Asset Performance Management
APM Classic
V4.6.11.0.0

Family Policies

Contents

Chapter 6: Admin   33

Chapter 7: Reference   36

Copyright Digital, part of GE Vernova

Chapter
1

Overview

Topics:

- Rules and Policies
- Family Policy Workflow

Rules and Policies

In APM, you can use rules and family policies to determine how records in the APM database will behave when you work with those records in APM.

Rules consist of code that is written in Visual Basic.Net (VB.Net), a programming language that is compatible with the language in which APM is written. If you have sufficient knowledge of writing VB.Net code, you can write rule code to be executed when certain changes occur in the APM database. You can write family-level or field-level rules.

As an alternative to family-level rules, you can use family policies to configure certain actions to occur when a record changes in the APM database. Family policies are created in a user interface where knowledge of Visual Basic.Net (VB.Net) is not required.

Within the **Rules and Policies** section of **Family Management**, you must specify whether you want to use rules, family policies, or neither for a particular family.

Note: For a single family, you can write family-level rules or family policies, not both. You can, however, use the Baseline Rule node in a family policy to execute any existing APM baseline rules that correspond to the policy's family and trigger. You can also use field-level rules and family policies for the same family.

Family Policy Workflow

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

1. Create a new family 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. Run the validation process to confirm that the policy logic is working correctly.

Chapter
2

Policy Management

**Topics:**

About Family Policies

You can use family policies to configure certain actions to occur when a record changes in the APM database. For example, you can specify that after a Reading record is added to the database, it should be linked to a Measurement Location record.

Note: For a single family, you can write family-level rules or family policies, not both. You can, however, use the Baseline Rule node in a family policy to execute any existing APM baseline rules that correspond to the policy's family and trigger.

When you create a family policy, you will configure a policy model to represent the inputs, logic, and actions that you want to execute when the corresponding trigger occurs. You can create up to six family policies for a single family, one for each supported type of trigger.

Policy Model

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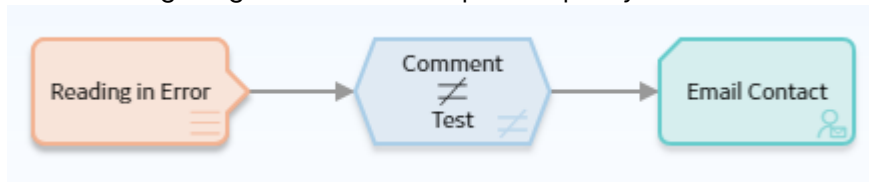
A policy model is made up of nodes and connections that define the policy logic. Specifically, the nodes in a model represent:

- The items that you want to monitor (for example, Reading in Error records).
- The conditions that should trigger actions to be taken (for example, a comment is something other than `test`).
- The actions that should be taken (for example, 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 basic principles for working with a policy model.

Family Policy Triggers

For any family, you can create one family policy for each available trigger. APM supports the following triggers for family policies.

Trigger	Description
Before Insert	Executes the policy before a record is created.
After Insert	Executes the policy after a record is created.
Before Update	Executes the policy after changes have been made to a record, before those changes are saved to the database.
After Update	Executes the policy after changes to a record have been saved.

Trigger	Description
Before Delete	Executes the policy before a record is deleted.
After Delete	Executes the policy after a record has been deleted.

Create a Family Policy

Procedure

1. In the module navigation menu, select **Admin** > **Configuration Manager** > **Family Management**.

    The **Family Management** page appears.
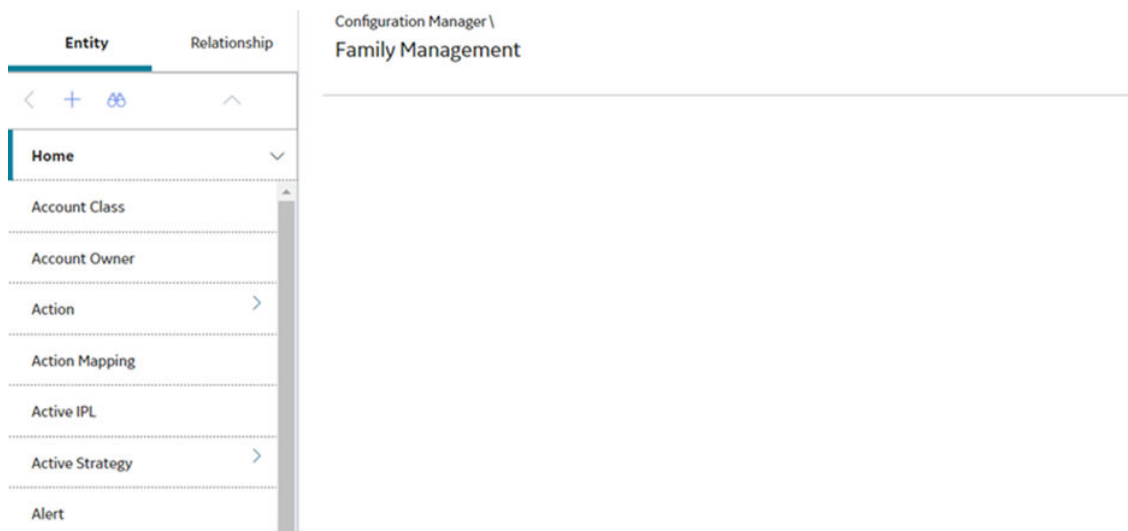
2. Select the family for which you want to create a family policy.

    Note:

    - If you want to access a new family from Policy Designer, you must first sign out of APM, and then sign in.
    - If a user is signed into APM at the time that you created a family, they cannot access the family unless they sign out of APM and then sign in.

    The workspace for the selected family appears.
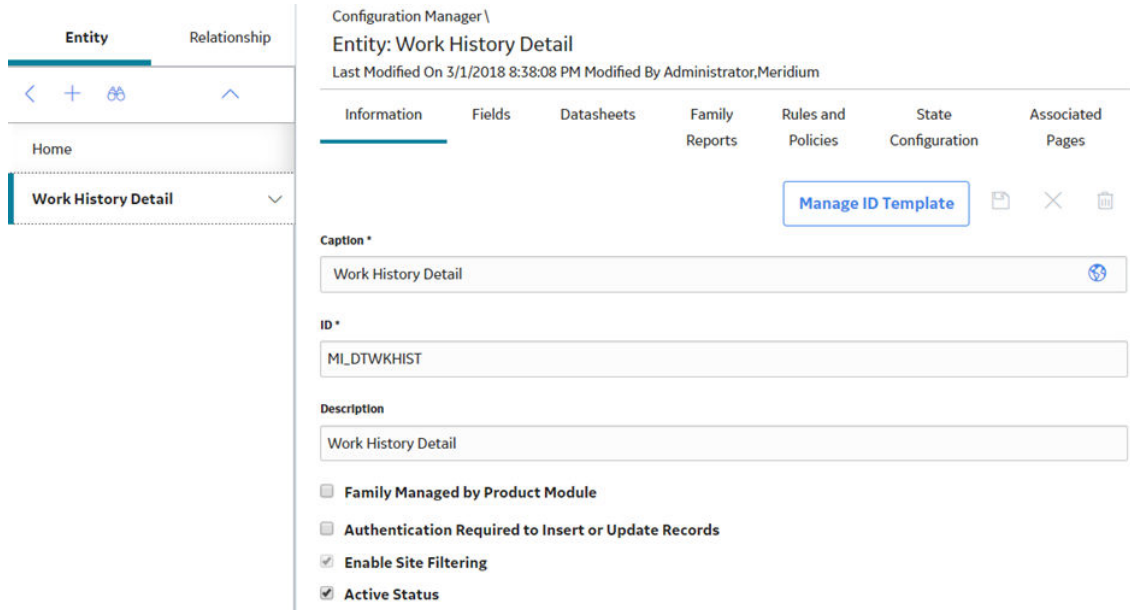
3. Select the **Rules and Policies** tab.

   The **Rules and Policies** section appears.
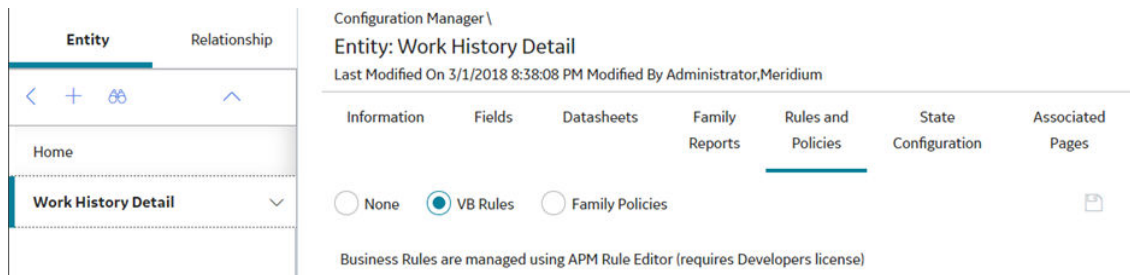
4. Select the **Family Policies** option, and then select 🖫.

   Important: If you select the **VB Rules** option, family policies will be used instead of any configured rules for the family.

   A list of the possible family policy triggers appears.
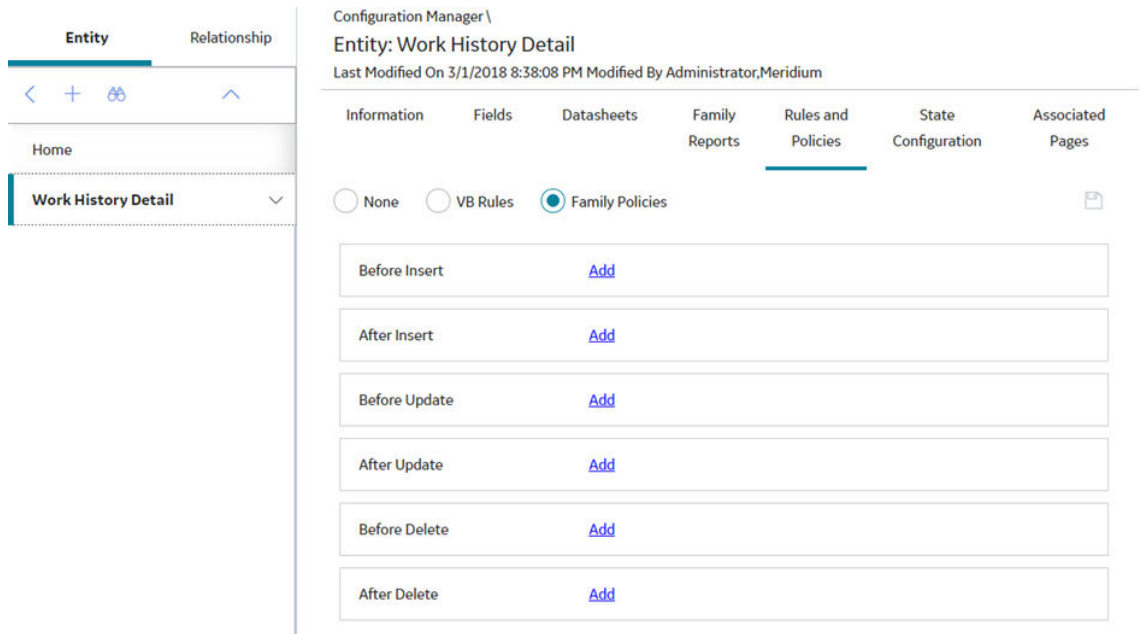
5. Next to the trigger for which you want to create a family policy, select **Add**.

   The **Family Policy** page appears, displaying the **Design** workspace where you can add nodes to the model canvas.
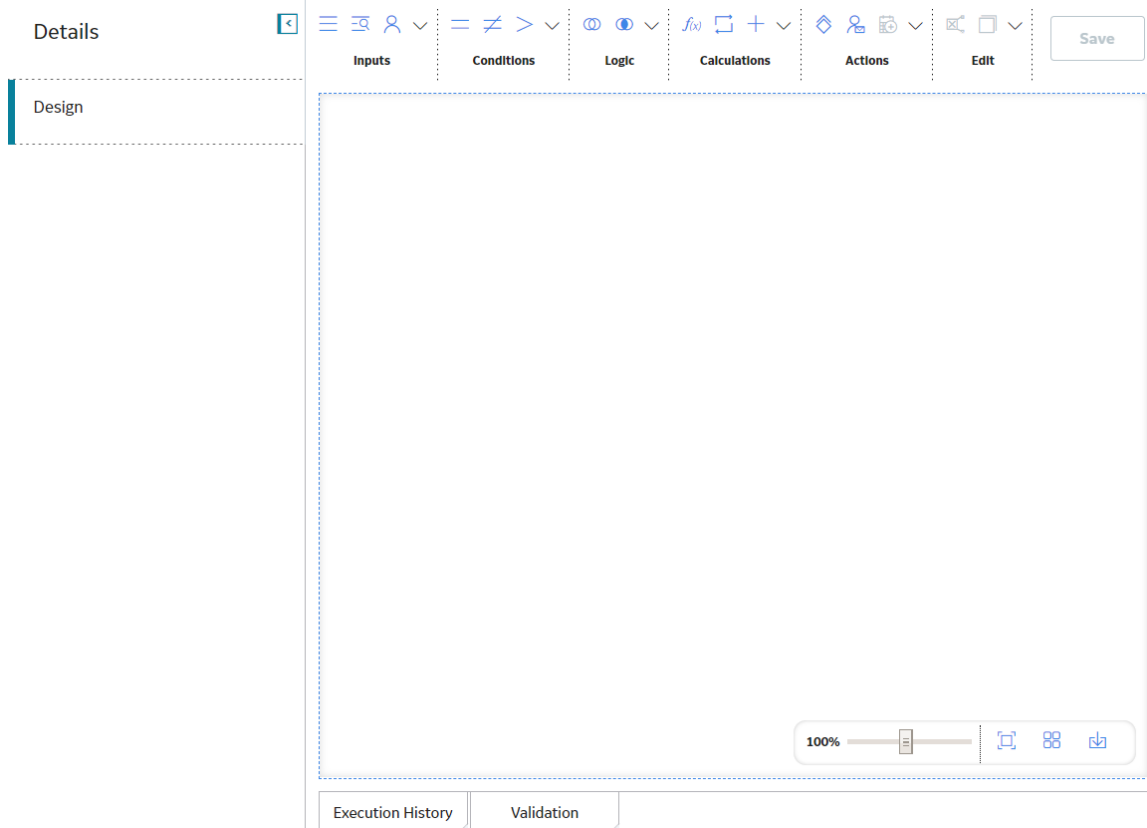
6. In the **Details** workspace, enter a brief description for the new family policy.

7. Select 💾.

A family policy is created.

Access a Family Policy

Procedure

1. Access **Configuration Manager** .
2. Select **Family Management**.

   The **Family Management** page appears.
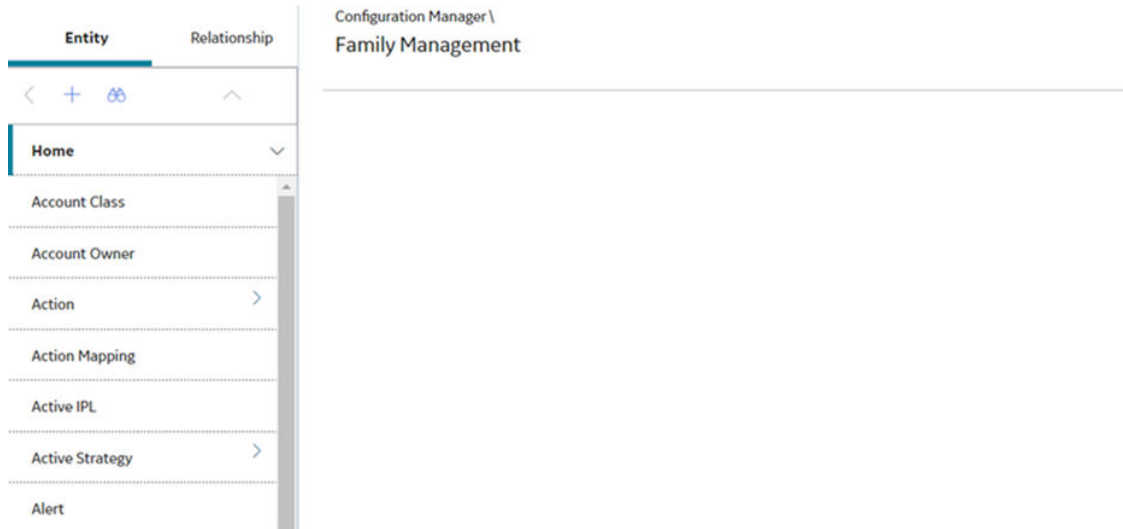
3. Select the family associated with the family policy that you want to open.

   The workspace for the selected family appears.
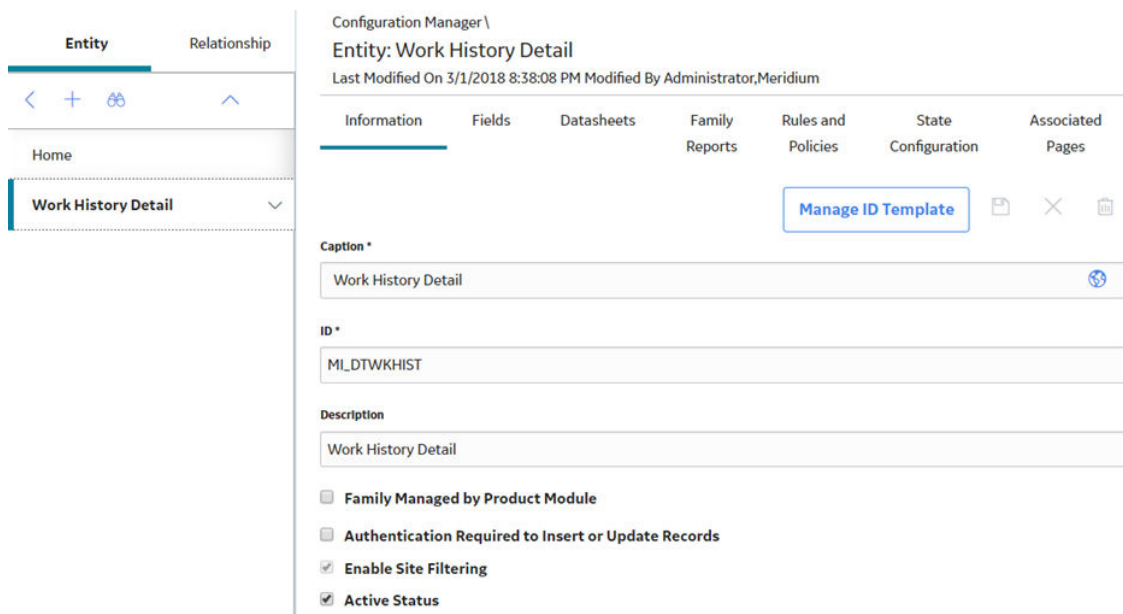
4. At the top of the workspace, select **Rules and Policies**.

The **Rules and Policies** section appears, displaying a list of the possible family policy triggers.
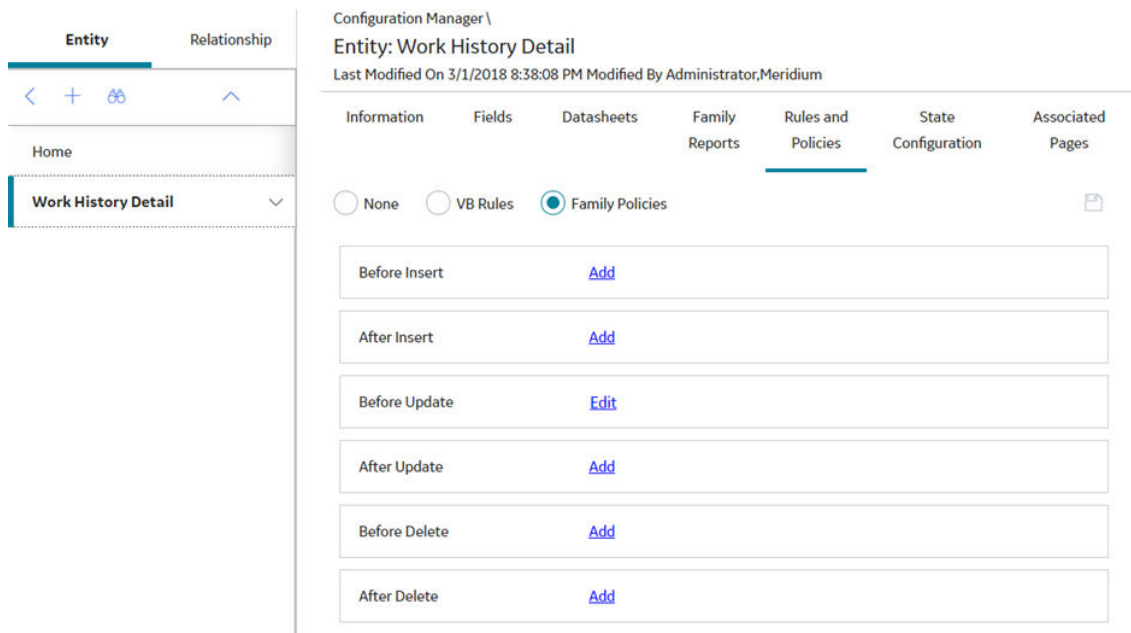
5. Next to the trigger associated with the family policy that you want to open, select the **Edit** link.

   The **Family Policy** page appears, displaying the **Design** workspace for the family policy that you selected.

Refresh Metadata for Family 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 Family Policy

About This Task

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

About This Task

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. 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 added during the deployment of the respective module), you can use these steps to revert the policy to baseline.

Procedure

1. Access the policy that you want to revert to the baseline version.
2. At the top of 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
3

Policy Models

Topics:

Policy Model Basic Principles - Family Policies

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:

- The first node in a policy model must be an Input node rather than a Query node.
- 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.
- A family policy cannot be saved if it contains any errors.
- 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 Reading in Error node represents records in the Reading in Error family. When this policy is executed, if the Comment field in the Reading in Error record that triggered the policy is something other than test, the APM system will send an email message to the email address that is specified in the **Properties** window for the Email Contact node.

The following table explains each of the policy model basic principles in the context of this example.

Policy Model Principle	Example
The first node in a policy model must be an Input node rather than a Query node.	The Reading in Error node is a Current Entity node, which is a type of Input node.
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 Comment from the Reading in Error 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 test 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.

Policy Model Principle	Example
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 Reading in Error 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 Reading in Error 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, 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 nodes were connected at different points.
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 in Family Policies

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.
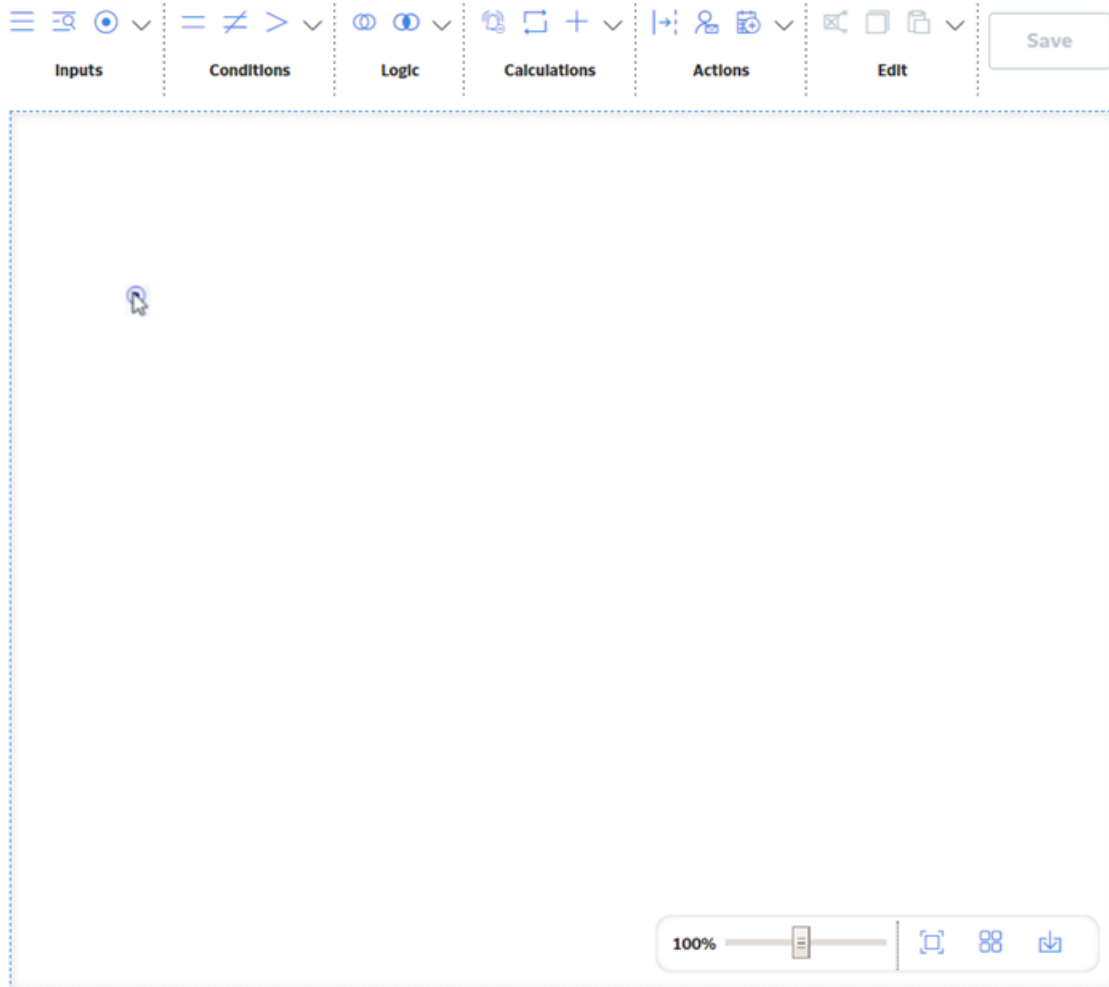
Tip: In each section of the toolbar, you can select the ⌄ button to see additional nodes and to see what each button represents.

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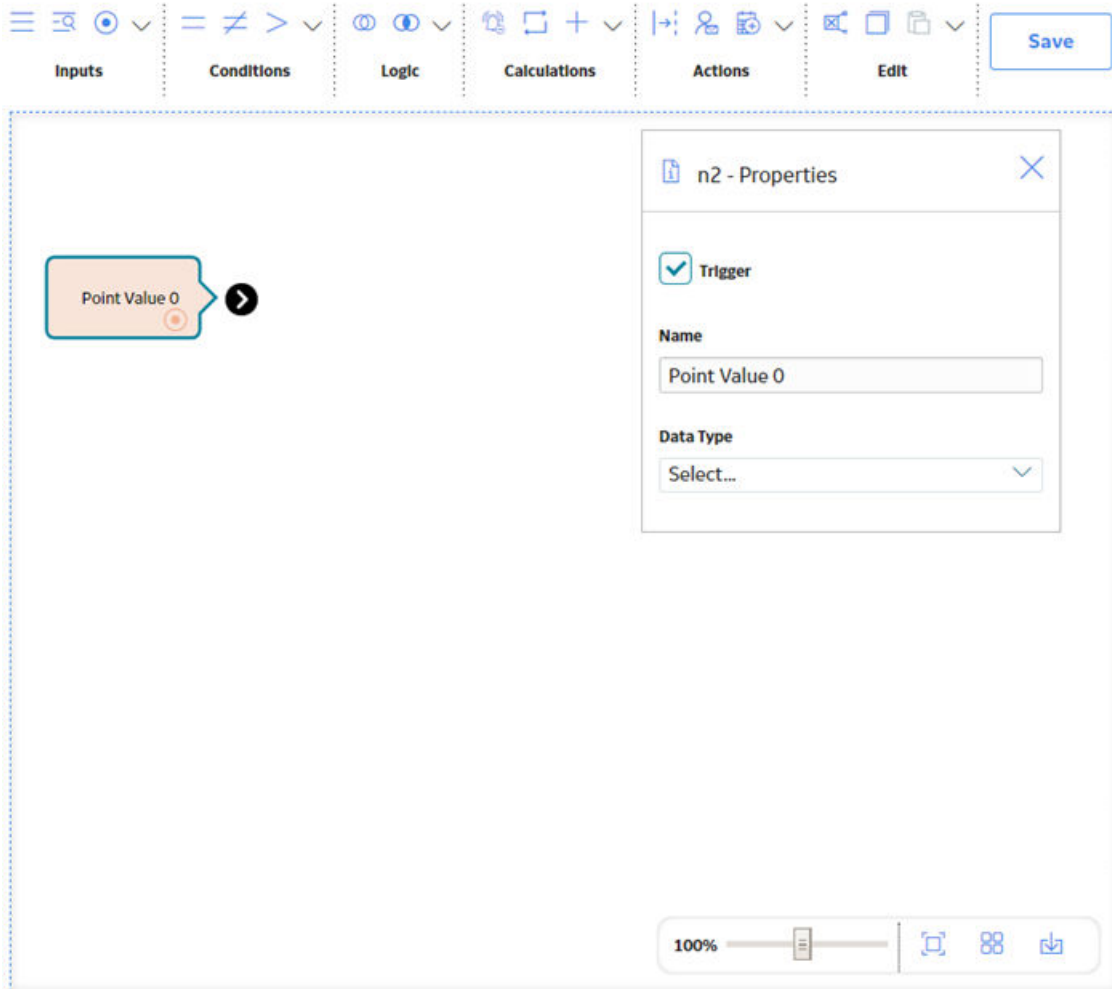
4. Select **Save**.
The policy is saved.

Next Steps

- Connect Nodes in a Policy Model
- Configure Node Properties

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.
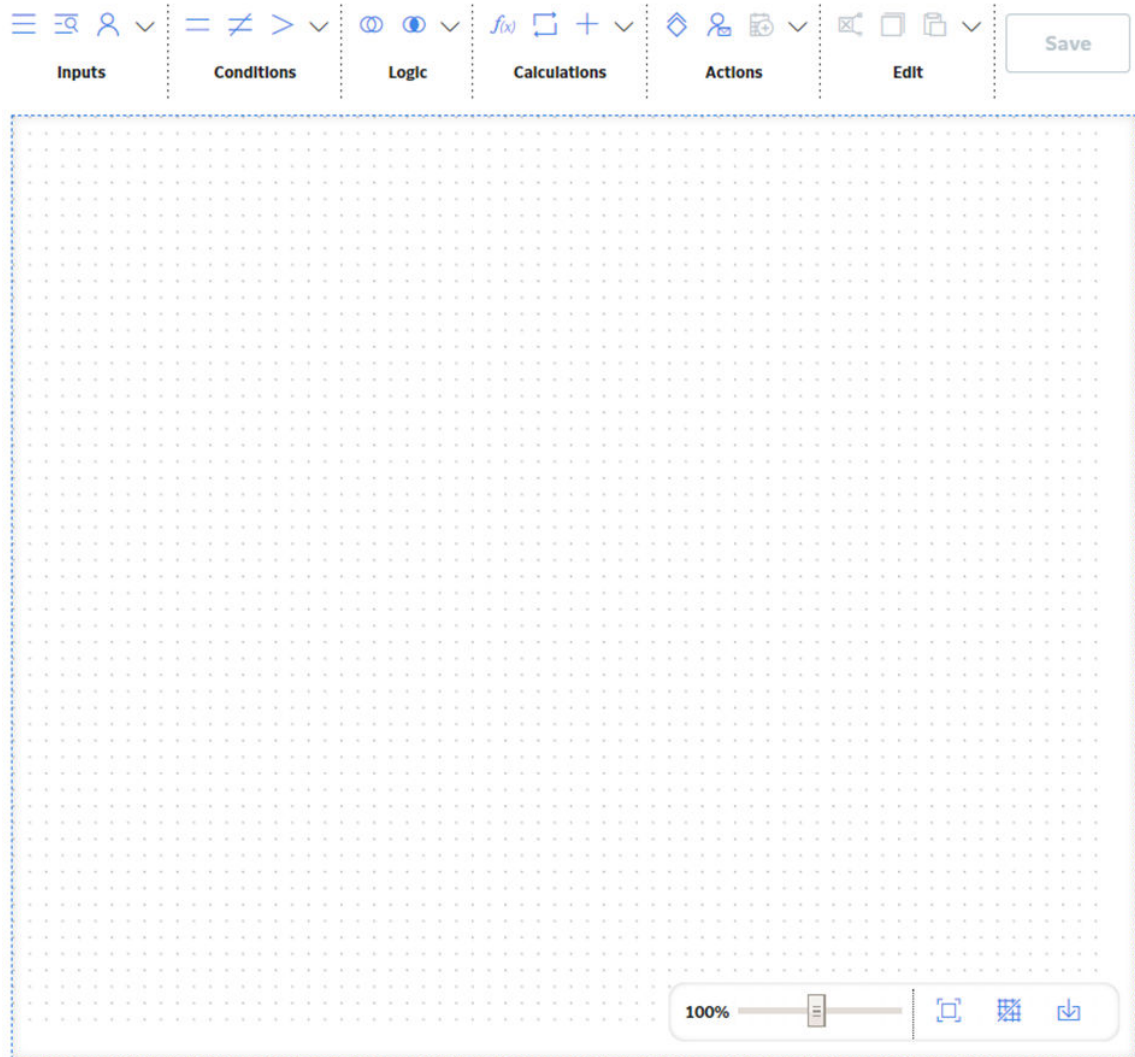
Connect Nodes in a Policy Model in Family Policies

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 that apply to connections:
  - You cannot add connections in a circular path (e.g., 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.
  - You cannot connect a node to same node more than once.

Procedure

1. Access the policy in which you want to connect 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.

Next Steps

- Configure Node Properties
- Configure Logic Paths for connections

Configure Node Properties in Family Policies

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**.

Next Steps

- Define Input Values
- Configure Logic Paths

Define Input Values in Family Policies

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

Complete one of the following sequences of 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. Select **Save**.
- 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.

     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. Select **Save**.

Example

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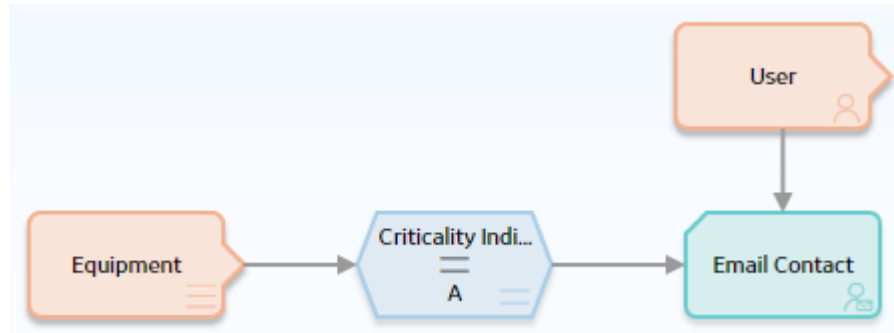
In this example, the Equipment Entity node is connected to an Equal Condition node. The **Properties** window for the Condition node, therefore, allows you to select the Equipment family and one of its fields. As shown in the following image:

- 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.

Next Steps

- Configure Logic Paths

Configure Logic Paths in Family Policies

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.

   ⌷ Properties　　　　　✕

   Logic Path

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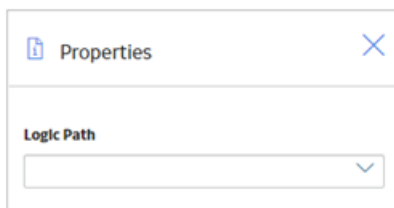
3. In the **Logic Path** box, specify whether the connection should be followed if the logical result of the preceding condition is true or false. Specifically:

   - 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.

   > **Example**
   >
   > 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 successor nodes:
    - Email Analyst (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. This 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.

Next Steps

- Define Input Values

Copy and Paste Nodes and Connections in Family Policies

Before You Begin

- Make sure you understand the basic principles of working with a policy model.
- **Note:**
    - 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 (e.g., 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.
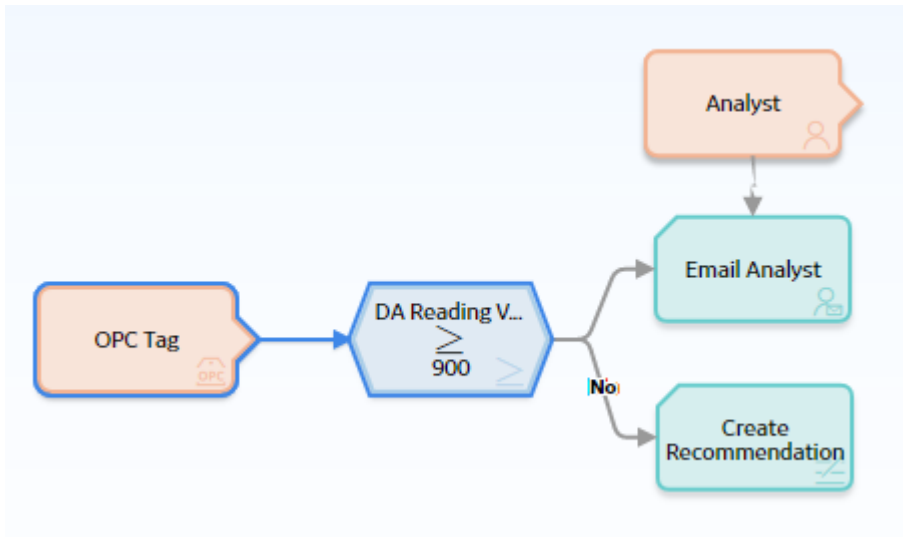
3. In the **Edit** section of the toolbar, select ⬚.

   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.
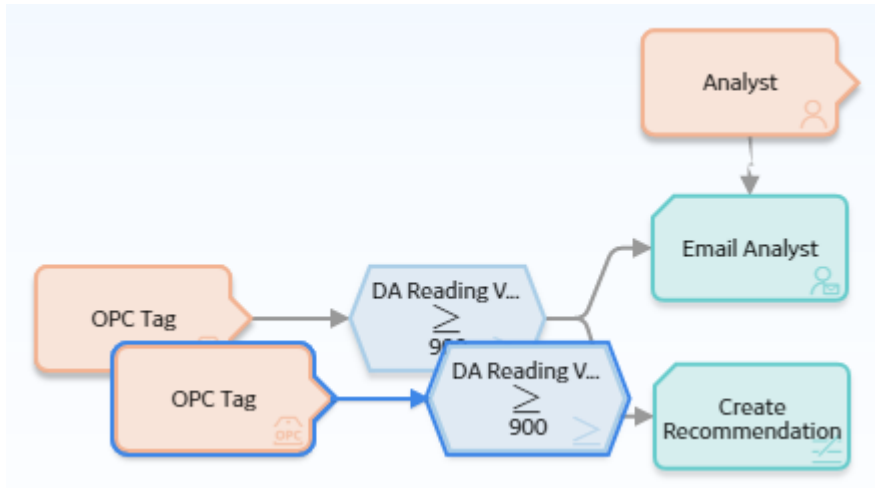
5. Drag the pasted nodes and connections to a new location as necessary.
6. Select **Save**.
   The policy is saved.

Next Steps

- Define Input Values

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

4

Policy Logic Validation

Topics:

About Validating Family 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.

Validate Policy Logic in Family Policies
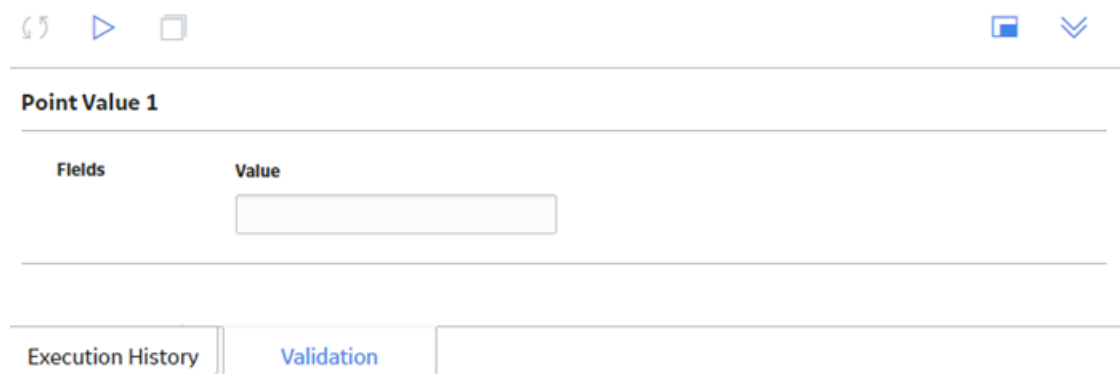
Before You Begin

- Add Nodes to the Model Canvas.

Procedure

1. Access the policy that you want to validate.
2. At the bottom of the **Design** workspace, select the **Validation** tab.

   The **Validation** pane appears. The pane displays one section for each Input node in the policy model that requires an input value.

3. Enter or select the test values that you want to use to validate the policy logic.
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 about the specific node's execution.

Chapter

5

Policy Execution

Topics:

- About Family Policy Execution
- Configure Family Policy Execution History Log Setting
- Access Execution History in Family Policies

About Family Policy Execution

When records in the APM database are created, modified, or deleted, any family policies configured for the corresponding family and trigger will be executed automatically. For example, if an After Insert family policy is configured for the Reading family, the policy will be executed after a Reading record is added to the database.

When a policy is executed, the logic in the policy model is evaluated and any resulting actions are taken. Each time a family policy is executed, the results of the execution are recorded in the execution log, which you can view on the **Execution History** pane in the Policy Design page.

A change to a record in the APM database will trigger the appropriate family policy regardless of the current user's permissions. However, if the user does not have permissions for an action a policy is taking, the policy will not execute, the transaction will be rolled back, and no changes will be made. Similarly, the transaction will be rolled back if an error occurs during policy execution.

Configure Family Policy Execution History Log Setting

About This Task

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

Important: The Family 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 Family 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.
	This setting does not affect the policy execution server log files.

Option	Description
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 in Family Policies

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, displaying a summary of past executions.

START TIME	END TIME	SUMMARY
01/06/2020 18:37:16	01/06/2020 18:37:16	Created event named test for asset Equipment ~ ~ Equipment Test Record.
01/06/2020 18:37:07	01/06/2020 18:37:07	No action taken.
01/06/2020 18:36:53	01/06/2020 18:36:53	No action taken.

Rows per page   50   100   200   500   1 - 5 of 5 Results   1

Execution History   Validation

3. Select an execution summary to view additional details on the policy canvas.

   On the policy canvas, the nodes in the model are color-coded to indicate the results of the execution.

   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**.

4. 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.

Chapter
6

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 family policies.

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

Procedure

In the module navigation menu, select **Admin** > **Application Settings** > **Policy Designer**. The **Policy Admin** page appears, displaying the **Execution History Settings** workspace.

Configure Execution History Retention Settings

About This Task

You can configure the execution history retention settings to specify the time for which execution history records for family policies must be retained in the APM database. You can configure the settings to retain the execution history records for executions that result in Errors, Warnings (Action Taken), Warning (No Action Taken), Success (Action Taken), or Success (No Action Taken), either for an indefinite period or for a specific duration. By default, the execution history records are configured to be retained for an indefinite period. If you configure the settings to retain these records for a specific duration, an automated job deletes the records that have been retained in the database for more than the specified duration. You can specify the interval at which the job must run.

Note:

- 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 execution history is retained in the APM system.
- The execution history retention settings are not specific to a family policy and are applicable for all family policies. The settings are also applicable to any execution history records for the Policy Designer.

- If a family policy has been executed only once, the execution history is retained in the database even if it exceeds the specified retention duration.
- Depending on the number of old execution history records that exists when the retention settings are configured, it may take multiple runs of the automated job to delete all the old execution history records.

This topic describes how to configure the settings to retain the execution history records for policies for a specific duration in the database depending on the execution result.

Procedure

1. Access the **Policy Admin** page.
2. For each execution result type which you do not want to retain execution history records indefinitely, 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 execution history logs in the database.

3. In the **Every** box, enter the time interval at which the automated job must run to delete old execution history records from the database.

   By default, the time interval that you enter is defined in hours. However, you can select the required unit from the drop-down list box to specify the interval in other units of time.

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

Chapter

7

Reference

Topics:

General Reference

About Specifying Dates and Times to Evaluate in Family Policies

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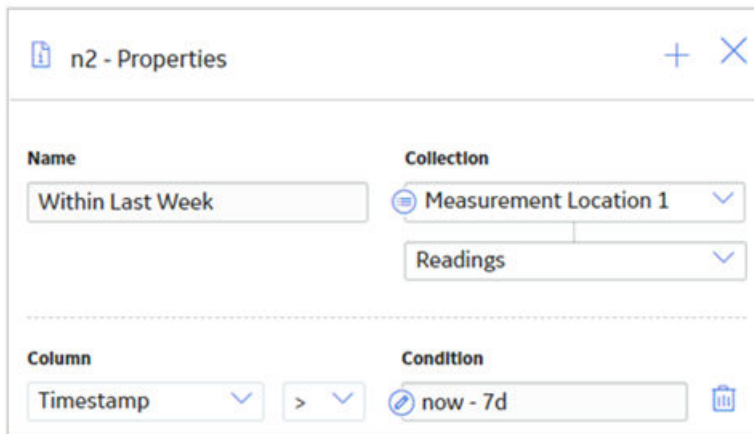
You can define specific date and time values in the standard format YYYY-MM-DD HH: mm: ss, where the time values use a 24-hour time format.

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.
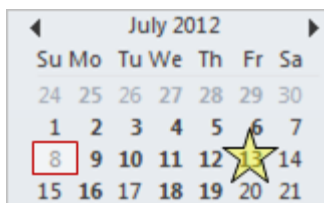
You can adjust relative date and time values by adding operators and variables to the relative date:

- 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	
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.

Constant	Meaning
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
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.
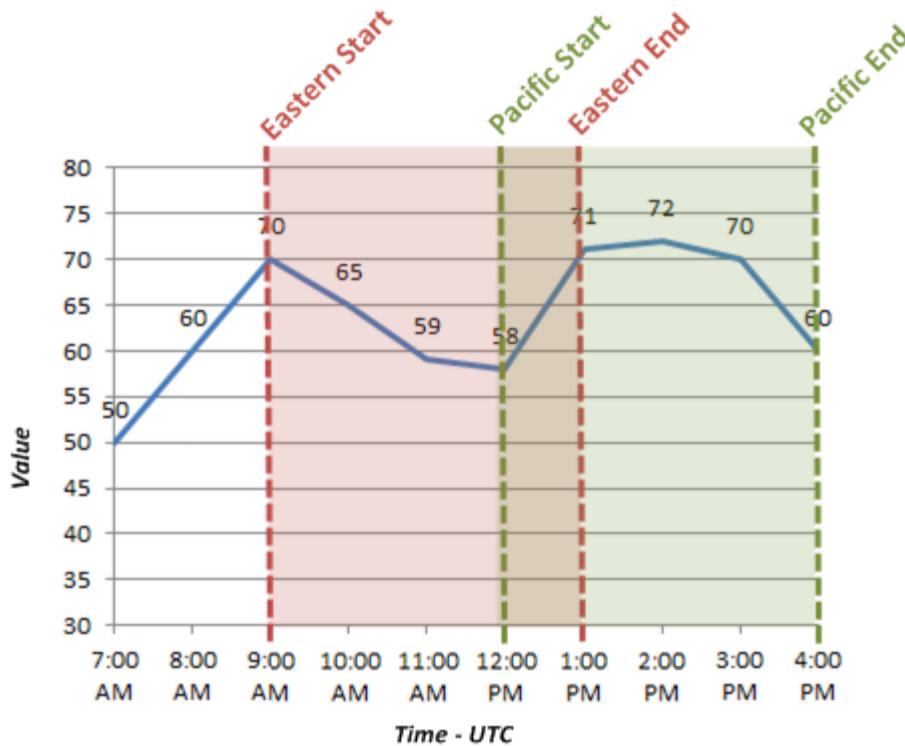
About Specifying Amounts of Time to Evaluate in Family Policies

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

Value	Description	Valid Number Type	Example
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 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:
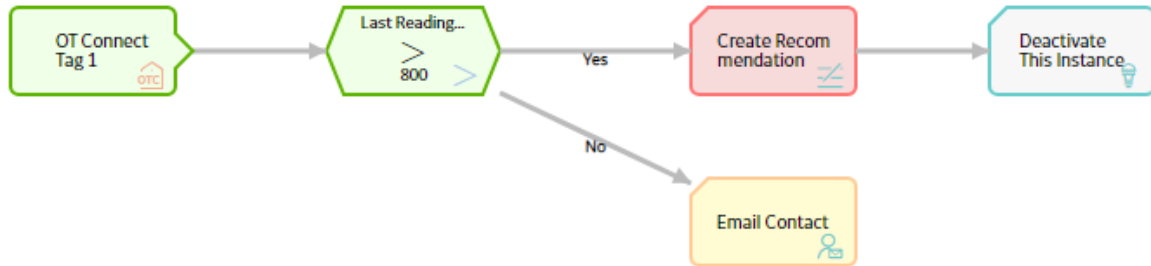
Node Execution Details Window

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.
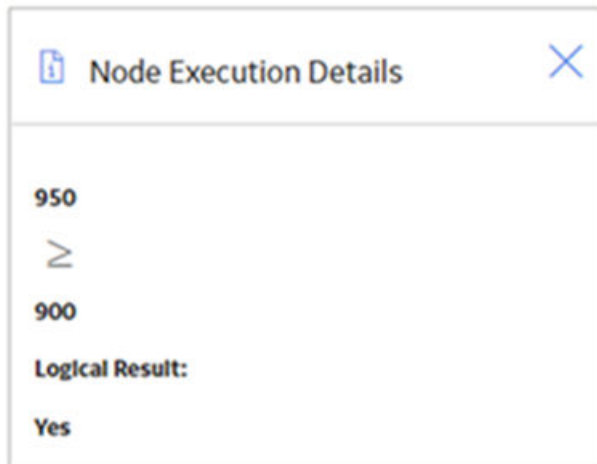
Note that the same node could have also been executed successfully with a logical result of No (e.g., 800 is not greater than or equal to 900).

About Execution Result Summaries in Family Policies

Each time a policy 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.

About Units of Measure in Family 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 Policy Records

Family Policy records store basic information about Family policies. The following table provides an alphabetical list and description of the fields that exist in the Family Policy family and that are displayed on the baseline Family 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 excluded from global search, and not to use Rules or Family Policies.

Note: The values in Family Policy records cannot be modified in Record Manager or accessed via the global search. Instead, you must access the policy in the family policies interface.

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 text box on the Details tab in the family policies interface.
Execution History Setting	Character	Defines how the execution history is recorded for the family policy.	Reflects the selection in the Execution History option in the **Details** workspace. The field is populated with the value Normal, by default.
Family	Character	The family ID of the family that is associated with the policy.	This field is populated automatically.
Model	Text	Code defining the logic that is represented by the policy model.	This field is populated automatically and does not appear on the Family Policy datasheet.
Trigger	Character	The trigger that is associated with the family policy.	This field is populated automatically.

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 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
Close Description	Text	A description of why the event was closed.	Populated automatically based on the properties defined for the associated Create 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 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.
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.

Field	Data Type	Description	Behavior and Usage
Policy Instance GUID	Binary	A unique identifier that is used internally by the APM system.	On the Policy Event datasheet, this field is labeled **Link to Policy** and contains a link to the policy instance that created the Policy Event record. **Note:** If the Policy Event is created by a Create Event node, the field is not automatically populated.
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.

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.

47

Field	Data Type	Description	Behavior and Usage
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.
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.

Field	Data Type	Description	Behavior and Usage
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. This field is required.
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. This field is required.
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.

Family 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 first node in a policy model must be an Input node rather than a Query node.
- 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.
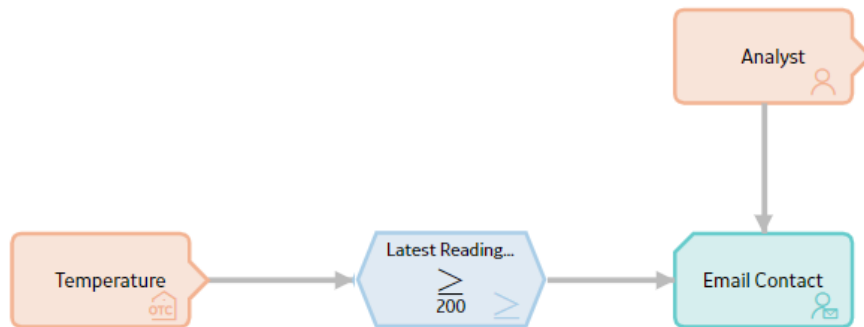
Configuring Node Properties

- 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 first node in a policy model must be an Input node rather than a Query node.	The Temperature node is an OPC Tag node, which is a type of Input node.
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.

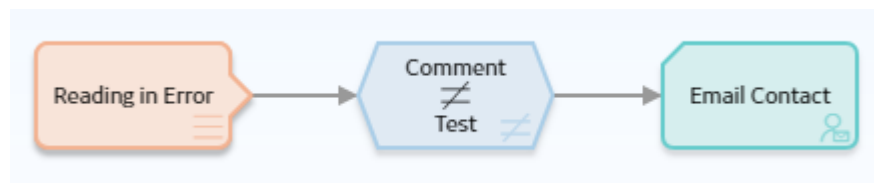| Policy Model Principle	Example
With the exception of the Or node and the Case node, a node will be executed only when all necessary preceding nodes have been successfully executed.	The Email Contact node will be executed only when all of the nodes preceding it have been successfully executed. If, for 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.

Family Policy Examples

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

Tip: Refer to the Policy Designer documentation for additional policy examples.

> ### Sending an Email When a Reading in Error Record is Created
>
> In the Rounds application, a Reading in Error is created when a new Reading record cannot be linked successfully to a Measurement Location during the process of uploading the reading from a mobile device. Suppose that you want to be alerted every time that a Reading in Error record is created in the database so that you can immediately address the issue. You could create family policy for the Reading in Error family that would be executed every time a new Reading in Error record is created.
>
> * Policy Model:
>
>
>
> * Policy Logic Summary:
>
>   This family policy is configured for the Reading in Error family using the After Insert trigger. If a Reading in Error record is added to the database with a comment other than test, an email message will be sent to the responsible user.
> * Individual Node Descriptions:

Node Name	Node Type	Description
Reading in Error	Current Entity	Represents the Reading in Error record that triggered the policy. Values associated with this record are evaluated by the policy.
Comment not equal to test	Comparison	Determines whether the value in the Comment field of the new Reading in Error record contains the value test. If it does not, the policy logic continues (i.e., an email message will be sent).
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 **Properties** window for the Email Contact node.

Monitoring Work History Records

Suppose that the Work History records you use in APM are imported from an external Enterprise Asset Management system on a regular schedule. Changes made to the Work History records in the external system may require action within APM. This example family policy, which is triggered before a Work History record is updated in the APM database, uses the Field Value Changing node to determine what actions should be taken based on whether the value of certain fields is being changed by the transaction.

- Policy Model:
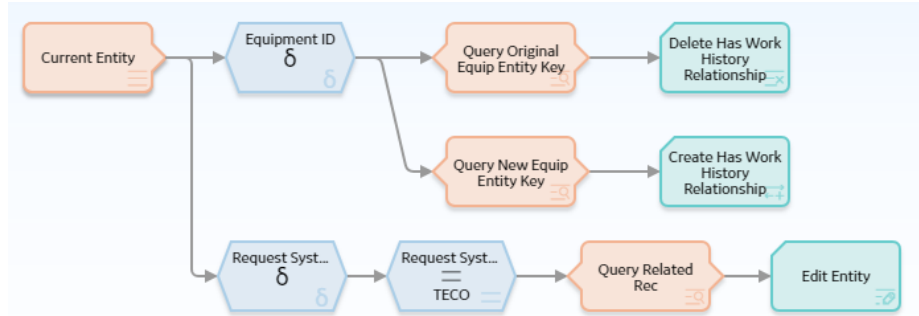
- Policy Logic Summary:

This family policy is configured for the Work History family using the Before Update trigger. This policy is designed to take two distinct actions. The first action is to check whether the Equipment ID specified in a Work History record is changing. If it is changing, the existing relationship between the Work History record and the Equipment record is deleted and a new relationship is created between the Work History record and the newly specified Equipment record.

The second action is to check whether the Request System Status code specified in the Work History record is changing. If it is changing, and the new value is TECO (i.e., technically completed), the status of any related recommendations is set to Implemented. This automates the process of reconciling completed work with the recommendation that originated the work request.

- Individual Node Descriptions:

Node Name	Node Type	Description
Current Entity	Current Entity	Represents the Work History record that triggers the policy. Values associated with this record are evaluated by the policy.
Equipment	Field Value Changing	Identifies whether the Equipment ID specified in the Work History record is changing.
Query Original Equip Enty Key	Query	Returns the record whose Equipment ID was previously specified in the Work History record.
Delete Has Work History Relationship	Delete Relationship	Deletes the existing relationship between the Work History record and the previously specified Equipment record.
Query New Equip Enty Key	Query	Returns the record whose Equipment ID is now specified in the Work History record.

Node Name	Node Type	Description
Create Has Work History Relationship	Create Relationship	Creates a new relationship between the Work History record and the newly specified Equipment record.
Request System Status	Field Value Changing	Identifies whether the value in the Request System Status field in the Work History record is changing.
Request System Status = TECO	Comparison	Determines whether the new value in the Request System Status field contains the value TECO. If it does, the policy logic continues.

Node Name	Node Type	Description
Query Related Rec	Query	Returns recommendations whose value in the Work Request Reference field matches the value in the Request ID field of the Work History record.
Edit Entity	Edit Entity	Updates the returned Recommendation records to change the recommendation status to Implemented.

Input Nodes

About Input Nodes in Family Policies

Input nodes represent various items and values that you can use as inputs to the policy logic.

Because Input nodes provide successor nodes with values to evaluate, the first node in a policy model must be an Input node. A Query node, however, cannot be the first node in a policy model.

Input Nodes

- Constant
- Current Entity and Current Relationship
- Current User
- Query Time Series
- Query

Constant Nodes in Family Policies

A Constant node is an Input node that represents a specific value that does not change from one policy execution to another. You can use a Constant node for input values that 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 selecting a value can help catch errors and facilitate policy execution.
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.

Current Entity and Current Relationship Nodes in Family Policies

Current Entity and Current Relationship nodes are Input nodes that represent the APM entity or relationship family, respectively, that is associated with the family policy. You can use these nodes to access information that is stored in the record whose changes triggered the family policy.

The Current Entity and Current Relationship nodes generate the following outputs:

- Any field in the record that triggered the family policy
- The following system fields in the family of the record that triggered the family policy: Entity Key, Content GUID, Entity ID, and Site Key

Note:

- The Current Entity node is available only for family policies associated with entity families and the Current Relationship node is available only for family policies associated with relationship families.
- There can be only one Current Entity or Current Relationship node in a single family policy.

Node Properties

Other than optionally specifying a name for the node, there are no properties to configure for a Current Entity or Current Relationship node.

> Current Entity or Current Relationship nodes are often the starting point in policy models because they provide successor nodes with fields to evaluate. For example, consider an AfterUpdate family policy belonging to the Work History entity family. Using a Current Entity node, you could configure a policy such that, when the policy is triggered by a change in a Work History record, successor nodes evaluate whether or not the value in the Work Order Status field of the corresponding record has changed. If it has, the policy sends an email message to notify users of the change. In addition, the policy updates the related Recommendation record to reflect the new status of the work order.

Current User Nodes in Family Policies

A Current User node is an Input node that represents information associated with the currently logged in APM user (i.e., the user who makes the change that triggers the family policy).
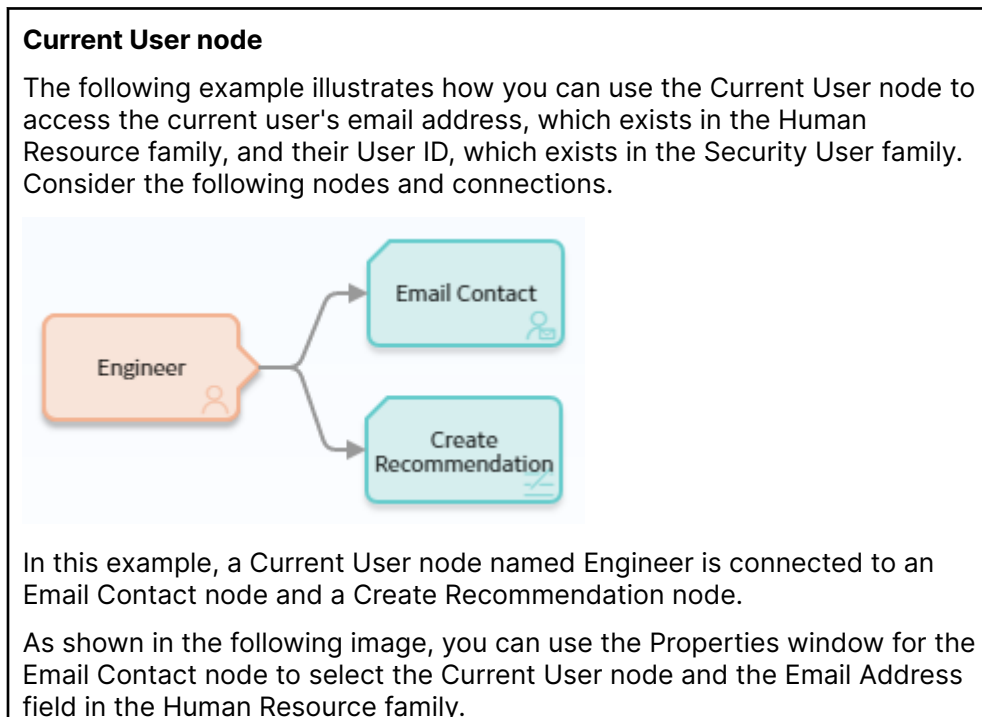
A Current User node generates the following output:

- Any field in the Security user or Human Resource record for the logged in user.
- The following system fields in the Security User record for the logged in user: Entity Key, Content GUID, and Entity ID.

Note: There can be only one Current User node in a single family policy.

Node Properties

Other than optionally specifying a name for the node, there are no properties to configure for a Current User node.

> **Current User node**
>
> The following example illustrates how you can use the Current User node to access the current user's email address, which exists in the Human Resource family, and their User ID, which exists in the Security User family. Consider the following nodes and connections.
>
>
>
> In this example, a Current User node named Engineer 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 Current 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 Current User node and, in this case, the User ID field in a field in the Security User family.

Query Time Series Nodes

A Query Time Series node is an Input node that queries Time Series data for an asset that uses the Predix Time Series service. You can use the Query Time Series node to retrieve readings from the default Time Series data source.

The Query Time Series node, requires you to provide the **Tag ID** as an input to the node. For example, the **Tag ID** to query could be retrieved from other records in the APM database using a Query node, or, determined by the policy logic and constructed from other node outputs using a Text node.

A Query Time Series node generates the following outputs:

- Time stamp, which represents the time stamp value for the most recent reading.
- Value, which represents the most recent reading value.
- Quality, which represents the quality of the most recent reading. The quality of the reading is indicated by one of the values described in the following table.

Numeric Value	Description
0	Bad Quality
1	Uncertain Quality
2	Not Applicable
3	Good Quality

Note: You must use the numeric value when you configure the nodes in the policy to operate on Quality values. However, in a policy output (for example, the text output from an Email Contact node), the description will be displayed. The example in this topic shows how to use the Quality value in a Collection Filter node.

- Collection, which represents all readings associated with the Time Series service within the specified time frame and maximum number of readings. Each reading in the collection includes time stamp, value, and quality details.
- Tag, which represents the ID of the Tag.

Node Properties

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

Item	Description	Notes
Time Series Tag	Specifies the ID of the Predix Time Series tag for which data will be retrieved	You can select 🖉 to specify the output of a predecessor node in this section.
Start Time	Specifies the beginning of the period within which the readings that you want to retrieve were recorded.	This is not a required field. However, we recommend that you specify an appropriate value in this field to retrieve the most relevant data.

Item	Description	Notes
End Time	Specifies the end of the period within which the readings that you want to retrieve were recorded.	This is not a required field. However, we recommend that you specify an appropriate value in this field to retrieve the most relevant data.
Max Readings	Specifies the maximum number of readings that you want to retrieve from the Time Series data source.	By default, 10000 readings are retrieved from the Time Series data source. However, as needed, you can specify any value up to the maximum number of points supported by the Time Series data source. **Note:** For best performance, we recommend that you specify the minimum number of readings required for the policy.

Item	Description	Notes
Query Operation	Specifies the type of data you want to retrieve from the Time Series data source.	By default, raw is selected. However, as needed, you can select **interpolated**, **average**, or **sum** to retrieve the required data. **Type of Data** / **Description** Raw — Retrieves data as it was ingested. Supported for all Time Series data sources. Interpolated — Retrieves one interpolated value for each interval within the specified time range. Supported for all Time Series data sources. Average — Retrieves one average value for each interval within the specified time range. Supported for Predix data sources only. Sum — Retrieves either, one value which is the sum of the raw data values in the specified time-range, or, one value which is the sum of the raw data values for each interval within the specified time range. Supported for Predix data sources only.

Item	Description	Notes
Interval (seconds)	Specifies the time interval in seconds.	This is required when query mode is Interpolated or Average; it is an optional field for Sum. This is the time interval in seconds that the query for interpolated or average data is based on.
Interpolated Mode	Specifies the interpolation mode.	Interpolated mode field is required when query mode is Interpolated.

Depending on the values specified in the **Properties** window, the collection of returned readings varies as described in the following table.

Start Time	End Time	Max Readings	Result
Date 1	No value	No value	Returns up to 10,000 earliest points starting on Date 1.
Date 1	No value	N	Returns up to N earliest points starting on Date 1.
No value	Date 2	No value	Returns up to 10,000 latest points ending on Date 2.
No value	Date 2	N	Returns up to N latest points ending on Date 2.
Date 1	Date 2	N	Returns up to N points starting on Date 1 and ending on Date 2.
No value	No value	No value	Returns up to 10,000 latest points ending on the current date.
No value	No value	N	Returns up to N latest points ending on the current date.

An example to help understand the use of Query Time Series node is provided in the Policy Designer documentation.

Query Nodes in Family Policies

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 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:

- A Query node cannot be the first node in a policy model.
- The Result Set is restricted to the 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 enter the path manually, or you can browse to the query by selecting 👓. The query that you choose must have an ID and a caption.
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.

Tip: Refer to the Policy Designer documentation to see an example of this node.

Condition, Logic, and Calculation Nodes

About Condition, Logic, and Calculation Nodes in Family Policies

Condition Nodes

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

- Comparison nodes
- Case nodes

Logic Nodes

- And
- Or

Calculation Nodes

The following Calculation nodes perform calculations on single values:

- Add
- Convert Type

- Divide
- Exponent
- Field Value Changing
- Is Null
- 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

The Baseline Rule node performs calculations based on the corresponding rule.

Add, Subtract, Multiply, Divide, Exponent and Remainder Nodes in Family Policies

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.

Tip: 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.
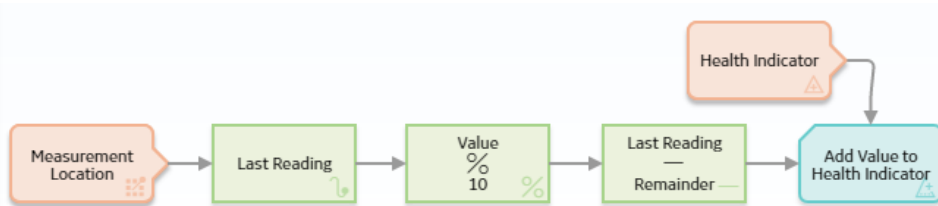
The following image shows the **Properties** window of a Remainder node. You can use the Remainder node to calculate the amount remaining after the Last Reading value is divided by 10. For example, if the last reading value is 87, the result of the remainder node is 7 (87/10=8 with a remainder of 7).

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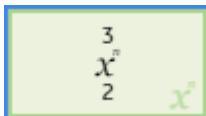
Node Type	Mathematical Calculation	Output Value
Add	3+2	5
Subtract	3-2	1
Multiply	3×2	6
Divide	3/2	1.5
Exponent	3^2	9
Remainder	3%2	1

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 1: 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 2: 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 3: 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 4: 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 5: 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 Family Policies

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.
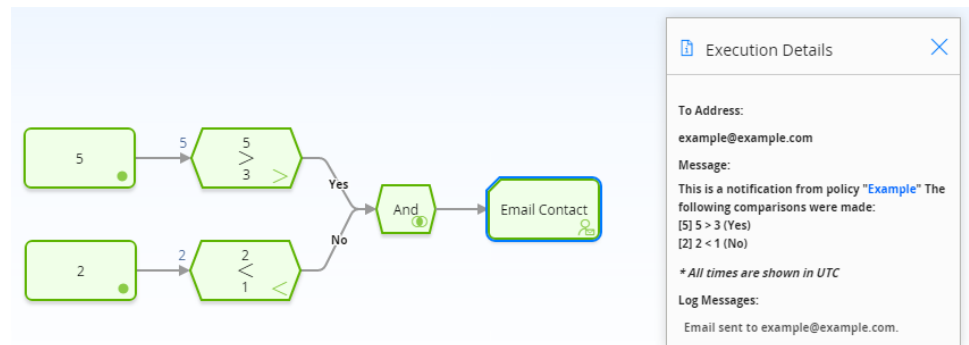
In this example, you can see that each node executed successfully. The logical result of the And node is yes because all incoming logic paths result in a value of true passed to the And node (that is, the logical result of each preceding condition node matches the logic path of the corresponding connection). 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.
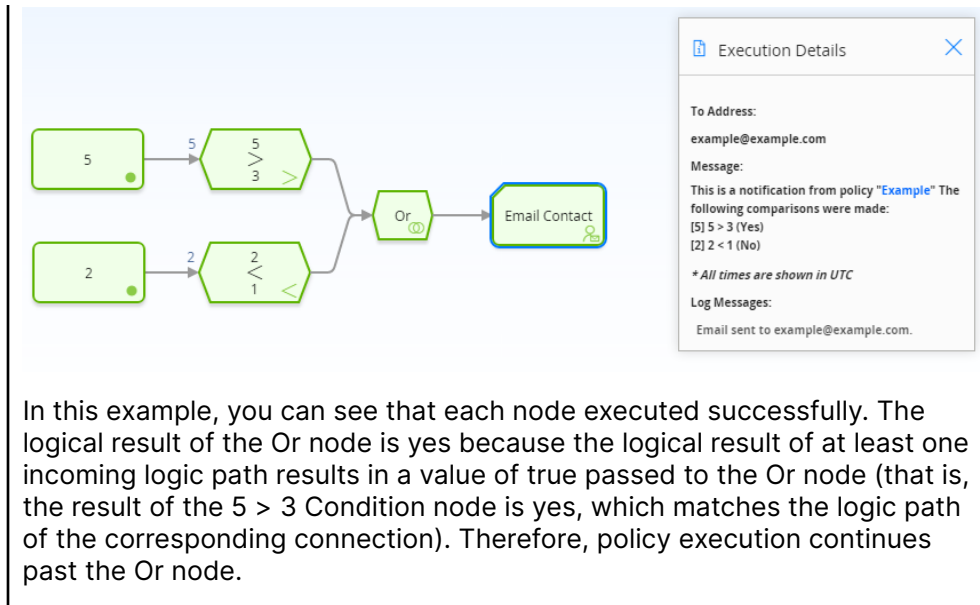
Average Nodes in Family Policies

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.

Tip: Refer to the Policy Designer documentation to see an example of this node.

Baseline Rule Nodes in Family Policies

A Baseline Rule node represents an action to execute baseline family-level rules in APM. You can use the Baseline Rule node in a family policy to run any existing APM baseline rules that correspond to the policy's family and trigger. In this way, you can use both family policies and APM baseline family-level rules.

Note: The Baseline Rule node executes only APM baseline rules, not any family-level rules that you have written. For a single family, you can write family-level rules or family policies, not both.

Node Properties

Other than optionally specifying a name for the node, there are no properties to configure for a Baseline Rule node.

Guidelines for using the Baseline Rule node

When working with the Baseline Rule node, consider the following behaviors and guidelines:

- You must ensure that the policy logic is designed so that only one Baseline Rule node will execute. A notification of this requirement will appear if more than one Baseline Rule node is added to the policy.
- When you validate a family policy that includes a Baseline Rule node, the validation will actually execute the corresponding rule. It is therefore recommended that the policy is fully tested in a development or quality assurance environment prior to implementing the policy in a production system, where testing the policy could have undesirable consequences.
- Where a family policy executes a Baseline Rule node, you may observe a noticeable delay when updating records that trigger the policy.

Baseline Rule Node

The APM baseline rules for the Reading family includes a Before Insert rule that sets the Status field on a Reading record according to when the reading was taken. If the Reading Taken Date is before the Next Date on the related Checkpoint Task, the Status is set to 'I' (i.e. Ignored), and the Next Date on the Checkpoint Task is not updated, leaving the previously set schedule for the checkpoint intact. This might indicate either that there is a maintenance issue with the related asset which needs additional attention, or that the reading was taken ahead of schedule. In the first scenario, you might want to ensure that any required maintenance action has been planned or completed. In the second scenario, you might want to reset the checkpoint schedule to avoid repeating an inspection unnecessarily.

The following example illustrates how you can configure a set of family policies for the Reading family to notify users that an Ignored reading has been inserted.

First, in a Before Insert policy, you can use the Baseline Rule node to run the existing APM baseline rule that sets the Status of the new reading. The following image shows this basic policy. No settings are required on either of the nodes in this policy.

Then, you can configure a second family policy on the After Insert trigger to send an email to the Rounds administrator if an Ignored reading was inserted. The following image shows this policy.
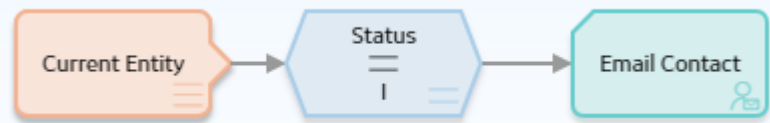
As you can see in the following image of the **Properties** window for the Equal node, the value in the Status field is checked to determine whether it is equal to 'I', which is the stored system code (not the display value) for the field.

And, finally, as you can see in the following image of the **Properties** window for the Email Contact node, you can configure the email address to which to send the email notification and a message.

Case Nodes in Family Policies

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.

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.

Item	Description	Notes
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.

Tip: Refer to the Policy Designer documentation to see an example of this node.

Collection Filter Nodes in Family Policies

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.

Item	Description	Notes
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

Tip: Refer to the Policy Designer documentation to see an example of this node.

Comparison Nodes in Family Policies

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 (i.e., 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.

Tip: Click here to see example of these nodes used within complete policy models.

Convert Type Nodes in Family Policies

You can use a Convert Type node to convert a value to another data type. The node converts the value to the corresponding value of the specified data type, and provides the converted value as its output. 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

Data Type of Input Value	Data Type Specified for Output Value					
	Boolean	**Decimal**	**Integer**	**String**	**Time & Date**	**Time Span**
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
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.
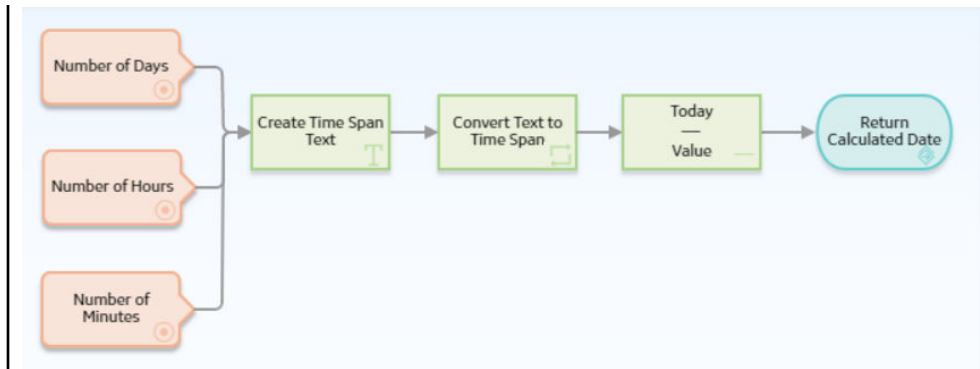
In this example, the Text node is configured to create a text string that indicates a time span, using the values represented by the Point Value nodes. Because the Subtract node cannot perform any calculation on a text string, the Convert Type node converts the string to a value of the Time Span data 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.
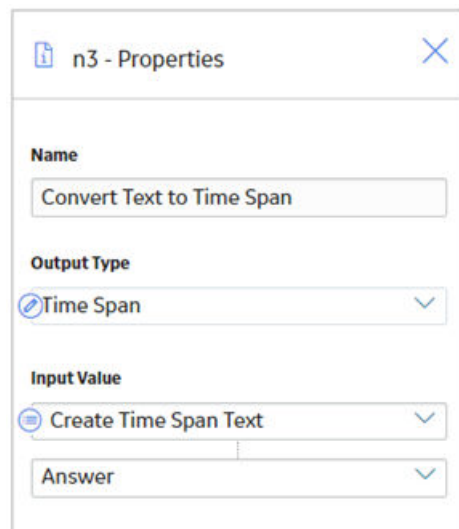
Count Nodes in Family Policies

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.

Tip: Refer to the Policy Designer documentation to see an example of this node.

Field Value Changing Nodes in Family Policies

A Field Value Changing node is a Condition node that you can use in a Before Update family policy to determine whether a value in a certain field is changing in the transaction that triggered the family policy.

The input of a Field Value Changing node must be a field from a Current Entity or Current Relationship node. The Field Value Changing node generates the following outputs.

- The logical result of the node (i.e., yes if the value in the specified field is changing in the corresponding transaction or no if it is not changing).

  Note: Logical results are used by the 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 Field Value Changing node to configure a logic path for the connection.
- Original Value, which is the value in the specified field before the change.

Node Properties

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

Item	Description	Notes
Field section	Specifies the field whose value will be evaluated to determine whether it is changing in the transaction that triggered the family policy.	You must specify a field from a Current Entity or Current Relationship node.

> **Field Value Changing node**
>
> The following example illustrates how you can use a Field Value Changing node to send an email message if the Criticality Indicator value in an Equipment record changes from A to some other value. Consider the following nodes and connections.
>
>
>
> In this example, the Field Value Changing node first determines whether or not the value in the Criticality Indicator field is changing. The following image shows the **Properties** window for the Field Value Changing node.

If the value in the Criticality Indicator field is changing (i.e., the logical result of the Field Value Changing node is yes), the Equal node then evaluates whether the initial value in the field was A, as shown in the following image.

Finally, if the original Criticality Indicator value was A, an email message is sent to the specified recipient(s).

Is Null Nodes in Family Policies

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 Valuesection	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.

Tip: Refer to the Policy Designer documentation to see an example of this node.

Last Nodes in Family Policies

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.

Tip: Refer to the Policy Designer documentation to see an example of this node.

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 numeric 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**.

Item	Description	Note
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.
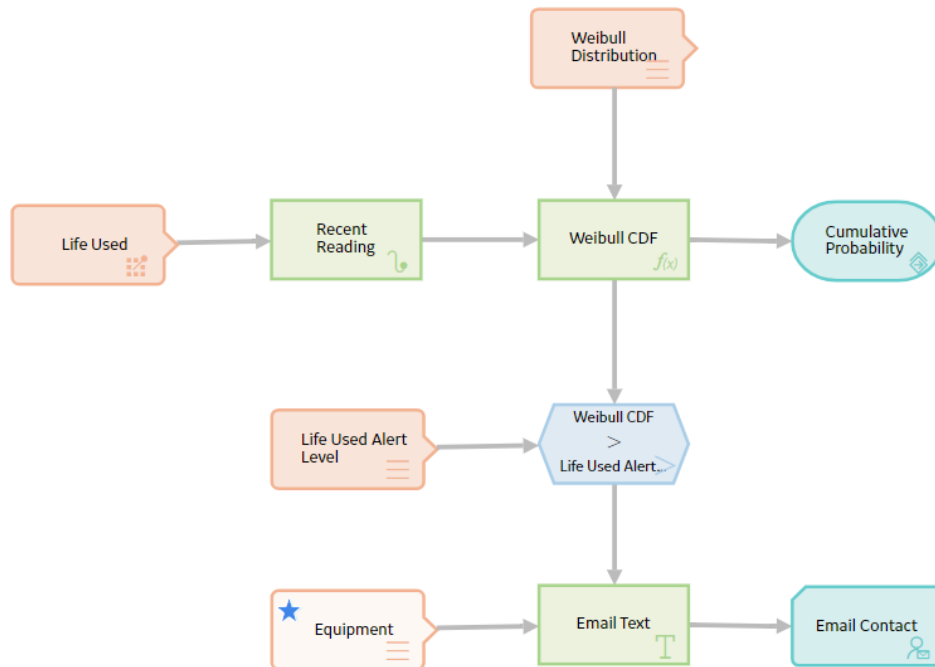
The threshold percentage of life used for the equipment is stored as a Technical Characteristic. The life data of the equipment (for example, hours for turbine, starts for crane, take-offs for aircraft, tonnes moved for mining equipment), is stored as Measurement Location readings.

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.
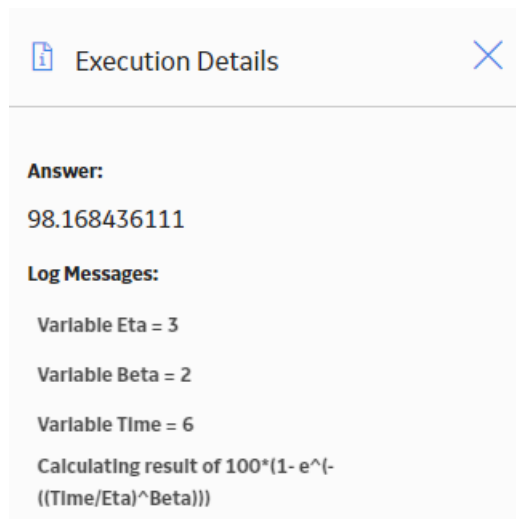
The Life Used Measurement Location input node provides the readings to the Recent Reading Last calculation node, which retrieves the latest measurement reading (that is, Time).

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 Family Policies

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.

Tip: Refer to the Policy Designer documentation to see examples of these nodes.

Round Nodes in Family Policies

A Round node is a Calculation node that you can use in a policy model to:

- Round a value to a specific number of decimal places. This is called decimal rounding.
  -or-
- 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** text 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** text 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 (i.e., 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 text 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 text box is empty (this is the same as typing 0 [zero]). This means that decimal values will be rounded to the nearest whole number.

Tip: Refer to the Policy Designer documentation to see an example of this node.

R Script Nodes in Family Policies

An R Script node is a Calculation node that you can use in a policy model to return a result that is 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 output of an R Script node contains the value 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 query 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 (i.e., a Vector of Values, a Matrix of Values, Data Frame, or a single value that is Numeric, Character, Boolean, or Time & Date).

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 Vector, Matrix and Data Frame inputs, the collections that you specify cannot contain any null values.
- 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.
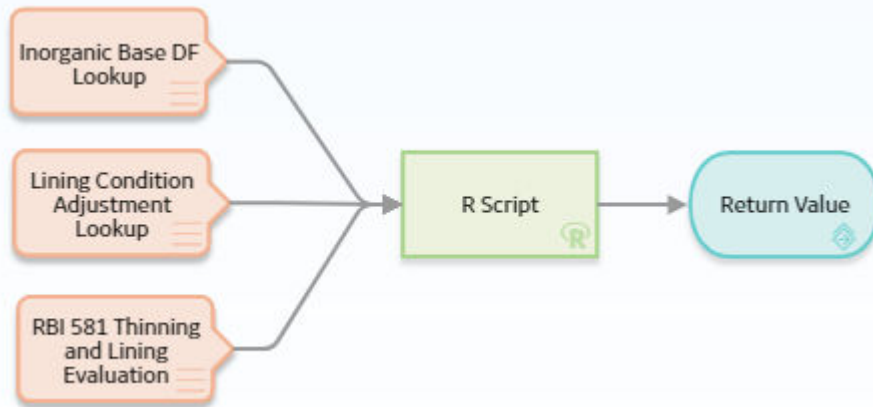
As shown in the following image, you can use the **Properties** window for the R Script node to select a particular R script that was created using the R script editor. The R script in this example requires three input values: LinerBaseDF, LiningConditionAdjustment, and OnlineMonitoringAdjustment. These 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.
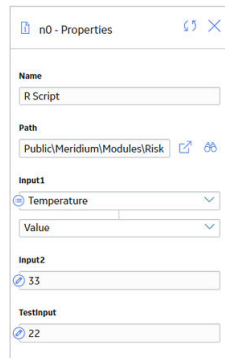
Calculating Average and Standard Deviation of a Reading Collection

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)
```

(0)	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.
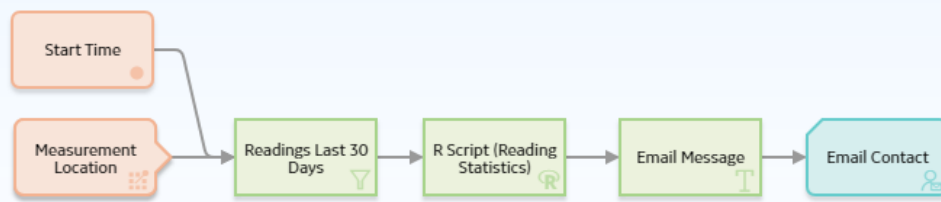
This policy first filters Measurement Location readings to only those readings recorded within the last 30 days. Then, the filtered readings are sent as a Data Frame input to the R script.

The following image shows the **Properties** window for the R Script node.

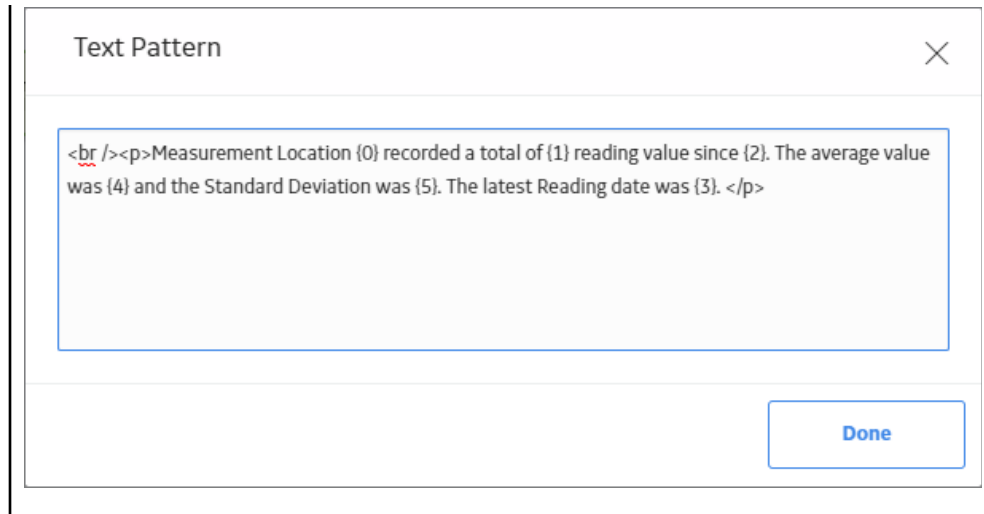Next, the calculated results returned by the R script are used in a Text node to construct a message that will ultimately be sent via email to a responsible user.

The following image shows the **Properties** window for the Text node.
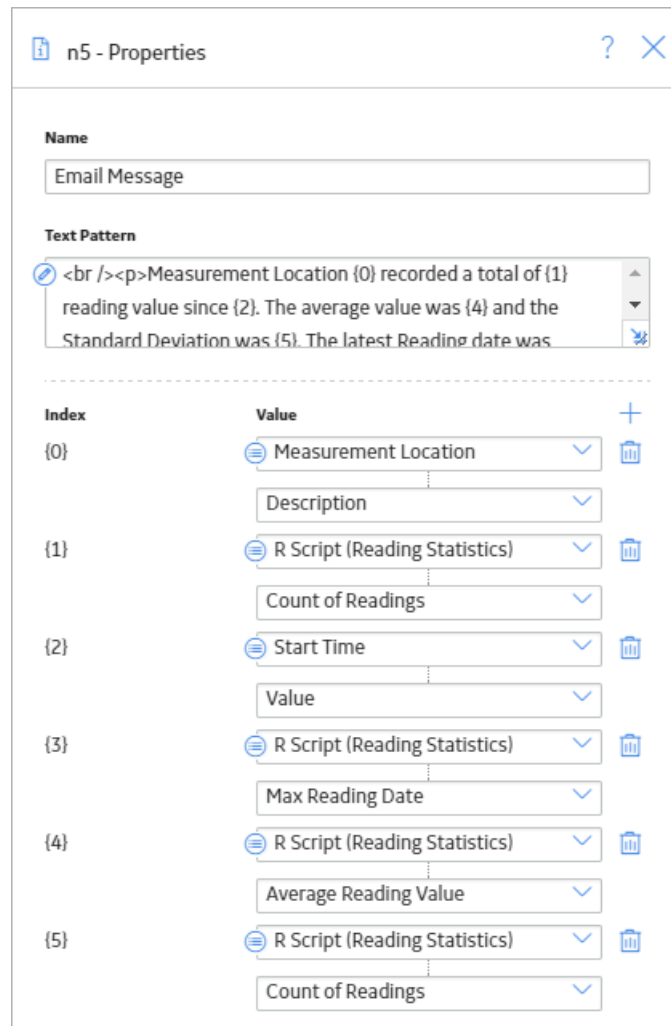
The following image shows the full text pattern specified in the Text node.

Sum Nodes in Family Policies

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.

Tip: Refer to the Policy Designer documentation to see an example of this node.

Text Nodes in Family Policies

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.
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).

Tip: Refer to the Policy Designer documentation to see an example of this node.

Threshold Statistics Nodes in Family Policies

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 (i.e., 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 (>)

Tip: Refer to the Policy Designer documentation to see an example of this node.

Action Nodes

About Action Nodes in Family 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
- Apply Strategy Template
- Cancel Transaction
- Create Entity
- Create Event
- Create Production Event
- Create Recommendation
- Create Relationship
- Delete Entity
- Delete Relationship
- Edit Entity

- [Email Contact](#)
- [Return Value](#)
- [Rule](#)
- [Sub Policy](#)

Add Value to Health Indicator Nodes in Family Policies

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.

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 an Add Value to Health Indicator node are the Content GUID and Entity Key of the record that the node creates.

Important: If you use this node in a Before Insert or After Insert policy for the Health Indicator Value family or in a Before Update or After Update policy for the Health Indicator family, make sure that the policy logic contains conditions to prevent a circular reference (that is, a scenario in which the action of the policy triggers the policy to execute continuously). The notification bar will display a warning if such a scenario is detected.

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.
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: Refer to the Policy Designer documentation to see an example of this node.

Apply Strategy Template Nodes in Family Policies

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 use the Apply Strategy Template node only if you have the Asset Strategy Management license.
- 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.

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.
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.

Apply Strategy Template node

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:

Cancel Transaction Nodes in Family Policies

The Cancel Transaction node represents an action to revert the transaction that triggered the policy. There are no outputs of the Cancel Transaction node.

Note: The nodes after the Cancel Transaction node in a policy model will not be executed. Therefore, you must add the Cancel Transaction node as the last node in a policy model.

Node Properties

The **Properties** window for a Cancel Transaction node contains the fields and sections described in the following table:

Field/Section	Description	Note
Reason	Specifies the reason for which the transaction that triggered the policy must be reverted.	This value is required. If the transaction that triggered the family policy was initiated by user action (for example, modifying an entity in Record Manager), the reason specified in this field will be displayed to the user in a window when the transaction is reverted.

Cancel Transaction node

The following example illustrates how you can use a Cancel Transaction node in a Before Update policy to prevent the creation of an invalid notification in the configured SAP system:
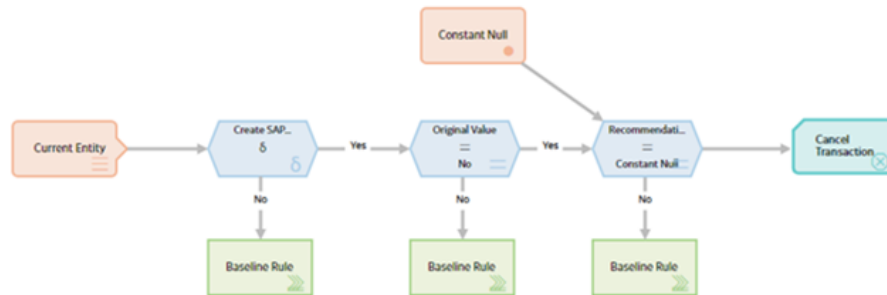
In this example, the Current Entity node represents a Recommendation record that has been modified to create an SAP Notification. A valid SAP Notification can be created from a Recommendation record only if the **Create SAP Notification** check box is selected and if a value is specified in the **Recommendation Basis** box. In the policy model, the Field Value Changing node verifies whether the **Create SAP Notification** check box is selected in the Recommendation record. If the **Create SAP Notification** check box is selected in the record, the subsequent Equal node in the policy model verifies whether the **Create SAP Notification** check box was previously not selected. If the check box was previously not selected, the other Equal node in the policy model verifies whether the value specified for the **Recommendation Basis** box of the Recommendation record is null. If the value specified in the box is null, the Cancel Transaction node reverts the modifications made to the Recommendation record so that an invalid SAP notification is not created in the configured SAP EAM system. Additionally, a window appears for the user who made the modifications to the Recommendation record, indicating the reason for reverting the changes.

The **Properties** window for the Cancel Transaction node described in this example is shown in the following image:
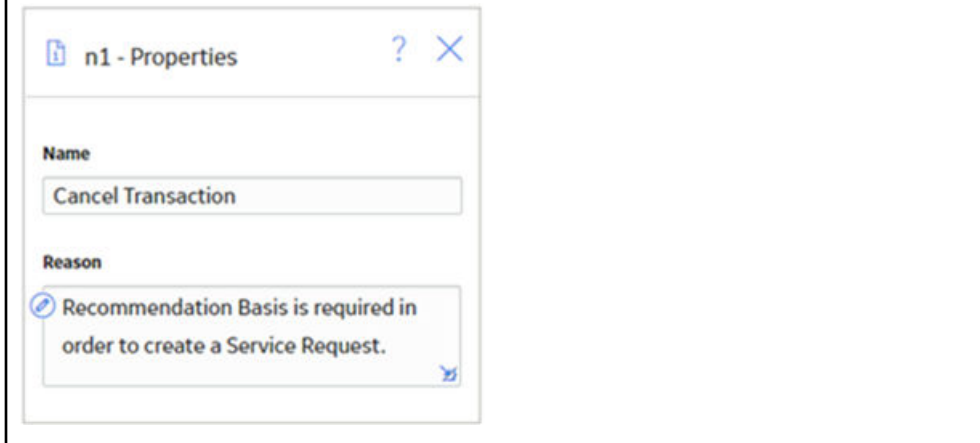
Create Entity Nodes in Family Policies

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.

The outputs of the Create Entity node are the following system fields for a record that the node creates: Content GUID, Entity Key, Entity ID, and Site Key.

Important: If you use the Create Entity node in a Before Insert or After Insert policy to create a record in the same family that triggers the policy, make sure that the policy logic contains conditions to prevent a circular reference (that is, a scenario in which the action of the policy triggers the policy to execute continuously). The notification bar will display a warning if such a scenario is detected.

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.

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.

Item	Description	Notes
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.

Item	Description	Notes
+	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.
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: Refer to the Policy Designer documentation to see an example of this node.

Create Event Nodes in Family Policies

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: Content GUID, Entity Key, Entity ID, and Site Key.

Important:

- If you use this node in a Before Insert or After Insert policy for the Policy Event family, make sure that the policy logic contains conditions to prevent a circular reference (that is, a scenario in which the action of the policy triggers the policy to execute continuously). The notification bar will display a warning if such a scenario is detected.
- The Close Event node provided in Policy Designer that automatically closes Policy Events with duration is not available in family policies. However, in a family policy, you can create an open Policy Event with duration by configuring the following settings in the **Properties** window of the Create Event node:

- In the **Has Duration** box, specify Yes.
- In the **End Time** box, do not specify any value.

If you want a Policy Event to automatically close, you can create a policy or a family policy that triggers an action that closes the Policy Event. You must configure the policy such that it finds the relevant Policy Event record (for example, by using a query based on the asset to which the open Policy Event is linked), and uses an Edit Entity node to modify the Policy Event record.

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** : The event is not associated with a specific type of event. • **Excursion** : 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.

Item	Description	Notes
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.
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. 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.	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 of Boolean type (that is, Yes or No).
Time Line Reset section	This field is not currently used.	None

Tip: Refer to the Policy Designer documentation to see an example of this node.

Create Production Event Nodes in Family Policies

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 use the Create Production Event node only if you have the Production Loss Accounting license.

The outputs of the Create Production Event node are the following system fields for the record that the node creates: Content GUID, Entity Key, Entity ID, and Site Key.

Important: If you use this node in a Before Insert or After Insert policy for the Production Event family, make sure that the policy logic contains conditions to prevent a circular reference (that is, a scenario in which the action of the policy triggers the policy to execute continuously). The notification bar will display a warning if such a scenario is detected.

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: Refer to the Policy Designer documentation to see an example of this node.

Create Recommendation Nodes in Family Policies

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: Content GUID, Entity Key, Entity ID, and Site Key.

Important: If you use this node in a Before Insert or After Insert policy for the Policy Recommendation family, make sure that the policy logic contains conditions to prevent a circular reference (that is, a scenario in which the action of the policy triggers the policy to execute continuously). The notification bar will display a warning if such a scenario is detected.

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 (i.e., 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: Refer to the Policy Designer documentation to see an example of this node.

Create Relationship Nodes in Family Policies

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 outputs of the Create Relationship node are the following system fields for the record that the node creates: Content GUID, Entity Key, Entity ID, and Site Key.

Important:

- In a Before Insert policy, you cannot use the Create Relationship node to create a relationship between the record that triggers the policy and the records that will be created due to the execution of the policy.
- In a Before Insert or After Insert policy, if you use this node to create a record in the same family that triggers the policy, make sure that the policy logic contains conditions to prevent a circular reference (that is, a scenario in which the action of the policy triggers the policy to execute continuously). The notification bar will display a warning if such a scenario is detected.

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.

In the Notes column, continuing:

• **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 of 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. However, in a Before Insert policy, you cannot create a relationship between the records that will be created by the Action node and the record that is configured to trigger the policy.

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.

Tip: Click here to see an example of this node used within a complete policy model.

Delete Entity Nodes in Family Policies

A Delete Entity node represents an action to delete from the APM database one or more records in any baseline or custom entity family.

Important: In a Before Update, Before Insert, Before Delete, or After Delete policy, it is not valid to use this node to delete the record that triggered the policy (i.e., the record represented by the Current Entity node).

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.

Delete Relationship Nodes in Family Policies

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.

Node Properties

The **Properties** window for a Delete 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 Delete Relationship node will delete.	The **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.

Tip: Click here to see an example of this node used within a complete policy model.

Edit Entity Nodes in Family Policies

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.

Important:

- If you use the Create Entity node in a Before Update or After Update policy to modify a record of the same family that triggers the policy, make sure that the policy logic contains conditions to prevent a circular reference (that is, a scenario in which the action of the policy triggers the policy to execute continuously). The notification bar will display a warning if such a scenario is detected.
- In a Before Delete or After Delete policy, it is not valid to use this node to modify the record that triggered the policy (that is, the record represented by the Current Entity node).

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.

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 keys that you specify must belong to the family selected in the **Family ID** list. **Note:** If no entity keys are passed to the node during execution, a warning message appears.
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
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.
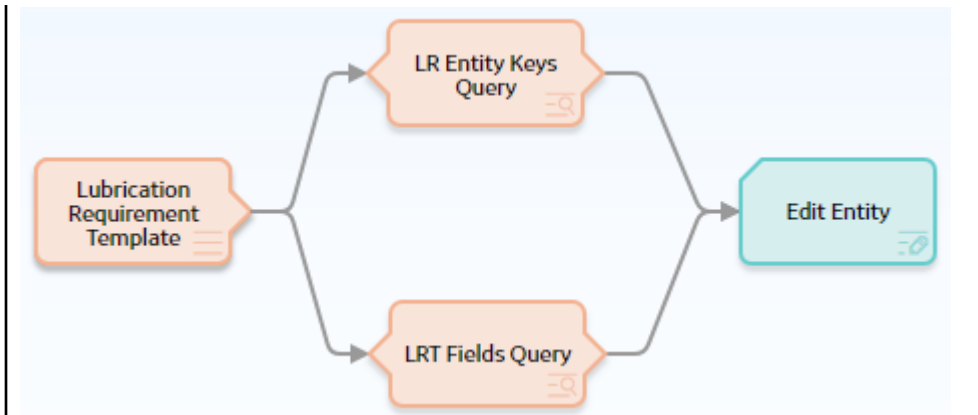
On executing this policy, all Lubrication Requirements related to the Lubrication Requirement Template that is specified in the policy instance are updated with the new values from the Lubrication Requirement Template record.

Email Contact Nodes in Family Policies

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.

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 section	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.
Message section	Content that you want to include in the email message in addition to the summary of the policy execution (which is always included automatically).	You can select to specify the output of a predecessor node in this section.

Tip: Click here to see an example of this node used within a complete policy model.

Return Value Nodes in Family Policies

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 in order to verify that certain values are being evaluated 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.

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.

Tip: Refer to the Policy Designer documentation to see an example of this node.

Rule Nodes in Family Policies

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 GE Vernova 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.

Sub Policy Nodes in Family Policies

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 Family Policy 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.

126

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, collection 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, each output of the sub policy is a collection of values.

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 or relationship 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.
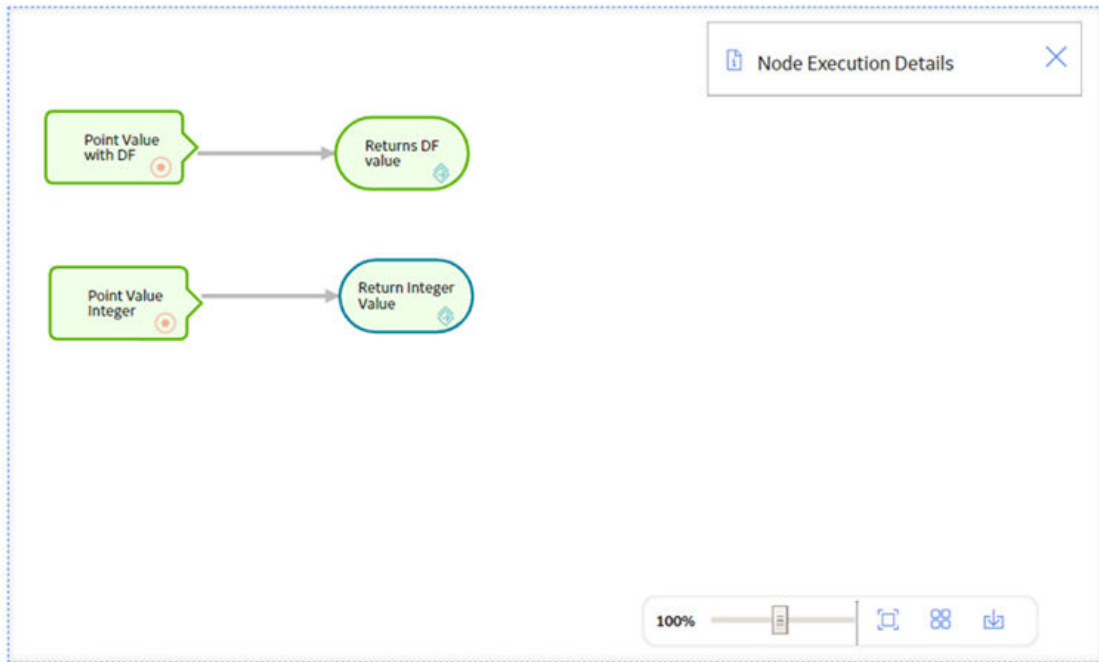
Sub Policy Node Execution Details

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:
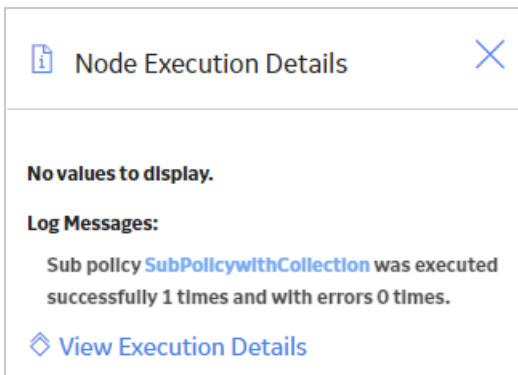
The following image is an example of the **Sub policy execution details for policy** window that appears when you select the **View Execution Details** link in the **Node Execution Details** window for a Sub Policy node:

Sub policy execution details for policy "SubPolicywithCollection" ✕

Node Execution Details ✕

Point Value with DF → Returns DF value

Point Value Integer → Return Integer Value

100%   ⌐ 88 ⤓

Done

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:
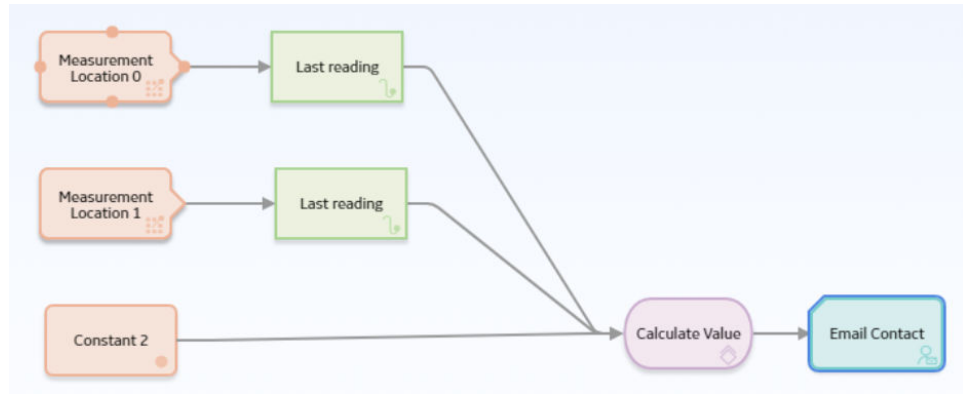
In this example, the latest reading values from two measurement locations and a constant value are passed into a sub policy, which calculates a value to be used in the Email node. The **Properties** window for the Sub Policy node is shown in the following image:

n4 - Properties ✕

Name

Calculate Value

Policy

Sub Policy Demo

Iterate Over Collection?

⚪ Yes 🔵 No

Execute Specific Instance?

⚪ Yes 🔵 No

Point Value 0

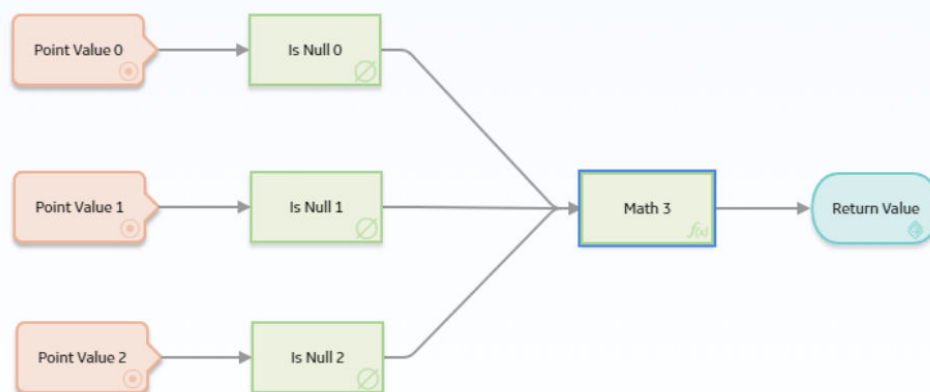Last Reading ⌄

Value ⌄

Point Value 1
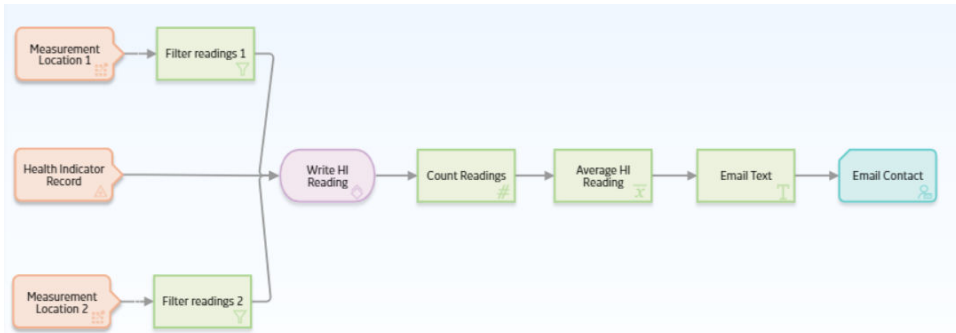
Last Reading ⌄

Value ⌄

Point Value 2

Constant 2 ⌄

Value ⌄

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.

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 .

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.