The following example illustrates how you can use a policy to execute an R script that calculates statistical information regarding readings related to a Measurement Location, and then send that information via email message to a designated recipient.

Consider the following R script and corresponding parameters.

```
1  maxDate <- max (dfInputReadings[[1]])
2  avgReading <- mean (as.numeric(dfInputReadings[[2]]))
3  StDev  <- (dfInputReadings[[2]])
4  rows <- nrow (dfInputReadings)
```

	ID	NAME	TYPE	DIRECTION	REQUIRED
	dfInputReadings	dfInputReadings	Data Frame	Input	✓
	avgReading	Average Reading Value	Single value - Numeric	Output	
	rows	Count of Readings	Single value - Numeric	Output	
	StDev	Standard Deviation	Single value - Numeric	Output	
	MaxDate	Max Reading Date	Single value - Time & Date	Output	

Parameters

This R script calculates the average reading value, count of readings, standard deviation, and most recent reading date of input values provided as a Data Frame.

The following policy is used to provide the input to the R script and to use the results of the R script in subsequent operations.
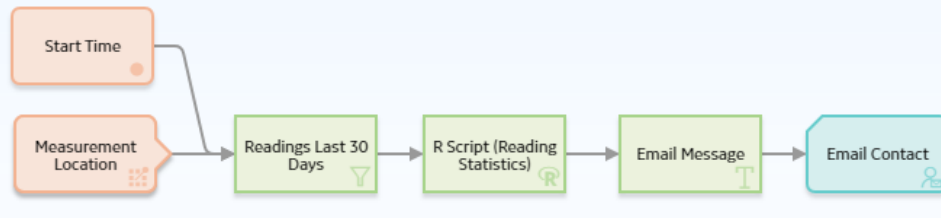
This policy first filters Measurement Location readings to only those readings recorded within the last 30 days. Then, the filtered readings are sent as a Data Frame input to the R script.

The following image shows the **Properties** window for the R Script node.
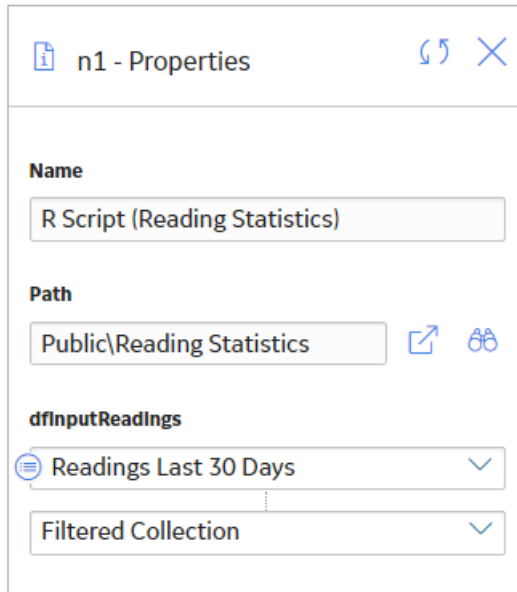
Next, the calculated results returned by the R script are used in a Text node to construct a message that will ultimately be sent via email to a responsible user.

The following image shows the **Properties** window for the Text node.

n5 - Properties

Name

Email Message

Text Pattern

<p>Measurement Location {0} recorded a total of {1} reading value since {2}. The average value was {4} and the Standard Deviation was {5}. The latest Reading date was

Index	Value	
{0}	Measurement Location	
	Description	
{1}	R Script (Reading Statistics)	
	Count of Readings	
{2}	Start Time	
	Value	
{3}	R Script (Reading Statistics)	
	Max Reading Date	
{4}	R Script (Reading Statistics)	
	Average Reading Value	
{5}	R Script (Reading Statistics)	
	Count of Readings	

The following image shows the full text pattern specified in the Text node.

Text Pattern

<p>Measurement Location {0} recorded a total of {1} reading value since {2}. The average value was {4} and the Standard Deviation was {5}. The latest Reading date was {3}. </p>

Done

191

Sum Nodes in Policy Designer

A Sum node is a Calculation node that you can use in a policy model to calculate the total value of data in a specified column of a collection.

The input for a Sum node must be a *collection* containing numeric or time span values. The output of a Sum node, Value, contains the result of the calculation for the specified column.

Node Properties

The **Properties** window for a Sum node contains the items that are described in the following table.

Item	Description	Notes
Collection section	Specifies the collection that contains the values for which you want to calculate the total value.	You can select ✎ to specify the output of a predecessor node in this section. **Note:** If the input specified for a Sum node is an empty collection, the resulting answer from the Sum node is automatically zero (0).
Collection Column section	Specifies the column that contains that values for which you want to calculate the total value.	This list contains the columns that are available in the selected collection. The column that you select must contain numeric or time span values.

Sum node

The following example illustrates how you can use a Sum node in a policy model to determine the total value of the Measurement Location readings that were taken within the last week. Consider the following nodes and connections.
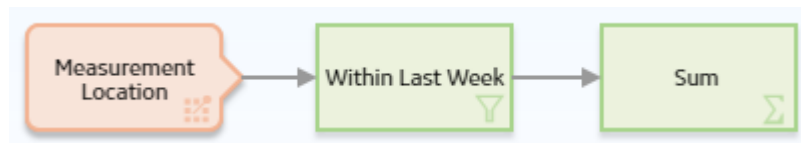
In this example, the Collection Filter node is used to filter the readings associated with the Measurement Location node to only the readings that were recorded within the last week. The Sum node is then used to calculate the total value of these readings. The following image shows what the **Properties** window looks like for the Sum node.
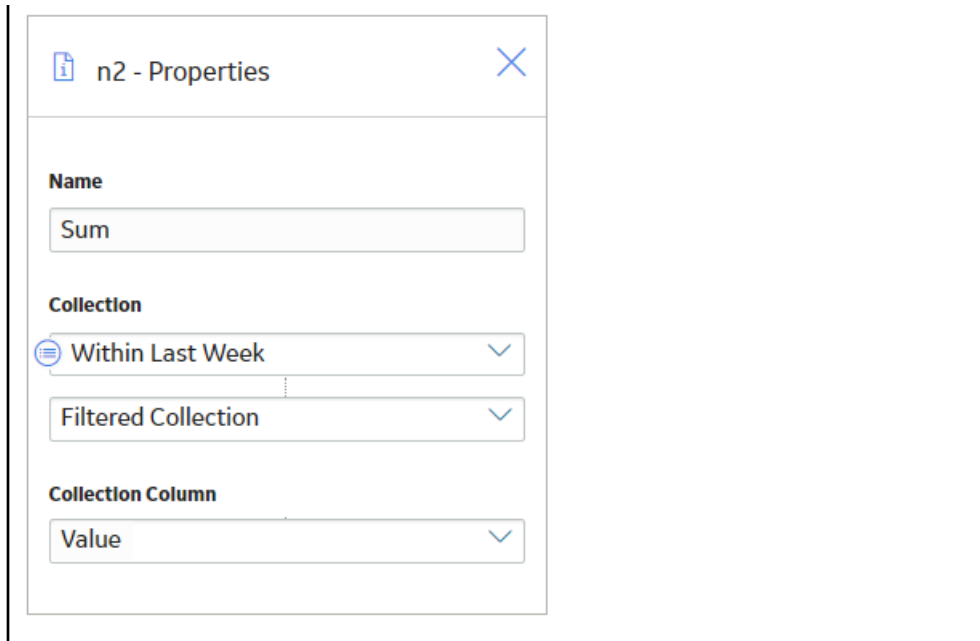
Text Nodes in Policy Designer

A Text node is a Calculation node that you can use in a policy model to create a custom text string based on constants and the values returned by other nodes in a policy execution.

The inputs for a Text node must be single values. The output of a Text node, Answer, contains the custom text string.

Node Properties

The **Properties** window for a Text node contains the items that are described in the following table.

Item	Description	Notes
Text Pattern box	Defines the pattern for the custom text string that will be the output of the Text node.	You must enter a valid .NET String.Format expression in this box. When you create the text pattern, the index numbers you use must be sequential starting at zero. However, the numbers do not have to be listed sequentially within the string and each number can be used multiple times.
Index / Value section	Specifies the value that each index specified in the **Text Pattern** box represents.	When the node is executed, the indexes in the text string will be replaced with the corresponding values you specify in this section. **Important:** You must define values for all indexes defined in the **Text Pattern** box. You can select ⊘ to specify the output of a predecessor node in this section. You can use the ＋ and 🗑 buttons in this section to add or delete values. **Note:** Constant values to represent dates (e.g., now, today, Sunday, October, etc.) or specific values (e.g., Pi or e) are not supported in this section. To use these one of these constants, you must define it using a Constant or Point Value node with an appropriate type.

.NET String.Format Expressions

The Text node supports .NET String.Format expressions, including the formatting capabilities supported by String.Format. The following table shows various example expressions and the resulting strings.

Text Pattern	Defined Index Values	Resulting String
The temperature of the motor is {0} degrees Celsius.	{0}=12	The temperature of the motor is 12 degrees Celsius.
The temperature of the {1} is {0:F2} degrees Celsius.	{0}=23.456 {1}=pump	The temperature of the pump is 23.46 degrees Celsius.

Text Pattern	Defined Index Values	Resulting String
The temperature was recorded at {0:t} on {0:d}.	{0}= 1/1/2016 05:30:00	The temperature was recorded at 5:30 AM on 1/1/2016.
The cost to repair the damaged part is {0:C2}.	{0}=1600	The cost to repair the damaged part is $1,600.00.

Note: For further explanation of valid expressions and formatting capabilities, refer to the String.Format documentation provided by the Microsoft Developer Network (MSDN).

Text node

The following example illustrates how you can use the Text node in a policy to create a custom email message using various values from the policy's execution.
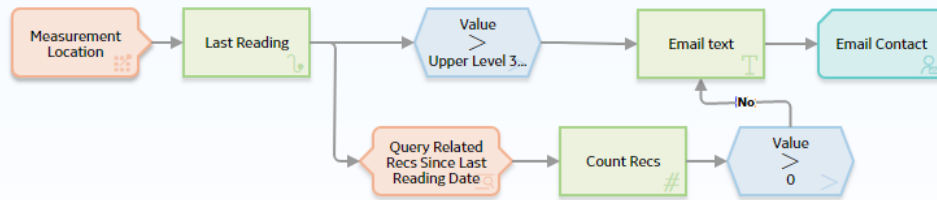
In this policy, the last reading related to a particular Measurement Location record is monitored. If the reading value exceeds the corresponding Upper Level 3 limit and no recommendations have been created since the reading was recorded, an email message is sent. The body of the email message includes a custom text message created using the Text node.

The following image shows the **Properties** window for the Text node.

You can see that various values from the policy execution are included within the text, thereby giving the recipient of the email further details and context surrounding the conditions that caused the message to be sent.

Assume that the policy returns the following values:

- **Measurement Location - Upper Level 3 Numeric Value**: 90.555
- **Measurement Location - Checkpoint ID**: chk-005
- **Last Reading - Timestamp**: 3/9/2016 12:27 PM

In this case, the output of the string node will contain the following text: Latest Reading for Measurement Location chk-005 taken on Wednesday, March 9, 2015 exceeds the Upper Level 3 limit of 90.55. No Recommendation has been created for this ML on or after Wednesday, March 9, 2015.

Threshold Statistics Nodes in Policy Designer

A Threshold Statistics node is a Calculation node that you can use in a policy model to determine the frequency and duration over which input values cross a defined threshold (that is, meet a defined condition).

The input for a Threshold Statistics node must be a *collection* with columns containing timestamps and numeric values. A Threshold Statistics node generates the following outputs:

- Count, which represents the number of times that input values crossed the defined threshold.

- Accumulated Time, which represents the amount of time over which input values crossed the defined threshold.

Node Properties

The **Properties** window for a Threshold Statistics node contains the items that are described in the following table.

Item	Description	Notes
Collection section	Specifies the collection for which you want to calculate a threshold statistic.	You can select ✐ to specify the output of a predecessor node in this section.
Timestamp Column list	Specifies the column that contains the timestamps that you want to use to calculate the accumulated time during which the input values cross the defined threshold.	This list contains the columns that are available in the selected collection.
Value Column list	Specifies the column that contains the input values that will be compared to the threshold value.	This list contains the columns that are available in the selected collection.
Threshold section	Specifies the threshold value that will be compared to the values that are defined in the **Value Column** list.	You can select ✐ to specify the output of a predecessor node in this section.
Operator list	Specifies the comparison operator that will be used to compare inputs values to the threshold value.	This list contains the following operators: • Less than (<) • Less than or equal (<=) • Equal (=) • Greater than or equal (>=) • Greater than (>)

Threshold Statistics node

The following example illustrates how you can use a Threshold Statistics node with Condition nodes to evaluate how many times and for how long reading values exceeded a defined threshold. Consider the following nodes and connections.
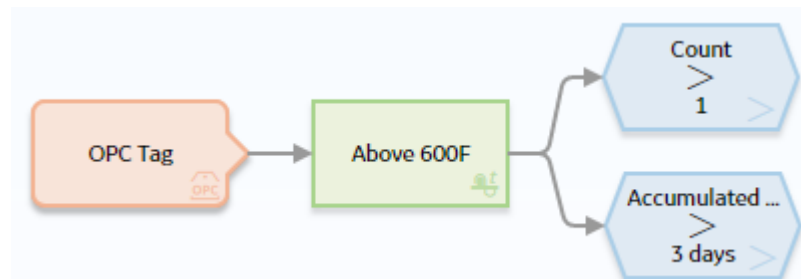
In this model, the collection of readings from the **Health Indicator** node, which represents a temperature, are filtered by the **Filter This Week** Collection Filter node to select the Readings for the current week. The filtered collection of Readings is passed to the Above 600 F Threshold Statistics node, which compares the filtered Readings to the defined threshold. The outputs from the Above 600 F Threshold

Statistics node,(that is, the number of times and the accumulated time for which the threshold was exceeded) are compared to limit values using two Greater Than nodes. If either Greater than condition is met, an Email node is used to alert.

As shown in the following image, you can use the **Properties** window for the Above 600 F Threshold Statistics node to:

- Indicate that you want to evaluate the filtered collection of Readings.
- Identify the Timestamp and Value columns of the HDA Readings collection as the columns that store the timestamps and values that you want to evaluate.
- Set the Threshold against which you want to compare the values.
- Indicate the operator to use when comparing the Values to the Threshold.

More About This Example

To further explain this example, consider the following graph, where:

- The x-axis displays timestamps.
- The y-axis displays reading values.
- The blue line displays reading values that were added to the Health Indicator on specific dates.
- The red line identifies the threshold value 600.
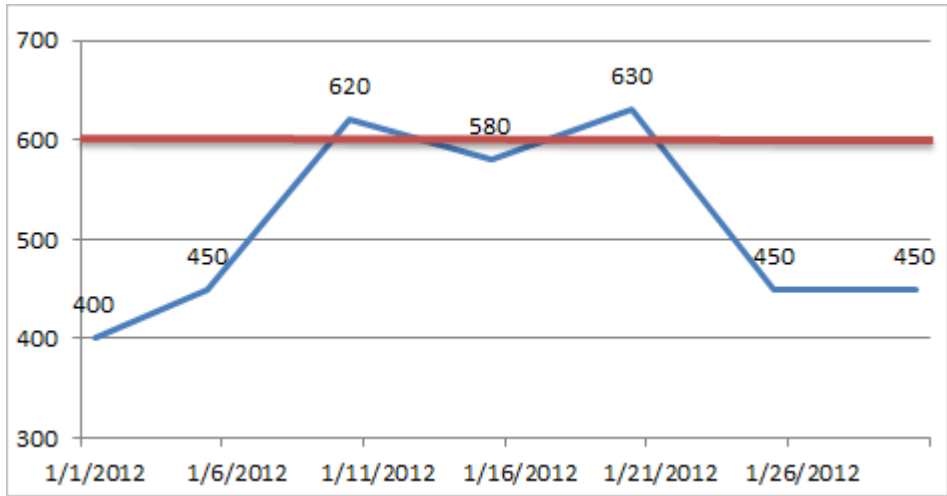
In the graph, you can see that a reading value above 600 occurred two times, where one value was 620 (on 2021-06-03) and the other value was 630 (on 2021-06-05). These values are treated as two separate occurrences of exceeding the threshold value because the reading value dropped below 600 in between. In this scenario, the condition Count > 1 would be true, but the condition Accumulated Time > 3 Days would be false.

Now consider the following graph in which the value recorded on 2021-06-04 was also above 600. In this scenario, the APM system determines that the threshold value was exceeded only one time, not three times. Specifically, whenever a reading value exceeds the threshold value, if the threshold value continues to be exceeded with each subsequent reading value, the APM system treats it as one occurrence of exceeding the threshold. Whenever a subsequent reading value drops below the threshold value, the next time another reading value exceeds it again, the APM system treats that as a separate occurrence.
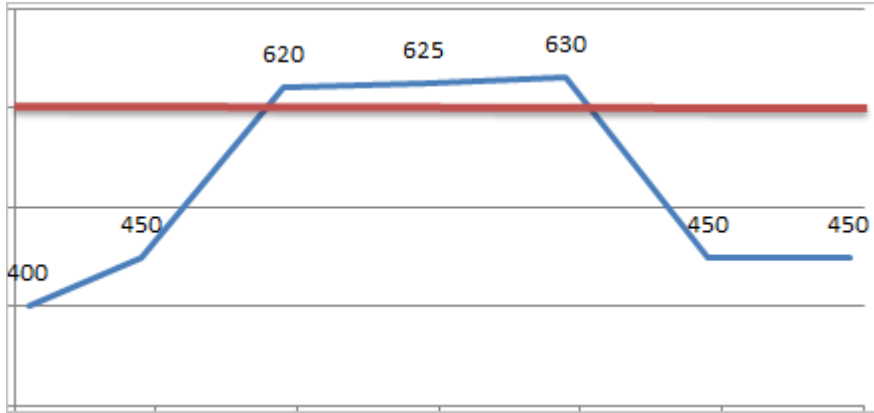
Here, you can see that the first reading value that exceeded the threshold value is 620. After that, two consecutive reading values also exceeded the threshold: 625 and 630. These additional reading values, however, are not treated as a second occurrence of exceeding the threshold value because they are consecutive.

The duration for which the threshold was exceeded each time is calculated as the difference between the timestamp of the first Reading which exceeds the threshold and the timestamp of the first Reading that does not exceed the threshold. In this case, the total time for which the threshold was exceeded is evaluated as exactly 3 days, which is the time between 2021-06-03 00:00 and 2021-06-06 00:00.

Therefore, both the conditions Count > 1 and Accumulated Time > 3 Days would be false.

Here, you can see that all the Readings evaluated exceed the threshold. In this case, the values are treated as a single occurrence and the time for which the threshold was exceeded is evaluated as the difference between the timestamps of the first and last Readings, even though the event is ongoing. In this case, the condition Count > 1 would be false and the condition Accumulated Time > 3 Days would be true.

APM applies the same principle whenever the first or last reading evaluated exceeds the threshold. Consider the following graph:
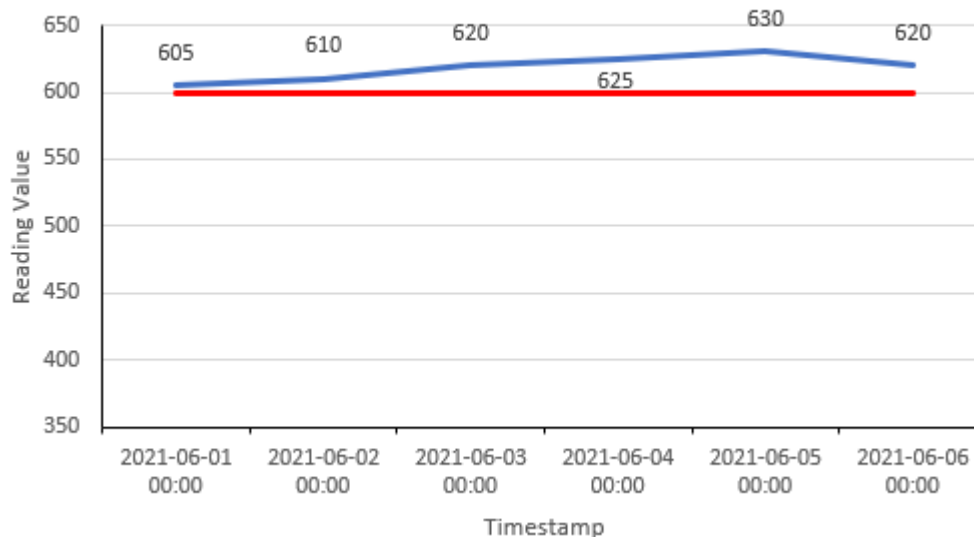
Here you can see that the first three Readings exceed the threshold, the next Reading is below the threshold, and the last two readings are above the threshold. In this case, there are two separate events, and the total time for which the threshold is exceeded is evaluated as 4 days, comprised of 3 days for the first event and 1 day for the second event. Therefore, the condition Count > 1 would be true and the condition Accumulated Time > 3 Days would be true.

Action Nodes

About Action Nodes in Policies

Action nodes represent actions that are performed when a policy is executed and the conditions preceding the Action node are met.

Action Nodes

- Add Value to Health Indicator
- API
- Apply Strategy Template
- Close Event
- Create Entity
- Create Event
- Create Production Event
- Create Recommendation
- Create Relationship

- Deactivate This Instance
- Delete Entity
- Delete Relationship
- Edit Entity
- Email Contact
- Return Value
- Rule
- State Transition
- Sub Policy

Add Value to Health Indicator Nodes in Policy Designer

An Add Value to Health Indicator node represents an action to create a Health Indicator Value record and link it to a Health Indicator record.

Note: You can access the Add Value to Health Indicator node only if the Asset Health license is active in your APM system.

When an Add Value to Health Indicator node is executed, a Health Indicator Value record is created with the timestamp and value defined by the policy logic and corresponding instances.

The outputs of the Add Value to Health Indicator node are the following system fields for the record that the node creates: *Entity Key* and Family Key.

Node Properties

The **Properties** window for an Add Value to Health Indicator node contains the items that are described in the following table.

Item	Description	Notes
Health Indicator Entity Key section	Specifies the entity key of the Health Indicator record to which the new Health Indicator Value record will be linked.	You must specify a Health Indicator record without a source. If an appropriate Health Indicator record does not yet exist, you can specify a Health Indicator node and then create the record via the **Instances** pane.
Timestamp section	Specifies the value that will populate the Timestamp field in the new Health Indicator Value record.	You can select to specify the output of a predecessor node in this section. The value in this section must be a timestamp.
Value (Numeric) section	Specifies the value that will populate the Value (Numeric) field in the new Health Indicator Value record.	You can select to specify the output of a predecessor node in this section. The value in this section must be numeric.

Tip: Click here to see an example policy in which an Add Value to Health Indicator node is used to create health indicator readings that represent the overall health of an asset.

API Nodes

An API node represents an action to execute an application programming interface (API) in APM.

The outputs of an API node are the API response codes (for example, 200 for a successful API call, along with the API response as applicable).

Before using this node, you must configure the API User Credentials.

Tip: The API response is usually a JSON object. You can use the JSON Parser node to convert the JSON object into single values and collections that can be processed by other policy nodes.

Important:

- If the API node is executed during policy execution, any action taken by the API will not be rolled back even if the policy execution results in an error.
- The API node is not executed during policy validation. Therefore, you must execute the policy to confirm that the API node functions as expected.

Node Properties

The **Properties** window of an API node contains the items described in the following table.

Item	Description	Notes
Name	The name of the node.	The value must be unique. It is displayed in the node on the policy canvas.
URL	The URL for the API call.	You can select to specify the output of a predecessor node in this section. Following are some examples of URL: • `api/v1/core/catalog/item/ 64269226118` • `apm-connect-ingestor:5000/v1/ bundles`
User	The display name of the API User Credential record that will be used to execute the API.	You can select an API User Credential from the drop-down list box in this section. Alternatively, you can select to specify the output of a predecessor node in this section. If you use this option, the value you specify must be the entity key of a valid API Node Credential record. For information, refer to API Node Credentials.
HTTP Verb	The verb to be used by the API call.	The following options are available: • Delete • Get • Patch • Post • Put

Item	Description	Notes
Payload	The payload for the API call.	The payload is usually a JSON object. You can select ✎ to specify the output of a predecessor node in this section. **Tip:** You can use a Text node to create a variable payload that depends on the logic of the policy.
	Adds a new row to the **HTTP Headers** section of the **Properties** window.	Each row represents a parameter that will be included in the HTTP headers of the API call.
HTTP Header	The name of the HTTP Header parameter.	You can select ✎ to specify the output of a predecessor node in this section.
Value	The value of the HTTP Header parameter.	You can select ✎ to specify the output of a predecessor node in this section.
🗑	Deletes the row from the **HTTP Headers** section of the **Properties** window.	

Tip: You can access Swagger documentation for published APIs by logging in to APM, and then entering the following URL in a new browser tab: `https://<APM base URL>/meridium/api/docs/index.html`.

This documentation does not cover all available APIs. If you need additional information, contact GE Vernova Support.

Using the API Node and the JSON Parser Node in a Policy

In the following example, the API node is used to retrieve a Dataset from the Catalog, which contains information about 360 View and related 360 View Tab Config records that you want to create. The JSON Parser node is used to convert the API response into a collection, which is then used in two Create Entity nodes to add records in the APM database based on the values in the Dataset. The new records are then linked using a Create Entity node.

The following images show how the API Text and Get Dataset (API) nodes are configured to retrieve the dataset from the catalog.

In most cases, you can determine the API URL, HTTP Verb, and Payload (if any) by accessing the network section in your browser's developer tools while performing the required action in the APM UI.

You can also see how the API response JSON is formatted. The following image shows the API response JSON for this example.

X Headers Preview Response Initiator Timing Cookies

▼ {datasetType: "Excel", columnDelimiter: "Comma", hasColumnDefinition: false, startRow: "", endRow: "",…}
    catalogItemCaption: "360 Views"
    catalogItemId: "360 Views"
    catalogItemKey: "3078736"
    columnDelimiter: "Comma"
    datasetType: "Excel"
    endRow: ""
    hasColumnDefinition: false
    msDataSet: "<NewDataSet>\r\n <xs:schema id=\"NewDataSet\" xm Show more (78.4 kB) Copy
  ▼ resultGrid: {columns: [{id: "MI_ASSE_ROLE_C", alias: "MI_ASSE_ROLE_C", dataType: "character"},…], data: [,…]}
    ▶ columns: [{id: "MI_ASSE_ROLE_C", alias: "MI_ASSE_ROLE_C", dataType: "character"},…]
    ▼ data: [,…]
      ▼ [0 … 99]
        ▶ 0: {MI_ASSE_ROLE_C: "MI Foundation User", MI_ASSE_TYPE_C: "MI_EQUIP000", ROLE_ID: "MI Foundation User",…}
        ▶ 1: {MI_ASSE_ROLE_C: "Log Administrator", MI_ASSE_TYPE_C: "MI_EQUIP000", ROLE_ID: "Log Administrator",…}
        ▶ 2: {MI_ASSE_ROLE_C: "Log Contributor", MI_ASSE_TYPE_C: "MI_EQUIP000", ROLE_ID: "Log Contributor",…}
        ▶ 3: {MI_ASSE_ROLE_C: "MI ACA Owner", MI_ASSE_TYPE_C: "MI_EQUIP000", ROLE_ID: "MI ACA Owner",…}
        ▶ 4: {MI_ASSE_ROLE_C: "MI AHI Administrator", MI_ASSE_TYPE_C: "MI_EQUIP000",…}
        ▶ 5: {MI_ASSE_ROLE_C: "MI AHI User", MI_ASSE_TYPE_C: "MI_EQUIP000", ROLE_ID: "MI AHI User",…}
        ▶ 6: {MI_ASSE_ROLE_C: "MI AHI View", MI_ASSE_TYPE_C: "MI_EQUIP000", ROLE_ID: "MI AHI View",…}
   ▶ 7: {MI_ASSE_ROLE_C: "MI Calibration Administrator", MI_ASSE_TYPE_C: "MI_EQUIP000",…}

The following image shows how the JSON Parser node is configured to retrieve the dataset from the catalog. Each JSON Path entry will create a column in the collection output from the node, populated with the values in the resultGrid.data section of the JSON response. The names of the columns are configured to match the IDs of the fields that we want to populate in the new entity records.
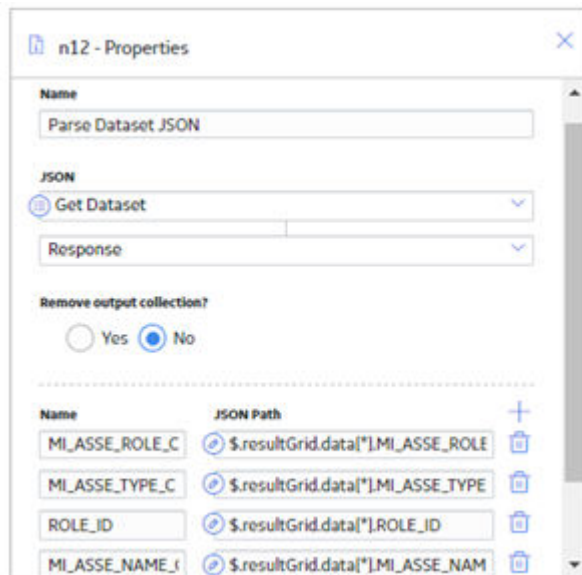
The following image shows the configuration of the Create 360 View (Create Entity) node, which uses the Auto-map feature to create a collection of entity records, using the values in the collection output from the Parse Dataset JSON (JSON Parser) node where the column ID in the collection output matches the field ID of the new entity. The Create 360 View node is also configured to set the value of the **Is Active** field on all the new entities to **Yes**.

Note that the Create 360 View Tab node uses the same output from the Parse Dataset JSON node. Each Create Entity node uses only the columns that match its own field IDs and ignores the other columns.
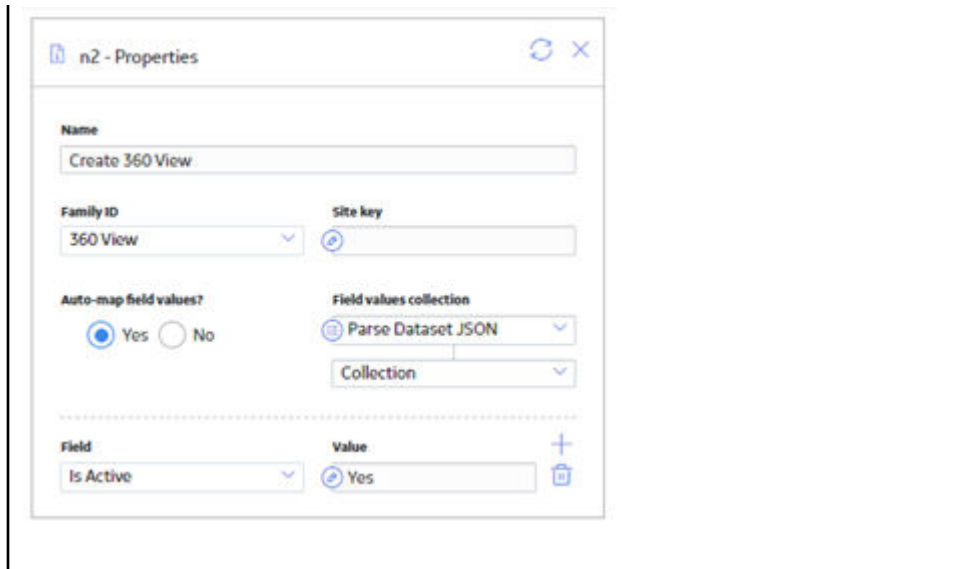
Apply Strategy Template Nodes in Policy Designer

The Apply Strategy Template node represents an action to apply an asset strategy template to an asset or asset strategy. You can use the Apply Strategy Template node to apply the asset strategy template as a copy or master to the asset or asset strategy. For more information on asset strategy and asset strategy templates, refer to the Asset Strategy Management documentation.

Note:

- You can access the Apply Strategy Template node only if the Asset Strategy Management license is active in your APM system.
- Even if the execution of the Apply Strategy Template node is successful, the specified template is applied to the asset or asset strategy only if the policy containing the node is successfully executed.

The Apply Strategy Template node has no outputs.

Node Properties

The **Properties** window for an Apply Strategy Template node contains the fields and sections described in the following table:

Field/Section	Description	Note
Strategy Template Key	Specifies the entity key of the asset strategy record that you want to apply as a template.	None.
Asset or Strategy Key	Specifies the entity key of the asset or asset strategy record to which you want to apply the template.	None.

Field/Section	Description	Note
Select option to apply template	Specifies whether the template must be applied as a copy or master.	In this section, you can select one of the following options: • **Apply the template as a copy** • **Apply the template as a master** **Note:** By default, the **Apply the template as a copy** option is selected.
Select portions of template to apply	Specifies whether both actions and risks, or only the risks associated with the template must be applied to the asset or asset strategy. **Note:** This section is enabled only if you select **Apply the template as a copy** in the **Select option to apply template** section.	In this section, you can select one of the following options: • **Apply both actions and risks** • **Apply risks only** **Note:** By default, the **Apply both actions and risks** option is selected.
Select option to handle existing Asset Strategies	Specifies whether the actions and risks that are already associated with the strategy must be deleted or the items of the template that you want to apply must be appended to the existing actions and risks associated with the strategy. **Note:** This section is enabled only if you select **Apply the template as a copy** in the **Select option to apply template** section.	In this section, you can select one of the following options: • **Mark existing items for deletion** • **Append the template items to the existing items** **Note:** By default, the **Mark existing items for deletion** option is selected.

The following example illustrates how you can use a Apply Strategy Template node to apply an appropriate asset strategy template as master to every Equipment record that is added to the APM database:

In this example, the Current Entity node represents the Equipment record that is added to the database. Based on the model of the equipment, the Query node identifies the asset strategy template that must be applied to the equipment. The Apply Strategy Template node then applies the template identified by the Query node as master to the equipment.

The **Properties** window for the Apply Strategy Template node described in this example is shown in the following image:

Close Event Nodes in Policy Designer

A Close Event node represents an action to populate fields in an existing Policy Event record to signify that the associated event has been closed.

When a Close Event node is executed, the following fields will be populated in the associated Policy Event record:

- End Time
- Close Description

To use a Close Event node, the policy model must also contain a Create Event node that is associated with the same Policy Event record (that is, with the same value specified in the **Event Name** section on the **Properties** window). The properties of the Create Event node must be configured such that:

- The **Has Duration** section is set to True.
- The **Start Time** section contains a value.
- The **End Time** section does not contain a value.

In the policy model, a dotted line will appear between Create Event and Close Event nodes that are associated with the same Policy Event record.

The Close Event node has no outputs.

Node Properties

The **Properties** window for a Close Event node contains the items that are described in the following table. In each section, you can select ✐ to display the output of a predecessor node. The values that you define in each section will be used to populate the corresponding fields in the associated Policy Event record.

Item	Description	Notes
Event Name section	Specifies the name of the event that you want to close.	**Important:** The value that you specify in this section must match the Event Name value for the corresponding Create Event node (that is the value in the Name field in the associated Policy Event record) in order for the event to be closed successfully. **Note:** If you specify the output of a predecessor node, whether the nodes match depends on the values used when the policy is executed. For example, you can specify the **Event Name** in the Create Event node using a field from an Entity node and specify the **Event Name** in the Close Event node using the output of a Case node. As long as the values used in execution match, the events will be recognized as matching.
End Time section	Specifies the timestamp that is associated with the end of the event.	This value is required. If you enter a date, you must use the correct format.
Close Description section	Specifies a description of why the event was closed.	None.

Tip: Click here to see an example of this node used within a complete policy model.

Create Entity Nodes in Policy Designer

A Create Entity node represents an action to create a new record in a baseline or custom entity family. The fields in the new record are populated with the input values that you specify in the **Properties** window for the node.

Note: The Create Entity node displays only the entity families for which your APM system has active licenses, and for which you have create permissions.

The outputs of the Create Entity node are the following system fields for a record that the node creates: collection, Entity Key, and Family Key.

Note: The Collection output is available only when the Create Entity node is used to create a collection of entities. In this case, the Entity Key output contains the entity key of the first entity created.

Node Properties

The **Properties** window for a Create Entity node contains the items described in the following table.

Item	Description	Notes
Family ID	Specifies the unique ID of the family for which you want to create a record.	The **Family ID** list contains all the baseline and custom entity families in APM for which you have update privileges.
Site Key	Specifies the Site Reference Key of the site to which the new record must be assigned.	None.
Auto-map field values	Specifies whether the fields of the newly created record must be automatically populated with the values of a collection.	If the values that you want to specify for the record are part of a collection, you can use this option to specify the values for the fields of the record. If you use this option to specify the values for only some fields of the record, you must manually specify the values for the remaining fields.

Item	Description	Notes
Field values collection	Specifies the collection that contains the values that you want to specify for the newly created records. **Note:** If the collection is the result of a query, make sure the query is run in unformatted mode.	This section is enabled only when you select **Yes** for the **Auto-map field values** option. Depending on the source of the collection, the fields of the newly created records are automatically populated with the collection values in the following ways: • For a collection that is created using a query, if the label of a column matches the Field ID of a field, the field in each record is populated with the corresponding value of the column. For example, if the Field ID of the Reading Taken Date field is MI_READING0_RDG_TAKEN_DT_D, the field in each record is populated with the corresponding value of the collection column whose caption is MI_READING0_RDG_TAKEN_DT_D. • For a collection that is not created using a query, if the Column ID of a column matches the Field ID of a field, the field in each record is populated with the corresponding value of the column. For example, if the Field ID of the Reading Taken Date field is MI_READING0_RDG_TAKEN_DT_D, the field in each record is populated with the corresponding value of the collection column whose Column ID is MI_READING0_RDG_TAKEN_DT_D. **Note:** If the Field ID of a field does not match the Column ID or name of any column in the collection, the field is not populated with any value.
+	Adds a new row to the **Properties** window.	Each newly added row represents a field in the record for which you want to specify a value. **Note:** If you specify a value for a field that is configured to be auto-populated from a collection, the value that you specify takes precedence over the corresponding value in the collection.

Item	Description	Notes
Field	Specifies the field in the newly created record for which you want to specify a value.	This list contains baseline and custom fields of the record.
Value	Specifies the new value for the corresponding field.	If a field has a complex behavior defined by the field-level rules (for example, rules for valid values) and field-level behaviors, this behavior will not be reflected in the **Properties** window or detected during policy validation. Therefore, make sure that the values you specify are valid according to the baseline or custom field-level rules for the corresponding field. If a field value is defined by a system code, you must specify the system code in this field and not the value that is displayed to the user.

Tip: Click here to see an example of this node used within a complete policy model.

Create Event Nodes in Policy Designer

A Create Event node represents an action to create a Policy Event record, which stores information about events that are associated with Equipment or Functional Location records.

When a Create Event node is executed, a Policy Event record will be created using the values that you specify on the **Properties** window for the node. One Policy Event record will be created for each Create Event node in the policy model that has a unique name.

The outputs of the Create Event node are the following system fields for the record that the node creates: *Entity Key* and Family Key.

If a Policy Event record is created for an event that has a duration, additional Policy Event records with the same name will not be created until the event is closed (that is, the End Time field in the Policy Event record contains a value).

Note: You can use a corresponding Close Event node to populate the End Time and Close Description fields in the same Policy Event record when subsequent conditions indicate that the event has closed.

Node Properties

The **Properties** window for a Create Event node contains the items that are described in the following table. The values that you define in each section will be used to populate the corresponding fields in the Policy Event record that is created.

Item	Description	Notes
Asset Key section	Specifies the entity key of the Equipment or Functional Location record that is associated with the event, (that is, the record to which the Policy Event record will be linked).	This value is required.
Event Name section	Specifies the name of the event.	This value is required. You can select to specify the output of a predecessor node in this section.
Description section	Specifies a description of the event.	You can select to specify the output of a predecessor node in this section.
Event Type list	Specifies the type of event.	You can select one of the following values: • **Generic**: Indicates that the event is not associated with a specific type of event. • **Excursion**: Indicates that the event is associated with an operation that is outside of an established operating window.
Severity list	Specifies the severity of the event.	You can select one of the following values: • **Information**: Routine events not affecting asset health. • **Warning**: Events indicating a low-risk or early warning of asset health issues. • **Alert**: Events indicating a high-risk or imminent warning of asset health issues.
Start Time section	Specifies the timestamp that is associated with the beginning of the event.	This value is required. You can select to specify the output of a predecessor node in this section. If you enter a date, you must use the correct format.

Item	Description	Notes
End Time section	Specifies the timestamp that is associated with the end of the event.	You should only define a value in this section when the **Has Duration** field is set to True.
		Important: If the policy contains a corresponding Close Event node, you should not specify values in this section. You should specify values in the **End Time** section on the **Properties** window that is displayed for the Close Event node.
		You can select to specify the output of a predecessor node in this section.
		If you enter a date, you must use the correct format.
Close Description section	Specifies a description of why the event was closed.	**Important:** If the policy contains a corresponding Close Event node, you should not specify values in this section. You should specify values in the **Close Description** section on the **Properties** window that is displayed for the Close Event node.
		You can select to specify the output of a predecessor node in this section.
Has Duration section	Specifies whether or not there is an end time associated with the event.	You can select to specify the output of a predecessor node in this section.
		The value that you define must be a logical result (that is, Yes or No.)
		Important: If the policy contains a corresponding Close Event node, you must set the value in this section to Yes.
Time Line Reset section	This field is not currently used.	None.

Tip: Click here to see an example of this node used within a complete policy model.

Create Production Event Nodes in Policy Designer

A Create Production Event node represents an action to create a Production Event record that is populated with the input values that you specify on the **Properties** window.

Note: You can access the Create Production Event node only if the Production Loss Accounting license is active in your APM system.

The outputs of the Create Production Event node are the following system fields for the record that the node creates: *Entity Key* and Family Key.

Node Properties

The **Properties** window for a Create Production Event node contains the items that are described in the following table. In each section, you can select ✎ to display the output of a predecessor node. The values that you define in each section will be used to populate the corresponding fields in the Production Event record that is created.

Item	Description	Notes
Causing Asset Key section	Specifies the Equipment record that represents the piece of equipment that caused the event.	The **Causing Asset Key** value must match exactly the Entity Key of the asset that caused the event.
Description section	Specifies a detailed description of the event.	None.
End Date section	Specifies the date that the event ended.	Along with the start date, the end date determines whether or not the production event will be available in the **Production Event** list in the **Production Event** workspace.
Headline section	Specifies a short description of the event.	None.
Source Production Unit section	Specifies the production unit to which loss is attributed.	The created Production Event record will be automatically linked to the specified Production Unit record.
Start Date section	Specifies the date that the event started.	Along with the end date, the start date determines whether or not the production event will be available in the **Production Event** list in the **Production Event** workspace.

Note: If you add a custom field to the Production Event family, that field will also appear in the **Properties** window for the Create Production Event node.

Tip: Click here to see an example of this node used within a complete policy model.

Create Recommendation Nodes in Policy Designer

A Create Recommendation node represents an action to create a Policy Recommendation record that is populated with the input values that you specify on the **Properties** window for the node and includes (in the Recommendation Description field) a summary of the policy logic that caused the record to be created.

Note: You cannot specify the value for the Recommendation Description field in the **Properties** window for the Create Recommendation node. If you want to specify your own description instead of the automated policy logic summary, you can use a Create Entity node.

The outputs of the Create Recommendation node are the following system fields for the record that the node creates: *Entity Key* and Family Key.

Node Properties

The **Properties** window for a Create Recommendation node contains the items that are described in the following table. In each section, you can select ✎ to display the output of a predecessor node. The values

that you define in each section will be used to populate the corresponding fields in the Policy Recommendation record that is created.

Item	Description	Notes
Associated Reference section	Specifies the Reference ID of the event or any other entity that originated the recommendation.	None.
State Assignee User ID	Specifies the user that is assigned to the initial state.	The state assignee value must match exactly a valid User ID for an active Security User.
Create Work Request section	Specifies whether a work request for the EAM system that you have configured in APM will be created from the Policy Recommendation record.	The value in this section must be a logical result (that is, Yes or No).
Equipment ID section	The Record ID of the Equipment record to which the Policy Recommendation record will be linked.	You do not need to specify a value in both the **Equipment ID** and **Functional Location ID** sections. If you specify a value in either section, the APM system will automatically create relationships between the related Equipment, Functional Location, and Recommendation records. However, if you do specify values in both sections, they must correspond to the same asset.
Event Start Date section	Specifies the timestamp that is associated with the beginning of the event for which the Policy Recommendation record is created.	If you enter a date, you must use the correct format.
Functional Location ID section	The Record ID of the Functional Location record to which the Policy Recommendation record will be linked.	You do not need to specify a value in both the **Equipment ID** and **Functional Location ID** sections. If you specify a value in either section, the APM system will automatically create relationships between the related Equipment, Functional Location, and Recommendation records. However, if you do specify values in both sections, they must correspond to the same asset.
Recommendation Headline section	A short description of the recommended action.	None.

Item	Description	Notes
Recommendation Priority section	Specifies a priority value used to rank the importance of the recommendation.	The value that you specify must be a valid system code and be valid according to any field-level rules that you have specified for the Recommendation Priority field.
Target Completion Date	The date by which the recommended action should be completed.	This value is required. If you enter a date, you must use the correct format.

Note: If you add a custom field to the Policy Recommendation family, that field will also appear in the **Properties** window for the Create Recommendation node.

Tip: Click here to see an example of this node used within a complete policy model.

Create Relationship Nodes in Policy Designer

A Create Relationship node represents an action to create one or more new records in any baseline or custom relationship family. Each new record creates a relationship between a specified predecessor and successor entity record. The Create Relationship node has no outputs.

Note: The Create Relationship node displays only the relationship families for which your APM system has active licenses, and for which you have create permissions.

Node Properties

The **Properties** window for a Create Relationship node contains the items that are described in the following table.

Item	Description	Notes
Family ID list	Specifies the type of record that the Create Relationship node will create.	The **Family ID** list contains all the baseline and custom relationship families in APM for which you have insert privileges.
Predecessor Entity Key(s) section	Specifies the keys of the predecessor entity records in the relationship.	You can create relationships in the following ways depending on how you specify values in the **Predecessor Entity Key(s)** and **Successor Entity Key(s)** sections:
Successor Entity Key(s) section	Specifies the keys of the successor entity records in the relationship.	**Single one-to-one relationship**: Specify a single key in each section. **Multiple one-to-one relationships**: Specify the same number of keys in each section. A relationship will be added between each pair of keys. **One-to-many relationship**: Specify a key in one section and multiple keys in the other section. In each section, you can select 🖉 to display the output of a predecessor node. **Tip:** You can specify the Entity Key output from an Action node that creates records (such as the Create Entity node) to use the newly created records as either the predecessors or successors of a relationship. Relationships will not be created in the following scenarios: - If the relationship you specify already exists. - If no relationship definition is defined for the records you specify. - When attempting to create multiple one-to-one relationships, if the number of keys in each section do not match.
Select Successor Family from Collection? section	Allows you to select how the successor family of the record(s) that will be created is defined.	The default setting is Yes. If you select No, the Successor Family Key(s) section is updated to allow you to select a single Family ID from a list.
Select Predecessor Family from Collection? section	Specifies whether the Predecessor Family will be supplied as a collection of Family Keys	The default setting is Yes. If you select No, the Predecessor Family Key(s) section is updated to allow you to select a single Family ID from a list.

Item	Description	Notes
Predecessor Family ID list	Specifies the type of predecessor family records in the relationship.	This list is displayed when **Select Predecessor Family from Collection?** is set to No. The list shows entity families which are predecessors for relationship definitions of the relationship family selected in the Relationship ID list.
Predecessor Family Key(s) section	Specifies the keys of the predecessor family records in the relationship.	You can select to specify the output of a predecessor node in this section. The input value can be one of the following: • single family key, where the predecessor records are in the same family. • collection of family keys of the same length as the collection of predecessor entity keys, where the predecessor records may be in different families.
Successor Family ID list	Specifies the type of successor family records in the relationship.	This list is displayed when **Select Successor Family from Collection?** is set to No. The list shows successor entity families for relationship definitions of the relationship family selected in the Relationship ID list.
Successor Family Key(s) section	Specifies the keys of the successor family records in the relationship.	You can select to specify the output of a predecessor node in this section. The input value can be one of the following: • single family key, where the successor records are in the same family. • collection of family keys of the same length as the collection of successor entity keys, where the successor records may be in different families.

Tip: Click here to see an example of this node used within a complete policy model.

Deactivate This Instance Nodes in Policy Designer

A Deactivate This Instance node represents an action to deactivate the current policy instance. You can use this node to ensure that an action is taken only once per instance.

When a Deactivate This Instance node is executed, the policy instance whose values caused the node to be executed will be deactivated automatically. If more than one policy instance is associated with the policy, all other policy instances will remain unchanged.

After the conditions that caused the instance to be deactivated have been addressed, you can reactivate the policy instance manually from the **Instances** pane in Policy Designer or the **Policies** section in Asset Health Manager.

The Deactivate This Instance node has no outputs.

Node Properties

Other than optionally specifying a name for the node, there are no properties to configure for a Deactivate This Instance node.

The following example illustrates how you can use a Deactivate This Instance node to ensure that only one Policy Recommendation record is created as a result of the policy conditions being met for a specific policy instance. Consider the following nodes and connection.

In this example, the OPC Tag node and the Condition node to which it is connected indicate that a Policy Recommendation record should be created when the DA Reading Value associated with the OPC Tag node exceeds 200.

Assume that the following records are associated with the OPC Tag node via policy instances:

- Pump 101, whose DA Reading Value is greater than 200.
- Pump 102, whose DA Reading Value is not greater than 200.

When the policy is executed, a Policy Recommendation record for Pump 101 will be created. Because the policy also includes a Deactivate This Instance node, after the Policy Recommendation record is created, the APM system will deactivate automatically the policy instance that is associated with Pump 101. The policy instance that is associated with Pump 102 will remain active.

Because the policy instance that is associated with Pump 101 is no longer active, even if the DA Reading Value associated with the pump continues to exceed 200, additional Policy Recommendation records will not be created.

Delete Entity Nodes in Policy Designer

A Delete Entity node represents an action to delete from the APM database one or more records in any baseline or custom entity family.

The Delete Entity node has no outputs.

Node Properties

The **Properties** window for a Delete Entity node contains the items that are described in the following table.

Item	Description	Notes
Entity Key(s) section	Specifies the entity key(s) of the record(s) that will be deleted.	You can select to specify the output of a predecessor node in this section. If you specify multiple records to be deleted at the same time, they do not have to be in the same family.
Select Family from Collection section	Allows you to select how the family of the record(s) that will be deleted is defined.	The default setting is Yes. If you select No, the Family Key(s) section is updated to allow you to select a single Family ID from a list.
Family Key(s) section	Specifies the family key(s) of the record(s) specified in the Entity Key(s) section.	You can select to specify the output of a predecessor node in this section. The input value can be one of the following: • single family key (such as, the records to be deleted are in the same family) • collection of family keys of the same length as the collection of entity keys (such as, the records to be deleted may be in different families)

Delete Relationship Nodes in Policy Designer

A Delete Relationship node represents an action to delete from the APM database one or more records in any baseline or custom relationship family.

Note:

- The Delete Relationship node displays only the relationship families for which your APM system has active licenses, and for which you have delete permissions.
- The Delete Relationship node has no outputs.

Node Properties

The **Properties** window for a Delete Relationship node contains the items that are described in the following table.

Item	Description	Notes
Relationship Family ID list	Specifies the type of record that the Delete Relationship node will delete.	The **Relationship Family ID** list contains all of the baseline and custom relationship families in APM for which you have delete privileges.
Predecessor Entity Key(s) section	Specifies the key(s) of the predecessor entity record(s) in the relationship that will be deleted.	You can delete the following types of relationships depending on how you specify values in the **Predecessor Entity Key(s)** and **Successor Entity Key(s)** sections:
Successor Entity Key(s) section	Specifies the key(s) of the successor entity record(s) in the relationship that will be deleted.	• **Single one-to-one relationship** : Specify a single key in each section. The relationship between the two entity records will be deleted. • **Multiple one-to-one relationships** : Specify the same number of keys in each section. The relationship between each pair of entity records will be deleted. • **One-to-many relationship** : Specify one key in the one section and multiple keys in the other section. The relationship between the single entity record and all the entity records specified in the other section will be deleted. In each section, you can select ✐ to display the output of a predecessor node. If you specify multiple predecessor and successor records, they do not have to be in the same family. Relationships will not be deleted in the following scenarios: • If a relationship you specify does not exist. However, other valid relationships in the same transaction will be deleted. • When attempting to delete multiple one-to-one relationships, if the number of keys in each section do not match.
Select Successor Family from Collection? section	Allows you to select how the successor family of the record(s) that will be created is defined.	The default setting is Yes. If you select No, the **Successor Family Key(s)** section is updated to allow you to select a single Family ID from a list.

Item	Description	Notes
Select Predecessor Family from Collection? section	Specifies whether the predecessor family will be supplied as a collection of family keys.	The default setting is Yes. If you select No, the **Predecessor Family Key(s)** section is updated to allow you to select a single Family ID from a list.
Predecessor Family ID list	Specifies the type of predecessor family records in the relationship.	This list is displayed when **Select Predecessor Family from Collection?** is set to No. The list shows entity families that are predecessors for relationship definitions of the relationship family selected in the **Relationship ID** list.
Predecessor Family Key(s) section	Specifies the keys of the predecessor family records in the relationship.	You can select ⟳ to specify the output of a predecessor node in this section. The input value can be one of the following: • A single family key, where the predecessor records are in the same family. • A collection of family keys of the same length as the collection of predecessor entity keys, where the predecessor records may be in different families.
Successor Family ID list	Specifies the type of successor family records in the relationship.	This list is displayed when **Select Successor Family from Collection?** is set to No. The list shows successor entity families for relationship definitions of the relationship family selected in the **Relationship ID** list.
Successor Family Key(s) section	Specifies the keys of the successor family records in the relationship.	You can select ⟳ to specify the output of a predecessor node in this section. The input value can be one of the following: • A single family key, where the successor records are in the same family. • A collection of family keys of the same length as the collection of successor entity keys, where the successor records may be in different families.

Tip: Refer to the Family Policies documentation to see an example of this node.

Edit Entity Nodes in Policy Designer

An Edit Entity node represents an action to modify a record of a baseline or custom entity family. The fields in the specified record are updated with the values that you specify in the **Properties** window for the node. The fields for which you do not specify any value in the **Properties** window are not modified.

Note: The Edit Entity node displays only the entity families for which your APM system has active licenses, and for which you have edit permissions.

The Edit Entity node has no outputs.

Node Properties

The **Properties** window for an Edit Entity node contains the items described in the following table.

Item	Description	Notes
Family ID	Specifies the unique ID of the family associated with the record that must be modified.	The **Family ID** list contains all the baseline and custom entity families in APM for which you have update privileges.
Entity Key(s)	Specifies the entity key of the record that must be modified.	The entity key that you specify must belong to the family selected in the **Family ID** list.
Auto-map field values	Specifies whether the field values for the records must be automatically updated from a collection.	If the values that you want to specify for the record are part of a collection, you can use this option to specify the values for the fields of the record. If you use this option to specify the values for only some fields of the record, you must manually specify the values for the remaining fields.

Item	Description	Notes
Field values collection	Specifies the collection that contains the values that you want to map to the records. **Note:** If you specify the result of a query, make sure you run the query in unformatted mode.	This section is enabled only when you select **Yes** for the **Auto-map field values** option. The records that are modified is determined as follows: • If the collection that you specify contains a column with the name ENTY_KEY, the records specified in the **Entity Key(s)** section that have matching Entity Keys are automatically modified. • If the collection does not contain a column with the name ENTY_KEY, all records specified in the **Entity Key(s)** section are updated with values from the first row in the collection. Depending on the source of the collection, the fields of a record are automatically updated with the collection values in the following ways: • For a collection that is created using a query, if the label of a column matches the Field ID of a field, the field in each record is updated with the corresponding value of the column. For example, if the Field ID of the Reading Taken Date field in the records is MI_READING0_RDG_TAKEN_DT_D, the field in each record is updated with the corresponding value of the collection column whose label is MI_READING0_RDG_TAKEN_DT_D. • For a collection that is not created using a query, if the Column ID of a column matches the Field ID of a field, the field in each record is updated with the corresponding value of the column. For example, if the Field ID of the Reading Taken Date field in the records is MI_READING0_RDG_TAKEN_DT_D, the field in each record is updated with the corresponding value of the collection column whose Column ID is MI_READING0_RDG_TAKEN_DT_D. **Note:** If the Field ID of a field does not match the Column ID or label of any column in the collection, the value in the field is not modified.

| Item	Description	Notes
Entity Key Column	Specifies the collection column that contains the entity keys of the records that you want to modify. **Note:** If you mapped the field values collection from an R Script node or a Sub Policy node, no values are available for selection in this drop-down list box.	This section is enabled only when you select **Yes** for the **Auto-map field values** option. Depending on the value that you specify in this drop-down list box, the records are modified as follows: • If any value in the specified entity key column of the field values collection matches an entity key specified in the **Entity Key(s)** section, the record associated with the entity key is automatically updated with the values in the collection row corresponding to the entity key. • If the values in the specified entity key column of the field values collection do not match any entity key specified in the **Entity Key(s)** section, the records are not modified. • If you do not select a value in this drop-down list box and if the field values collection contains a column with the name ENTY_KEY, the values of the **ENTY_KEY** column are compared with the entity keys specified in the **Entity Key(s)** section, and the associated records are updated accordingly. • If the collection has only a single row, all records associated with the entity keys specified in the **Entity Key(s)** section are updated with the values in that row.
+	Adds a new row to the **Properties** window.	Each row represents a field that you want to update in the record. **Note:** If you specify a value for a field that is configured to be updated from a collection, the value that you specify takes precedence over the corresponding value in the collection.

Item	Description	Notes
Field	Specifies a field that you want to update in the record.	This list contains both baseline and custom fields of the record.
Value	Specifies the new value for the corresponding field.	If a field has a complex behavior defined by the field-level rules (for example, rules for valid values) and field-level behaviors, this behavior will not be reflected in the **Properties** window or detected during policy validation. Therefore, make sure that the values you specify are valid according to the baseline or custom field-level rules for the corresponding field. If a field value is defined by a system code, you must specify the system code in this field and not the value that is displayed to the user. **Note:** Irrespective of the Unit of Measure (UOM) Conversion Set configured for your user account, the value that you specify in this field is considered to be in the base UOM of the field.

Edit Entity node

In this example, the following queries are used:

- LR Entity Keys Query: Returns the Entity Keys of the Lubrication Requirement records related to the Lubrication Requirement Template.
- LRT Fields Query: Returns the current values in some fields of the Lubrication Requirement Template record specified in a policy instance. The Column IDs in the query are defined to match the Field IDs of the Lubrication Requirement family.

As shown in the following image of the **Properties** window for the Edit Entity node, the results of the LR Entity Keys Query provide the entity keys of the Lubrication Requirement records that will be updated. The node is additionally configured to automatically update the fields of the Lubrication Requirement records with the results of the LRT Fields Query. No field values are specified in addition to the values to be updated from the LRT Fields Query.
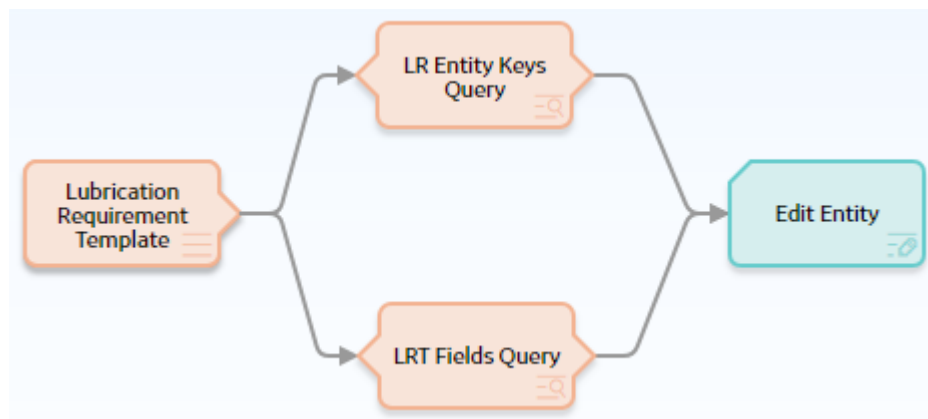
On executing this policy, all Lubrication Requirements related to the Lubrication Requirement Template that is specified in the policy instance are updated with the new values from the Lubrication Requirement Template record.

Email Contact Nodes in Policy Designer

An Email Contact node represents an action to send an email message. When an Email Contact node is executed, an email message with a summary of the policy execution will be sent to the specified recipient(s). Emails sent via this node use the **From** address specified in the **Email Settings** section of Operations Manager. You can define a subject, message, and a table of results to be included in the email.

The Email Contact node has no outputs.

Node Properties

The **Properties** window for an Email Contact node contains the items that are described in the following table.

Item	Description	Notes
To Address	The email address(es) to which the message should be sent.	You can select to specify the output of a predecessor node in this section. When you specify a constant value, you can enter one or more email addresses directly in the text box, or you can select the button to select a recipient via the **Choose Users** window. If you enter more than one email address in the text box, each email address must be separated by a comma or semicolon. The format of the email address that you enter is validated; if the format is incorrect, an error message appears.
Message	Content that you want to include in the email message in addition to the summary of the policy execution (which is included by default).	You can select to specify the output of a predecessor node in this section.
Subject	The subject line that you want to appear in the email message. The length is limited to 255 characters.	You can select to specify the output of a predecessor node in this section. If you do not specify a value in this section, the email subject line is set to "Notification from Policy <Name>").
Show Policy logic summary?	Specifies whether the Policy Logic Summary is included in the email message or not.	The option **Yes** is selected by default. In case you do not need the summary displayed in the email message, you can select the option **No**.
Include Policy name in email body	Specifies whether the Policy name is included in the email message or not.	The check box is selected by default. If you do not need the Policy name displayed in the email body, clear the check box.
Include Time Zone message in email body	Specifies whether the Policy email body should include default Time Zone message or not.	The check box is selected by default. If you do not need default Time Zone displayed in the email body, clear the check box.

Item	Description	Notes
Results Table	Specifies how the query result is shared in an email message.	You can select from the list to **Embed in email message**, **Attach to email as file**, or **Both.** **Note:** The attachment is shared only in the CSV format. **Note:** Attached files are stored in APM as Reference Document records. If the attached file size exceeds the configured limit, the Reference Document is created but the file is not attached to the email message. You need to access APM to open the Reference Document.
Include query/collection results section	The query or collection results are included in the email message. By default, the results are displayed in the table format.	You can select query or collection from the drop down ▶ . You can select 🖉 to specify the output of a predecessor node in this section. **Note:** Numeric values are displayed to two decimal places. Dates and times are displayed in UTC. **Note:** Tables embedded in the email body are truncated to display the first 500 rows only. **Note:** To avoid display issues with some email clients, it is recommended to limit the number of columns in the embedded table to 30.

Tip: Click here to see an example of this node used within a complete policy model.

Return Value Nodes in Policy Designer

A Return Value node represents an action to return a specific value. You can use this node for a variety of reason, such as:

- To include specified values in the execution results summary for a policy.
- While designing a policy, to show the result of a specific node to verify that results are as expected in cases where the value would not otherwise be displayed in the validation results (because validation results only display the values that are used by a subsequent node). Once the policy logic is fully validated, you can remove the Return Value nodes to make the policy smaller and reduce the amount of information included in the execution history summary field.
- When used within a baseline policy for a APM analysis (for example, Risk Based Inspection), to pass values from the policy back to the analysis process that triggers the policy.
- When used within a sub policy, to define the output values of the Sub Policy node.

The Return Value node has no outputs.

Note: The name that you specify for a Return Value node should be unique unless the policy logic is configured such that only one Return Value node with a given name can be executed each time that the policy is executed.

Node Properties

The **Properties** window for a Return Value node contains the items that are described in the following table.

Item	Description	Notes
Return Value section	Specifies the value that you want to return.	You can select ✎ to specify the output of a predecessor node in this section.

> **Return Value node**
>
> The following example illustrates how you can use a Return Value node to include relevant values in the execution results summary of the policy. Consider the following nodes and connections.
>
>
>
> In this policy, the two Return Value nodes are configured to return the Accumulated Time and Count values, respectively, from the Threshold Statistics node. Because the Return Value nodes are included in this policy, the Accumulated Time and Count values are easily accessible in the execution results summary of the policy.

Rule Nodes in Policy Designer

A Rule node represents an action to execute a custom rule that provides functionality that the existing nodes in Policy Designer do not provide.

The inputs and outputs of a Rule node are defined by a rule that will be executed when the policy is executed.

Important: The improper implementation of a rule through the Rule node could severely impact the performance of policy executions. If you want to use this node, you should contact APM for additional instructions and assistance.

Node Properties

The **Properties** window for a Rule node contains the items that are described in the following table.

Item	Description	Notes
Rule Path box	Specifies the Catalog path to the rule project that contains the rule that will be executed when the policy is executed.	You can enter the path manually, or you can browse to it by selecting the 🔍 button.
Rule Class list	Specifies the class containing the specified rule.	None.
Additional sections corresponding to the inputs defined by the specified rule.	Specifies the values for the inputs defined by the rule.	You can select ✏️ to specify the output of a predecessor node in this section.

State Transition Nodes

A State Transition node represents an action to change the state of one or more records in any baseline or custom entity family. You can specify either the state to which you want to move the records or the state transition that you want to perform.

A State Transition node has no outputs.

Table 8: Node Properties

Item	Description	Notes
the EntityKey(s) section	Specifies the entity keys of the records for which the state will be transitioned.	You can select ✎ to specify the output of a predecessor node in this section. You can use this node to transition the state of a collection of records from different families, provided that the target state ID or transition ID is the same for all the state transitions you want to implement.

| Item	Description	Notes
Select Family from Collection section	Allows you to select how the family of the records that will be deleted is defined.	The default setting is **Yes**. If you select **No**, the **Family Key(s)** section is updated to allow you to select a single Family ID from a list.

Item	Description	Notes
Family Key(s) section	Specifies the family keys of the records specified in the Entity Keys section.	You can select to specify the output of a predecessor node in this section. You can choose one of the following input values: • Single family key: This is applicable when all the records to be transitioned are in the same family. • Collection of family keys of the same length as the collection of entity keys: This is applicable when the records to be transitioned may belong to different families.
State Transition Mode	Specifies the mode of operation of the node (Do Operation or Advance to State).	The Do Operation mode is selected by default.

Item	Description	Notes
Operation ID	The ID of the state transition operation you want to apply to the records.	You can select ✎ to specify the output of a predecessor node in this section. This field is enabled when the Do Operation state transition mode is selected.
The State ID section	The ID of the state to which you want to transition the records.	You can select ✎ to specify the output of a predecessor node in this section. This field is enabled when the Advance to State state transition mode is selected.

The State Transition Node

The following example shows how to use the State Transition node to change the state of a collection of records. A policy like this can be scheduled to run on a regular schedule to automate the identification of records that are ready for transition to the next state, enabling you to focus on higher value tasks.
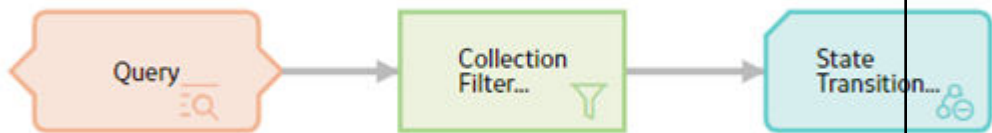
In this example, a query is executed to find the entity and family keys of Performance Recommendation records in the Proposed state, and a Collection Filter node is used to select the records that meet the conditions for transitioning to the Pending Approval state. Note that the records are not required to be in the same Performance Recommendation sub-family; they can be in any sub-family that shares the parent

family's state machine, as shown in the following image.

The State Transition node is configured to operate on a collection, and the family keys and entity keys inputs are mapped from the Collection Filter node. The node is configured to use the Advance to State state transition mode and specifies the state ID for the Pending Approval state, which is **MI_PENDINGAPPROVAL**. The following image shows the properties window for the State Transition node:

Sub Policy Nodes in Policy Designer

A Sub Policy node is an Action node that you can use in the policy model to pass values from one policy (the calling policy) to be evaluated or acted on by a different policy (the sub policy). Results from the sub policy may be returned to the calling policy for further evaluation or action.

A sub policy can be created for commonly used policy logic to reduce policy development time and ensure consistency. It also allows large policy models to be broken down into a series of sub policies, which are easier to understand and perform better in the Policy Designer user interface.

Note: You can configure a Sub Policy node to pass values to only those policies for which you have Designer or User permissions.

Node Properties

The **Properties** window for a Sub Policy node contains items that are described in the following table.

Item	Description	Note
Policy	Name of the sub policy to call.	None.
Iterate Over Collection?	Specifies whether the sub policy must be executed for each row in a *collection* that is used as an input for the node.	None.
Execute Specific Instance?	Specifies whether a specific instance associated with the sub policy must be executed.	If you select **Yes**, in the **Instance Id** box that appears, you can specify the name of the sub policy instance that you want to be executed.
Additional sections corresponding to the inputs defined by the Point Value nodes contained in the sub policy.	Specifies the values to be represented by the Point Value nodes in the sub policy.	A Point Value node in the sub policy can represent a single value. If the Sub Policy node is configured to iterate the execution of the associated sub policy, you can specify a column of an input collection as the input for a Point Value node in the sub policy.

Working with Sub Policies

You must configure the sub policy such that all the required inputs are defined as single value Point Value nodes (that is, data frame inputs are not supported). Results from the sub policy may be passed back to the calling policy by using Return Value nodes.

If you configure the sub policy to be executed for each row of an input collection, the output of the sub policy node is a collection that contains one column for each Return Value node, with rows containing the values for each execution.

If you configure the sub policy for a single execution, the outputs of the sub policy node include:

- The value of each Return Value node in the sub policy as an individual output.
- A collection with two columns, Name and Value, which contain values from all the Return Value nodes in the sub policy.

Important:

- Make sure that the sub policies called by a policy are active. If a sub policy is inactive, a message appears in the notification bar and the calling policy cannot be activated.
- Changes to Point Value or Return Value nodes, or the name of a sub policy will not be automatically reflected in calling policies that use the sub policy. It is possible to specify additional security for a sub policy to minimize the inadvertent impact to other policies.
- While it is possible for a sub policy to act as a calling policy for another sub policy, it is essential that a circular execution path is not created. In other words, if policy A calls policy B, which calls policy C, then policy C must not call policy A or policy B. If such a circular execution path is detected while you are editing the policy, a message appears in the notification bar. However, a circular execution path where the sub policy acts on the same entity that triggers the calling policy is not detected and no message appears in the notification bar. Make sure that such a circular execution path is not created in a policy.
- If an error occurs during the execution of a sub policy, execution of the calling policy fails, irrespective of the execution status of other sub policies called by the policy.
- If a Return Value node in a sub policy represents a collection, other nodes in the calling policy model that use the output of the Sub Policy node cannot process the values in the columns of the collection.
- You cannot use a Create Event node paired with a Close Event node in a sub policy to create and close Policy Events with duration, unless you are calling a specific instance of the sub policy because the Close Event node depends on the open Policy Event that is linked to the policy instance.

238

Sub Policy Node Execution Details

After you validate or execute a policy that contains a Sub Policy node, you can select the Sub Policy node to view the execution details of the node in the **Node Execution Details** window. Along with viewing the execution details of the node, you can select the **View Execution Details** link in the **Node Execution Details** window to view the policy model and execution details of the sub policy that is mapped to the node.

The following image is an example of the **Node Execution Details** window for a Sub Policy node:
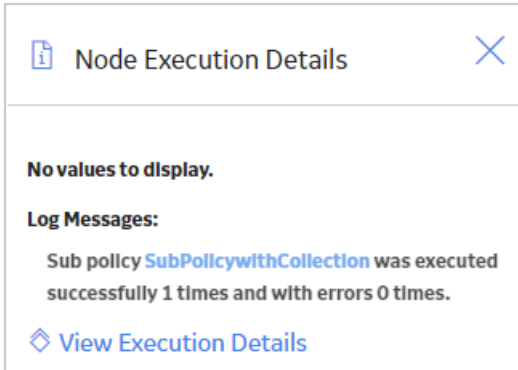

The following image is an example of the **Sub policy execution details for policy** window that appears when you select the **View Execution Details** link in the **Node Execution Details** window for a Sub Policy node:
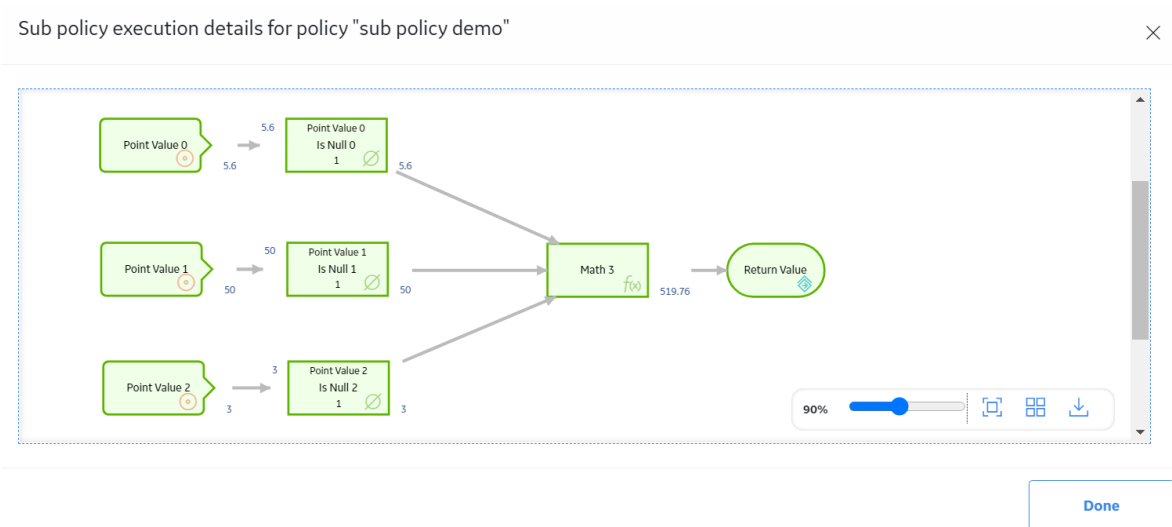
Sub Policy Node Configured for Single Execution

The following is an example of how a Sub Policy node can be used to implement a standard calculation method which could be applied in any number of other policies. Consider the following nodes and connections:
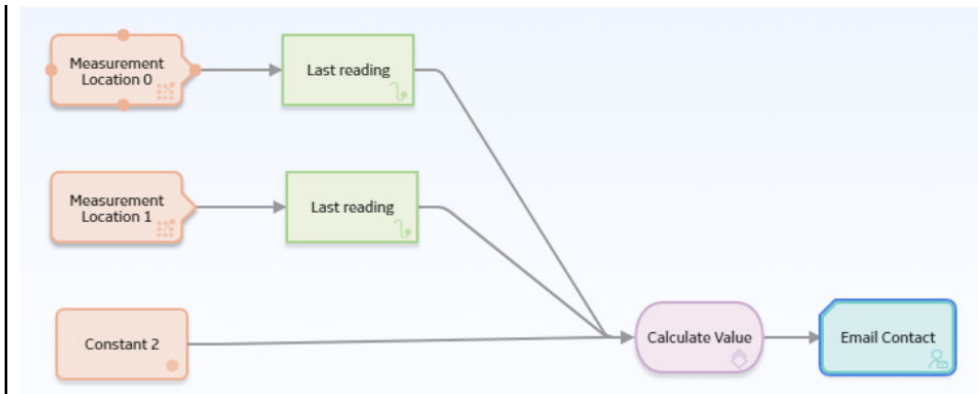
In this example, the latest reading values from two measurement locations and a constant value are passed into a sub policy, which calculates a value to be used in the Email node. The **Properties** window for the Sub Policy node is shown in the following image:
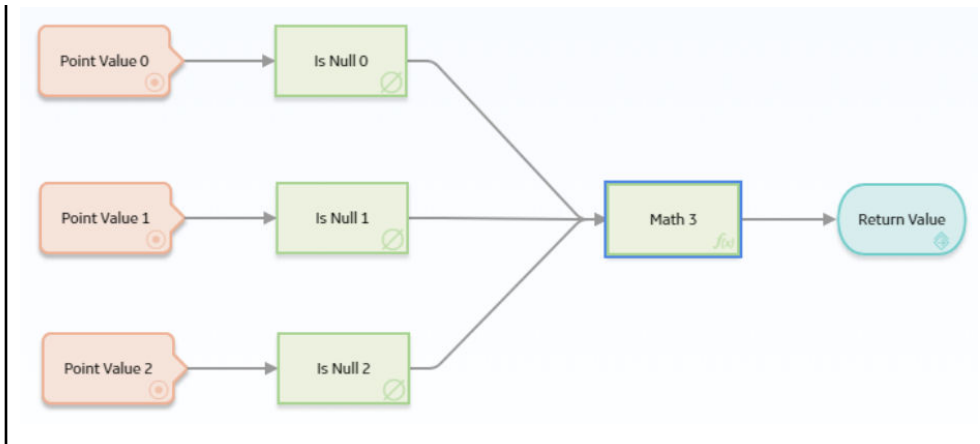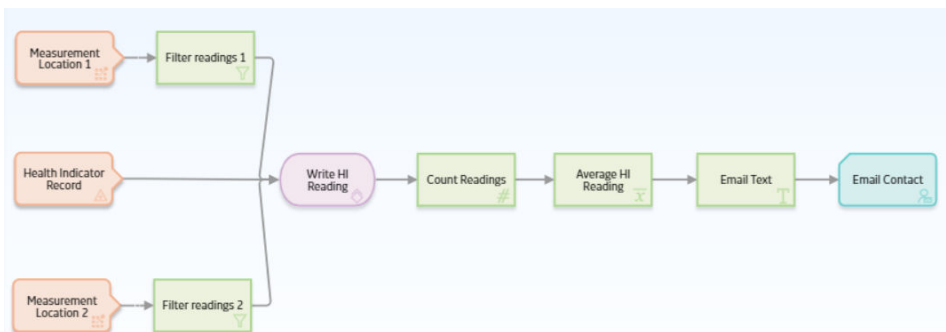

The sub policy used in the above example includes Is Null nodes which provide default values to be used if the calling policy does not supply an input value, as shown in the following image:

Sub Policy Node Configured for Iterated Execution

The following is an example of how a Sub Policy node can be used to add a collection of calculated values to a health indicator.

In this example, the Collection Filter nodes are used to filter the readings that are taken for the last two days from two Measurement Location nodes. An Entity node is used to define the health indicator record to which the readings must be added. The collections of readings and time stamps are passed to the sub policy and a single health indicator entity key value is used for each iteration of the sub policy. Each execution of the sub policy calculates a new value based on the readings from the two measurement locations and updates it to the health indicator. The following image shows the **Properties** window for the Sub Policy node:

The following image shows the sub policy used in the above example:

The sub policy contains four Point Value input nodes. The Reading Value 1, Reading Value 2, and Health Indicator Key Point Value nodes are configured to represent Decimal data type values, and the Reading Date/Time node is configured to represent a Time & Date data type value. Even though the calling policy is configured to pass a collection of values, each iteration of the sub policy will use a single value from each input collection.

The values represented by the Reading Value 1, Reading Value 2 nodes are passed to a Math node which calculates a new value to be written to the health indicator by the Add Value to Health Indicator node. The Return Value node is configured to return the single value output from the Math node. In the calling policy, the corresponding output from the Sub Policy node is a collection made up of the results from each iteration of the sub policy.

Baseline Policies

About Baseline Policies

APM provides the following policies that can be used as sub policies to perform calculations on dates:

- DateAdd
- DateDiff
- String Replace Sub-Policy on page 245
- Concatenate Text Email Example on page 246

Note: You cannot modify the name of a baseline policy.

DateAdd Policy

You can use the DateAdd policy as a sub policy to add a time interval to a time stamp value. The DateAdd policy is configured as an alternative for the date add functions in queries and R scripts.

The DateAdd policy uses the following inputs to perform the addition:

- Time stamp to which you want to add the time interval.
- Time interval that you want to add to the time stamp.
- Unit of time in which the interval is defined.

How DateAdd Policy Works

The following image illustrates the DateAdd policy:
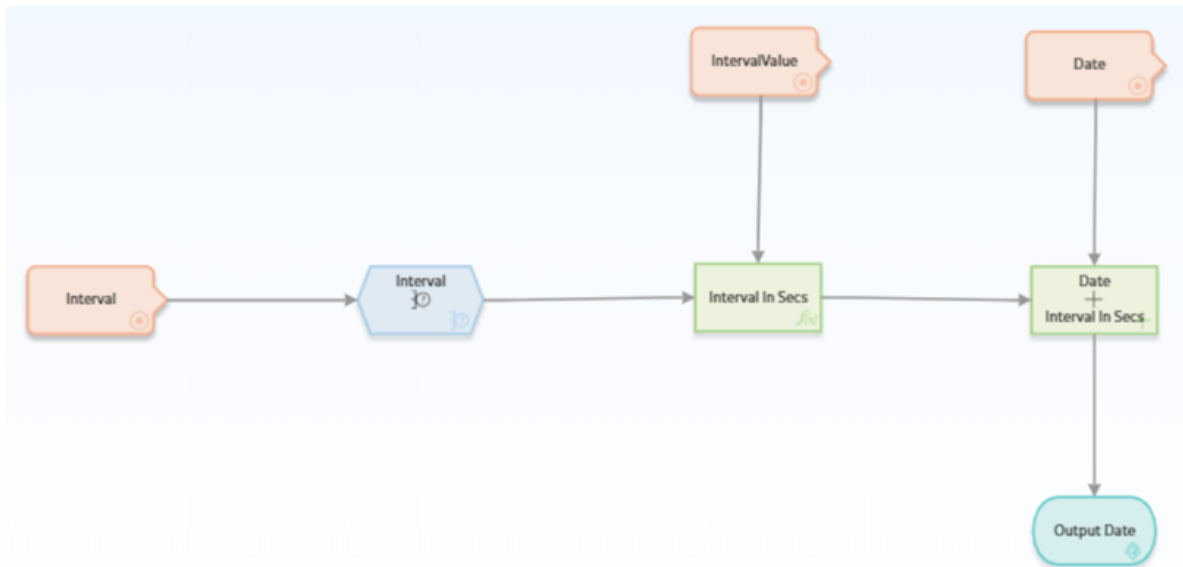
In this policy, the interval value, the unit of time in which the interval is defined, and the time stamp to which you want to add the time interval are passed as inputs from the calling policy, and they are represented by the Point Value nodes. The Case and Math node use the time interval value and its unit to calculate the number of seconds in the specified interval. The Add node then adds the number of seconds to the time stamp and the resultant date is passed back to the calling policy by the Return Value node.

DateDiff Policy

You can use the DateDiff policy as a sub policy to calculate the difference between dates in terms of the following units of time:

- Years
- Weeks
- Days
- Hours
- Minutes

You can use the DateDiff policy as an alternative for the date difference functions in queries and R Scripts to calculate time difference between dates.

How DateDiff Policy Works

The following image illustrates the DateDiff policy:
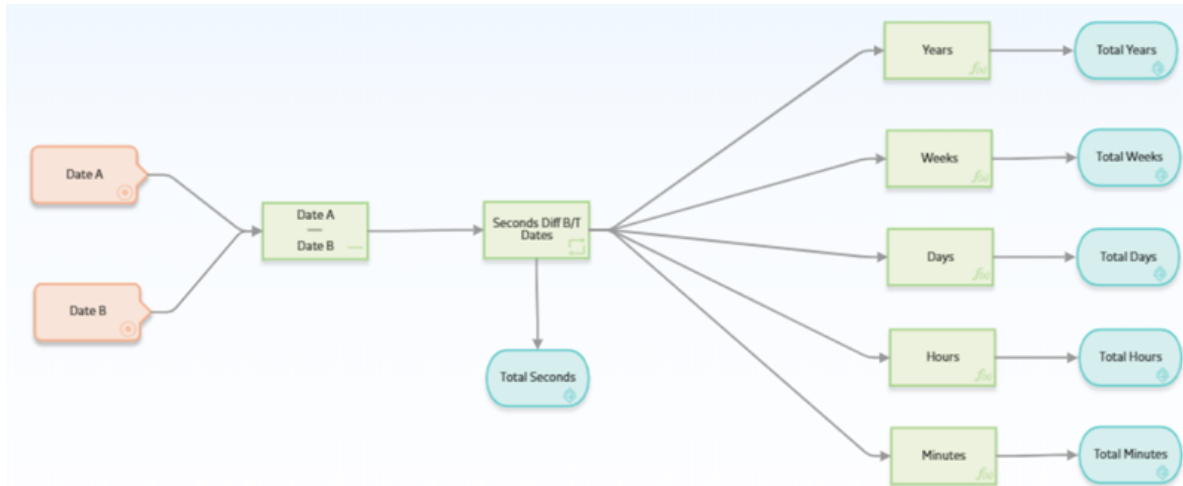
In this policy, a Subtract node is used along with a Convert Type node to calculate the number of seconds between dates. The dates are passed as inputs from the calling policy, and they are represented by the Point Value nodes in this policy. The number of seconds between the dates is then converted by the Math nodes to years, weeks, days, hours, and minutes and passed back to the calling policy via the Return Value nodes. Depending on the unit of time in which the difference between the dates is required in the calling policy, you can map the desired output from the Sub Policy node to the appropriate node in the calling policy.

String Replace Sub-Policy

You can use the String Replace Sub-Policy to replace a string with a different string within a single text value, or, by using the sub policy in a loop, a collection of text values.

The String Replace Sub-Policy provides one output value or collection: **Output String**.

The following table describes the inputs used by the String Replace Sub-Policy.

Input Value	Notes
Input String	Any text value.
Search String	The text value that you want to search for within the Input String. This input accepts Regular Expressions, allowing for more complex searches.
Replacement String	The text value with which you want to replace instances of the Search String within the Input String.
Ignore Case?	If true, the string replacement function ignores the case of characters in the input string when searching for matching strings to replace.

How String Replace Sub-Policy Works

The following image illustrates the String Replace Sub-Policy:
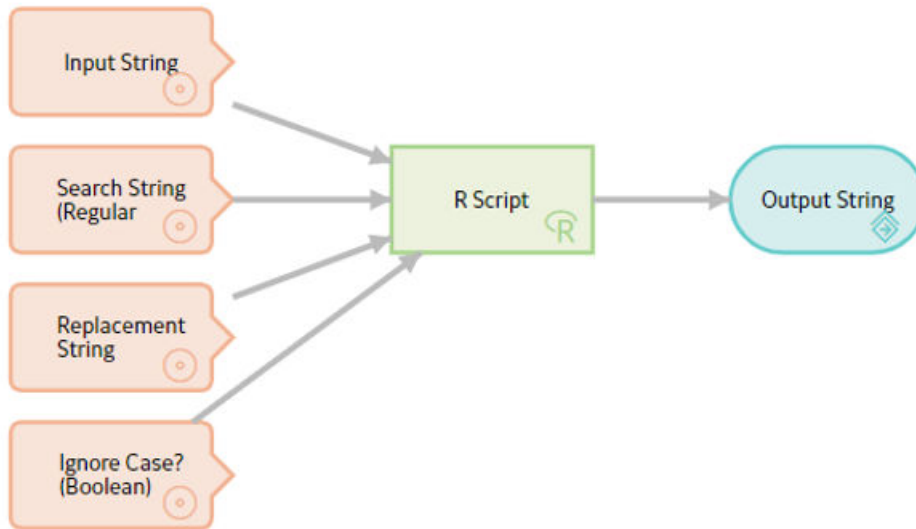
This policy uses the R Script `Public\Meridium\Modules\Policy Manager\RScripts\StringReplace`, which contains the following code:

```
outputText <- gsub(pattern, replacementText, inputText, ignore.case =
        as.logical(ignoreCase), perl = FALSE, fixed = FALSE, useBytes
= FALSE)
```

Each time the sub-policy is executed, the R script is evaluated. It uses the base R gsub() function to replace all instances of the Search String (represented by pattern in the R Script) in the Input String (represented by **inputText**), with the Replacement String (represented by **replacementText**). If the value of Ignore Case? evaluates to True, the search and replace is not case sensitive.

For more advanced use cases, you can use a Regular Expression as the Search String. For example, if you wanted to replace any numeric value in the Input String with the word "number", you would enter `\d+` as the Search String and `number` as the Replacement String.

Concatenate Text Email Example

You can use the Concatenate Text Email Example policy to send an email message to all the users in a security group, which means that there is no requirement to modify the policy when users are added to or removed from the group.

This policy is intended as an example of how to use the associated R Script to concatenate a collection of text values, using an optional delimiter string. The R Script can be reused as-is in other policies that require similar functionality, or you can copy and modify the R Script and example policy to achieve your objective.

How Concatenate Text Email Example Works

The following image illustrates the Concatenate Text Email Example policy:
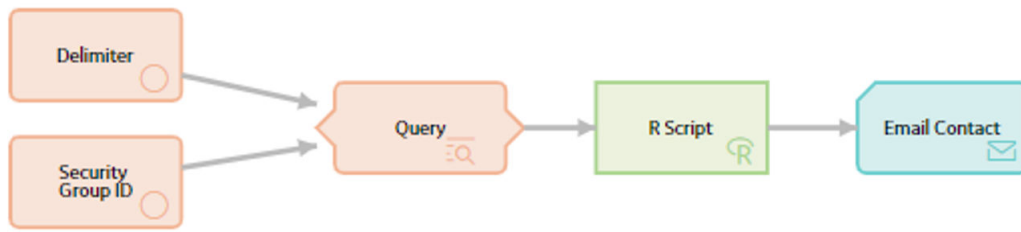
This policy uses the query **Public\Meridium\Modules\Policy Manager\Queries\Email Addresses** for a Security Group to retrieve the email addresses of all users in the security group with the ID specified in the Security Group ID Constant node.

The policy then uses the R Script **Public\Meridium\Modules\PolicyManager\RScripts \ConcatenateText** to concatenate the list of email addresses, adding the value specified in the Delimiter Constant node, a comma, between each email address.The R Script contains the following code:

```
concatenatedText <- paste(inputTextCollection, collapse = delimiter)
```

The Concatenated Text output from the R Script node is then used as the input to the To Address section of the Email Contact node.

Glossary

collection

A set of results from a Query node or a set of readings associated with input nodes such as Measurement Location, Health Indicator, or OT Connect Tag .

policy instances

Identifies the records to which logic in the policy model will be applied.

entity key

The Entity Key output can be used in subsequent action nodes, such as the Create Relationship node, but will return a value of 0 if used in a calculation or a Return Value node, since the Entity Key is not available until the record is saved at the end of policy execution.

policy model

The nodes and connections that define the policy logic.

primary node

The node that represents the primary record in a policy instance.

primary record

The record in a policy instance to which most or all other records in the instance are linked.