

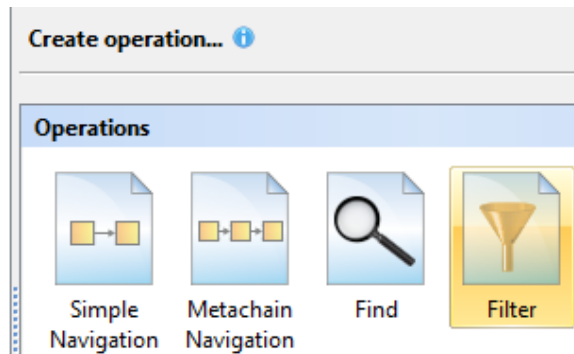
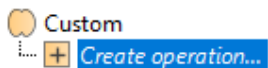


Case 10. Filtering Diagrams by Modification Date

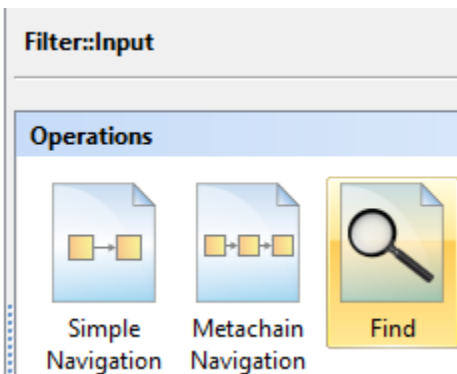
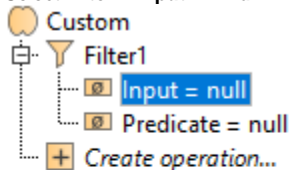
To filter diagrams by modification date

1. Create a **Generic Table**.
2. Set the **Element Type** to **Diagram**. Make sure to select the **Include Subtypes** check box. Click **OK**.
3. Set the **Scope** where the diagrams will be searched for.
4. Click the  icon in the **Scope** field and select **Custom**. The **Query** dialog opens.
5. Select **Create operation > Filter**.

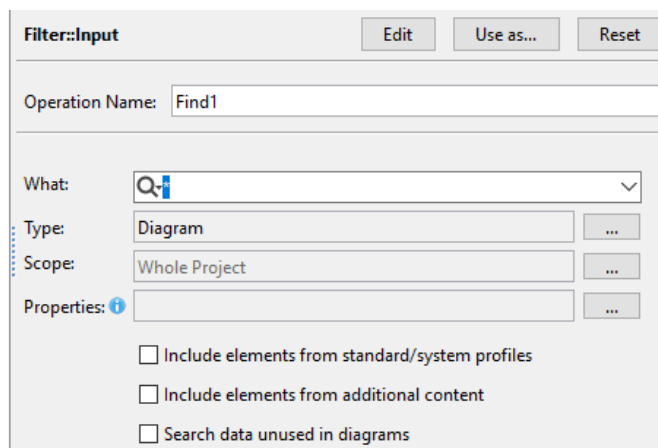
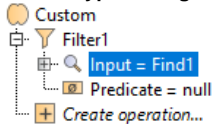
 If you cannot see the **Filter** operation, make sure the **Expert** mode is enabled.



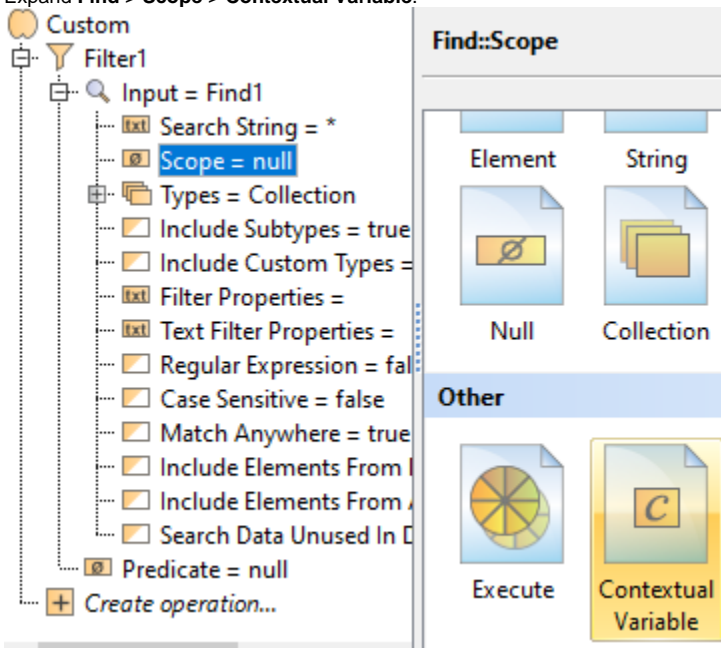
6. Select **Filter > Input > Find**.




7. Set the **Type** to **Diagram**.

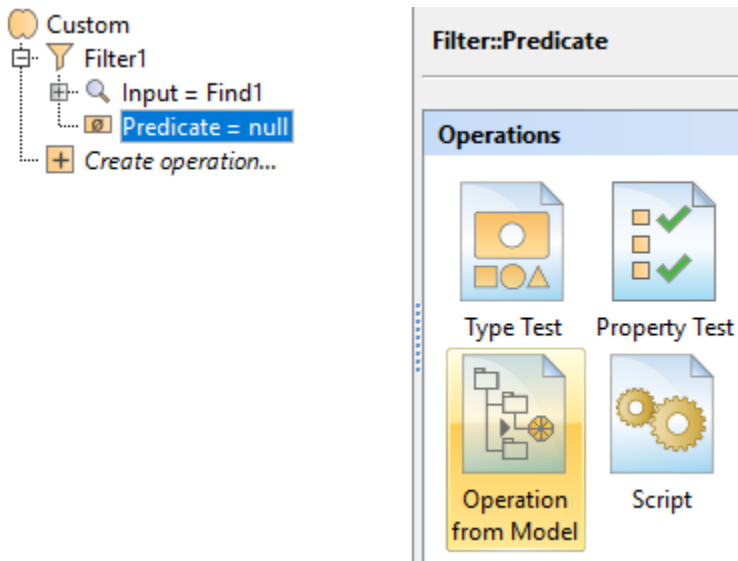


8. Expand **Find > Scope > Contextual Variable**.



9. Select **Filter > Predicate > Operation from Model > GreaterThanOrEquals**.

 If you cannot see the **GreaterThanOrEquals** operation, make sure the **Apply Filter** box is not checked.



10. Select **Predicate > Body > A > Script**.

The screenshot shows the configuration of a custom filter operation. The tree structure is as follows:

- Custom
 - Filter1
 - Input = Find1
 - Predicate = Nested Operation
 - Body = GreaterThanOrEquals1
 - A = null
 - B = null
 - arg

The 'Operation from Model::A' panel displays various operations: Simple Navigation, Metachain Navigation, Find, Filter, Type Test, Property Test, Union, Exclude, Operation from Model, and Script (highlighted).

11. From the Language drop-down list, select **Groovy**.
12. Insert the following script as the **Body**:

Groovy

```
import java.text.DateFormat;  
import java.util.Date;  
  
if (arg1) {  
    Date date1 = DateFormat.getDateInstance(DateFormat.SHORT).parse(arg1.iterator().next());  
}
```

The screenshot shows the configuration of the custom filter operation with the script body inserted. The tree structure is as follows:

- Custom
 - Filter1
 - Input = Find1
 - Predicate = Nested Operation
 - Body = GreaterThanOrEquals1
 - A = Script1
 - B = null
 - arg

The 'Operation from Model::A' panel shows the configuration for 'Script1':

- Operation Name: Script1
- Language: Groovy
- Body: Script1(arg1)

```
1 import java.text.DateFormat;  
2 import java.util.Date;  
3  
4 if (arg1) {  
5     Date date1 = DateFormat.getDateInstance(DateFormat.SHORT).parse(arg1.iterator().next());  
6 }
```

13. Select **A > arg1 > Reset**.

The screenshot shows the configuration of the custom filter operation with the 'Reset' button highlighted. The tree structure is as follows:

- Custom
 - Filter1
 - Input = Find1
 - Predicate = Nested Operation
 - Body = GreaterThanOrEquals1
 - A = Script1
 - arg1 = arg
 - B = null
 - arg

The 'Script::arg1' panel shows the configuration for 'Script1':

- Operation Name:
- Parameter Name: arg1
- Value: arg

14. Select A > arg1 > Simple Navigation.

Custom

Filter1

Input = Find1

Predicate = Nested Operation

Body = GreaterThanOrEquals1

A = Script1

arg1 = null

Create parameter...

B = null

arg

Create parameter...

Create operation...

Script::arg1

Parameter Name: arg1

Operations

Simple Navigation

Metachain Navigation

Type Test

Property Test

15. Select **Modification date** and set **Is Applied** to true.

Custom

Filter1

Input = Find1

Predicate = Nested Operation

Body = GreaterThanOrEquals1

A = Script1

arg1 = Simple Navigation

Create parameter...

B = null

arg

Create parameter...

Create operation...

Script::arg1

Edit

Use as...

Reset

Remove

Parameter Name: arg1

Relation Criterion	Is Applied	Direc...	Pro...	Res...
metricType («ValidationBasedMet...	<input type="checkbox"/> false			
modelIdentifier («fmu»)	<input type="checkbox"/> false			
modelName («fmu»)	<input type="checkbox"/> false			
Modification date («DiagramInfo»)	<input checked="" type="checkbox"/> true	Sourc...		
modifiedAt («AttachedFile»)	<input type="checkbox"/> false			
n («virtual»)	<input type="checkbox"/> false			
name («SimulinkPort»)	<input type="checkbox"/> false			
name («ModelicaPort»)	<input type="checkbox"/> false			
name («ModelicaBlock»)	<input type="checkbox"/> false			
name («SimulinkBlock»)	<input type="checkbox"/> false			

16. Select **Predicate > Body > B > Script**.

The screenshot shows a model editor on the left with a tree structure. The tree is expanded to show a nested operation. The 'Body' of the operation is 'GreaterThanOrEquals1', which has a child 'A = Script1'. 'A = Script1' has a child 'arg1 = Simple Navigation', which has a child 'Create parameter...'. 'Create parameter...' has a child 'B = null', which has a child 'arg', which has a child 'Create parameter...'. The 'Create parameter...' node is highlighted. On the right, the 'Operation from Model::B' dialog is open. It shows a list of operations: Simple Navigation, Metachain Navigation, Find, Filter, Type Test, Property Test, Union, Exclude, Operation from Model, and Script. The 'Script' operation is highlighted.

17. From the Language drop-down list, select **Groovy**.

18. Insert the following script as the **Body**:

Groovy

```
Date M1 = new Date();
M1.setMonth(M1.getMonth() - 1);
M1
```

The screenshot shows the same model editor as before, but now the 'B = Script1' node is highlighted. On the right, the 'Operation from Model::B' dialog is open. The 'Operation Name' is 'Script1'. The 'Language' is set to 'Groovy'. The 'Body' is 'Script1(arg1)'. The script is inserted into the 'Body' field:

```
1 Date M1 = new Date();
2 M1.setMonth(M1.getMonth() - 1);
3 M1
```

19. Click **OK**. To view the column, make sure to display it via the **Columns** button in the diagram toolbar > **Select Columns** > set the **Modification date** column to **true**.

Sample model

The model used in these examples is the *Case Studies for Querying the Model* sample model. To open this model, you need to download [case studies for querying the model.mdzip](#).