Infor PLM Accelerate 11

Query Builder Guide

Publication Information

Publication date: May 29, 2018

Contents

About this guide

Intended audience

This guide is intended for system administrators. This document assumes that you have at least some knowledge in:

- System architecture and function for your Infor system
- SQL Server database

For the most up-to-date list of software and hardware requirements for Infor products, see the documentation for your system.

Related documents

You can find the documents in the product documentation section of the Infor Xtreme Support portal, as described in the "Contacting Infor" section below.

Contacting Infor

If you have questions about Infor products, go to the Infor Xtreme Support portal at www.infor.com/inforxtreme.

If we update this document after the product release, we will post the new version on this Web site. We recommend that you check this Web site periodically for updated documentation.

If you have comments about Infor documentation, contact documentation@infor.com.

Chapter 1. Overview

The Query Builder application enables administrators to build reusable AML queries using a graphical user interface (GUI). This application includes a mechanism to specify desired Properties, xProperties, and Search Parameters.

Extended properties (or xProperties) enable users to assign properties to items based on the item classification. Users can assign xProperties to multiple Item Types. For more information about xProperties and xClasses, refer to the Extended Classification document.

The Query Builder also enables administrators to include a Relationship structure that can go either up or down, depending on business requirements.

The Query Builder includes an **Execute Query** function that sends the AML request. This function enables end users to test out their query and refine their search as needed.

The intended use of the Query Builder application is to create queries that can be used in other applications. Infor provides another Out Of The Box (OOTB) application called Tree Grid View which makes use of the Query Builder capability. The Tree Grid View creates a Relationship Tab for an ItemType where end users can examine contextual data structures for a given item. For more information on the Tree Grid View, refer to the *Tree Grid View Administrator Guide*.

This guide walks through building the following sample query:

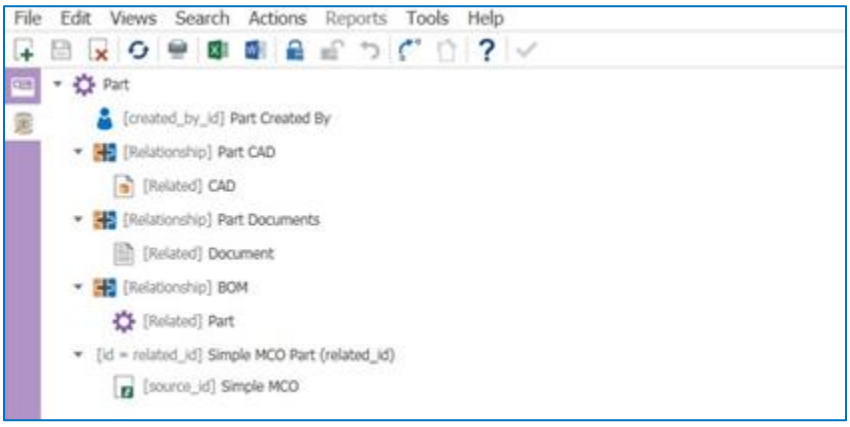

Figure 1.

Chapter 2. Developing Query Definitions

Reusable queries are defined by the Query Definition item, found under **Administration\Configuration\Query Definitions** in the TOC.

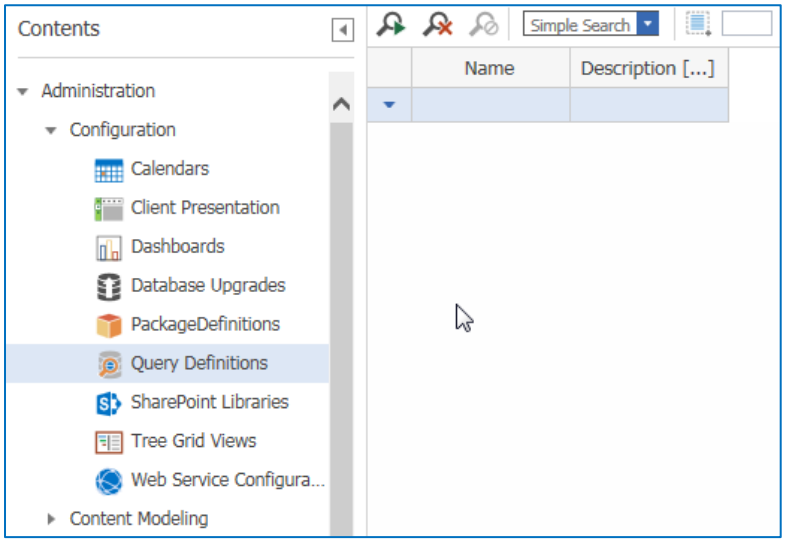

Figure 2.

Each Query Definition starts with an initial **Context Item** before building it out to a more extensive structure. The basic steps for creating a Query Definition are as follows:

1. Define the Context Item.
2. Build the Relationship Structure.
3. Apply Search Conditions.

Defining a Context Item

The following example creates a Query Definition item for Part ItemType.

1. Create a new query definition and specify a unique **Name** and a **Context Item Type**.

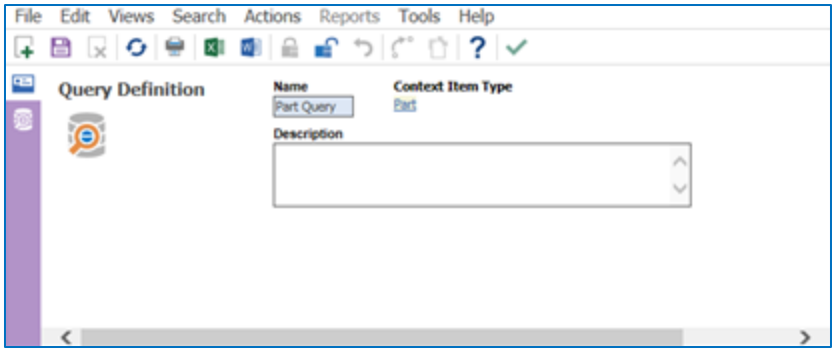

Figure 3.

2. Click the **Show Editor** button on the left sidebar to go into Query-Editing Mode.
3. Right-click the Context Item and select Edit Query Element --> Selected Properties.

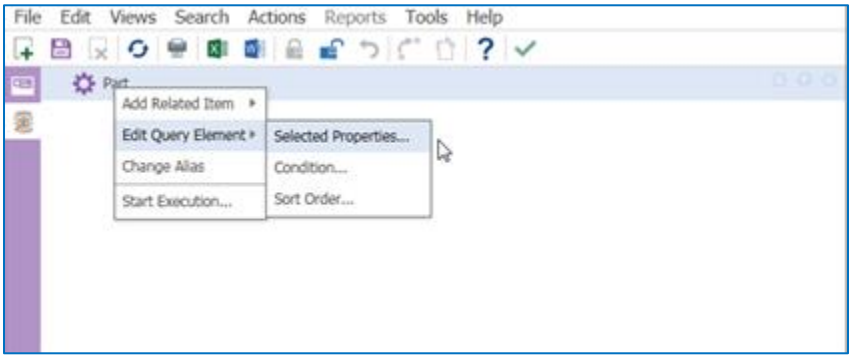

Figure 4.

Select the desired properties that the query should return. In this example they are: `id`, `Part Number`, `Name`, `State`, and `managed_by_id`.

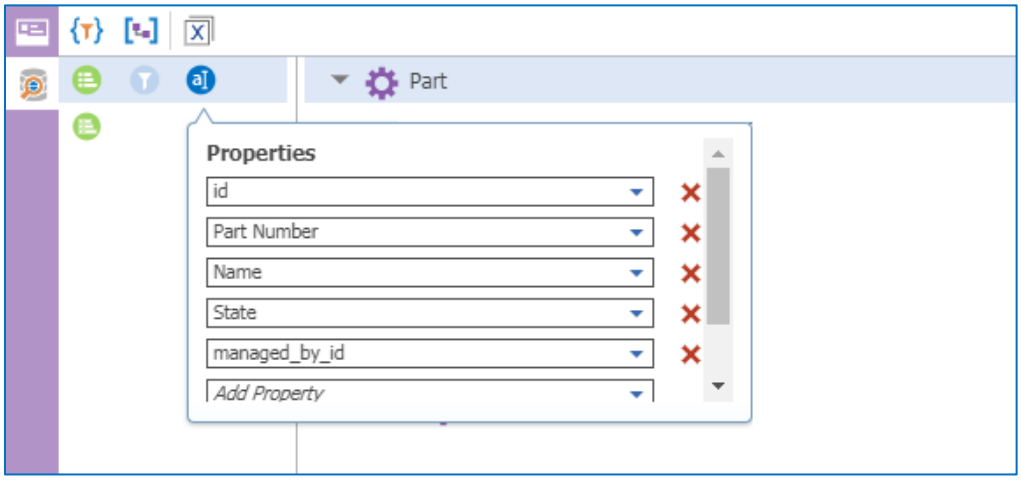

Figure 5.

Extended properties enable you to add values such as cost to an object in the database.

4. Right-click on the Context Item again and select **Edit Query Element --> Sort Order**.
5. Choose the properties that should be used for the Order By command and specify whether they should be in ascending or descending order.

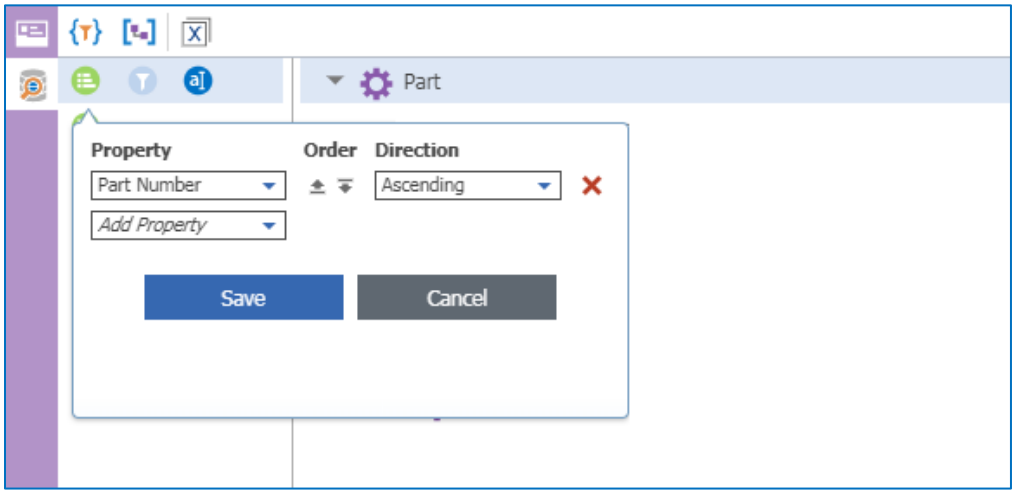

Figure 6.

6. Save the Query.

Building a Relationship Structure

It is possible to define a Relationship from the context item that can go either up or down the data structure. 'Down' refers to a relationship direction starting from the Item representing the *source* side of the relationship and following the related content to the *related* Item. 'Up' by contrast refers to a relationship direction starting from the Item representing the *related* side of the relationship and following the related content to the *source* Item. It is also possible to set up a recursive structure, such as the Part --> Part BOM --> Part, to reuse the same logic for multiple levels. The following procedures show how to expand the query both up and down for a Part Query and introduces recursion.

Building a Downward Relationship Structure

1. Right-click on the Part Context Item and select **Add Related Item --> Using Relationship**.

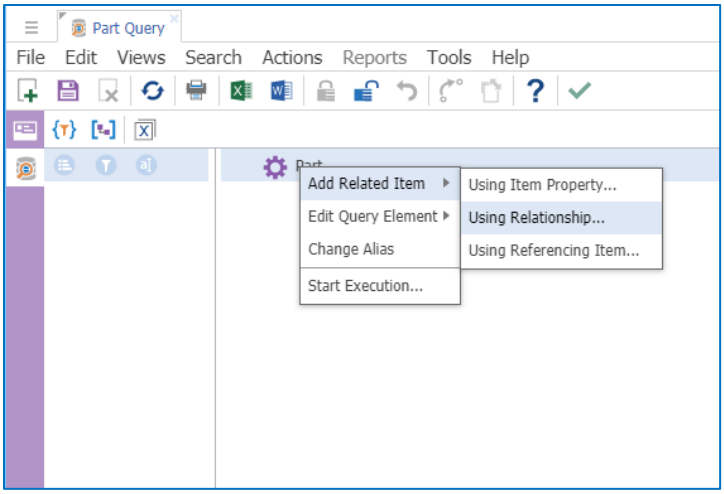

Figure 7.

2. Select the Relationship to include in the structure. In this case, select **BOM,** check **Include Related** and click **Add**.

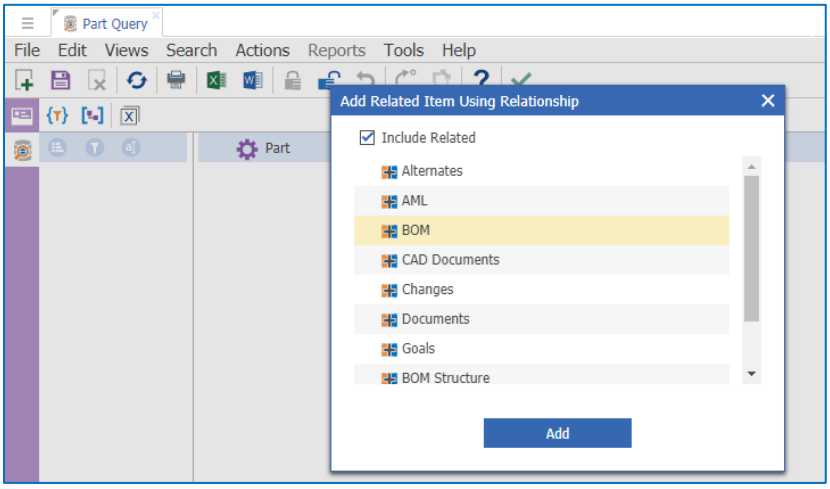

Figure 8.

The query now looks like the following. Note that the alias 'Part_1' is assigned to the related Part Query Element in the list.

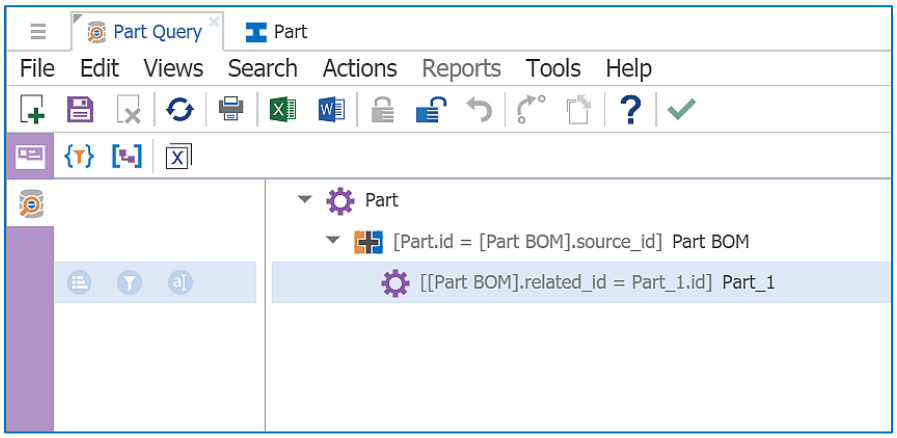

Figure 9.

3. Right-click on the Part Context Item and select **Add Related Item --> Using Relationship**. In this case, Press **CTRL+click** to select **Documents** and **CAD Documents**. Check **Include Related**.

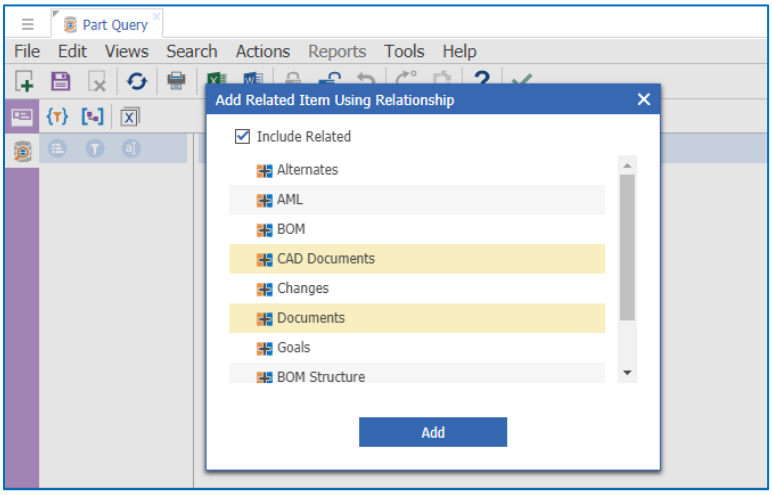

Figure 10.

4. Right-click on the top Context Item and select **Add Related Item --> Using Item Property**. Select an Item Property to add to the query structure. In this case, select **created_by_id**.

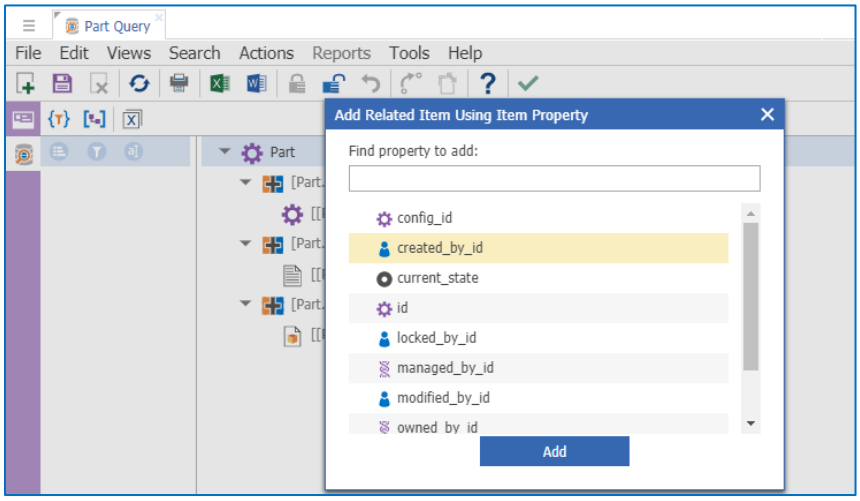

Figure 11.

5. Use the following procedure to specify the desired Properties for each new element in the Query Definition:

Note: Optionally, you can also set Order By settings as well.

a. For [created_by_id] User, specify the following properties:

- `Item ID`
- `First Name`

- Last Name
- Login Name

b. For [Relationship] Part CAD, specify the following property:
   - Sequence

c. For [Related] CAD, specify the following properties:
   - id
   - Document Number
   - Name
   - State

d. For [Relationship] Part Document, specify the following property:
   - Sequence

e. For [Related] Document, specify the following properties:
   - id
   - Document Number
   - Name
   - State

6. Change the Alias on the **CAD Documents** element to “Part CAD”:
   a. Right-click on **CAD Documents** and select **Change Alias**.

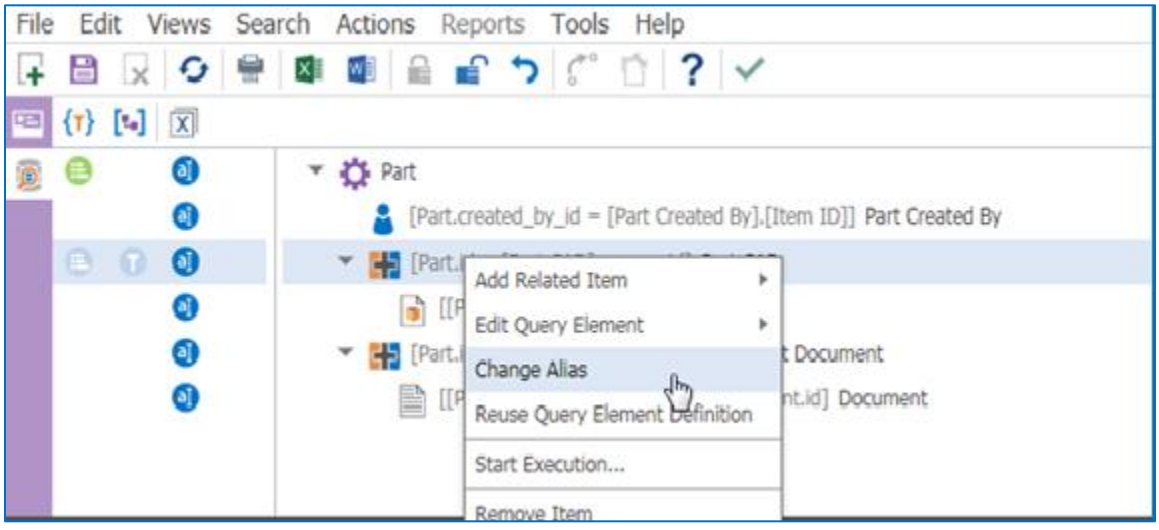

Figure 12.

   b. Enter **Part CAD** as the name and click the check mark.

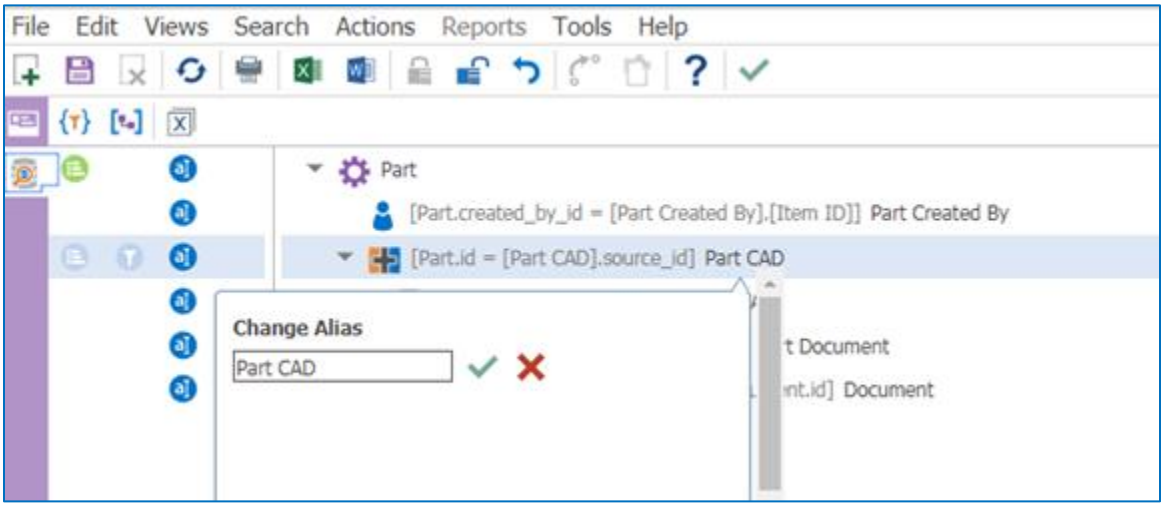

Figure 13.

7. Use the same procedure to change **Documents** Relationship to **Part Documents**.
8. Use the same procedure to change **User** Relationship to **Part Created By**.

Note: Each element on a given level must have a unique alias.

9. Save the Query Definition.

Building an Upward Relationship Structure

1. Right-click on the top Part element and select **Add Related Item --> Using Referencing Item**.

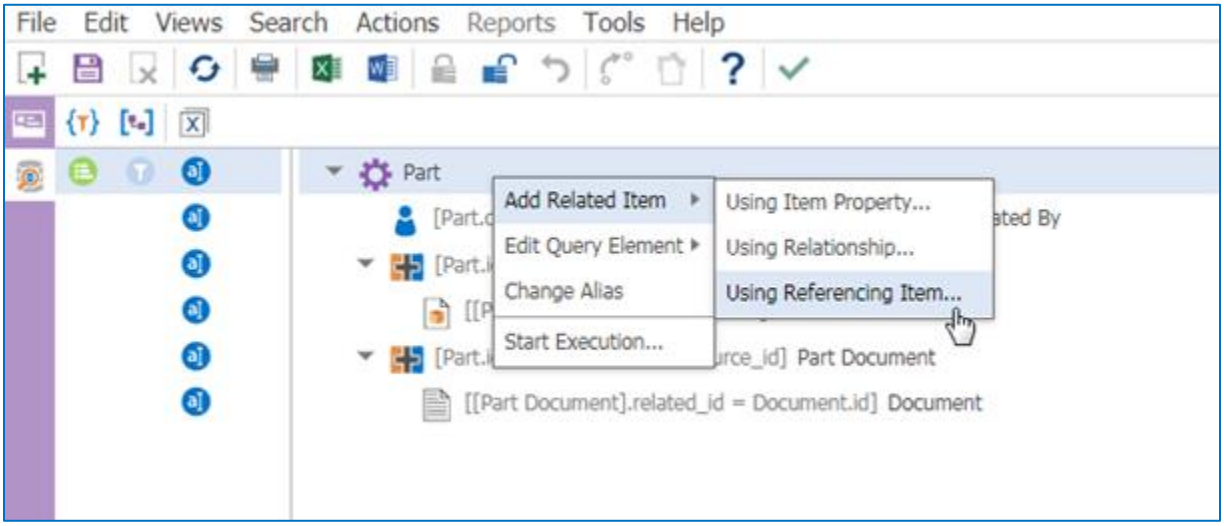

Figure 14.

2. Select an appropriate Referencing item from the structure. In this case, select **Simple MCO Part**.

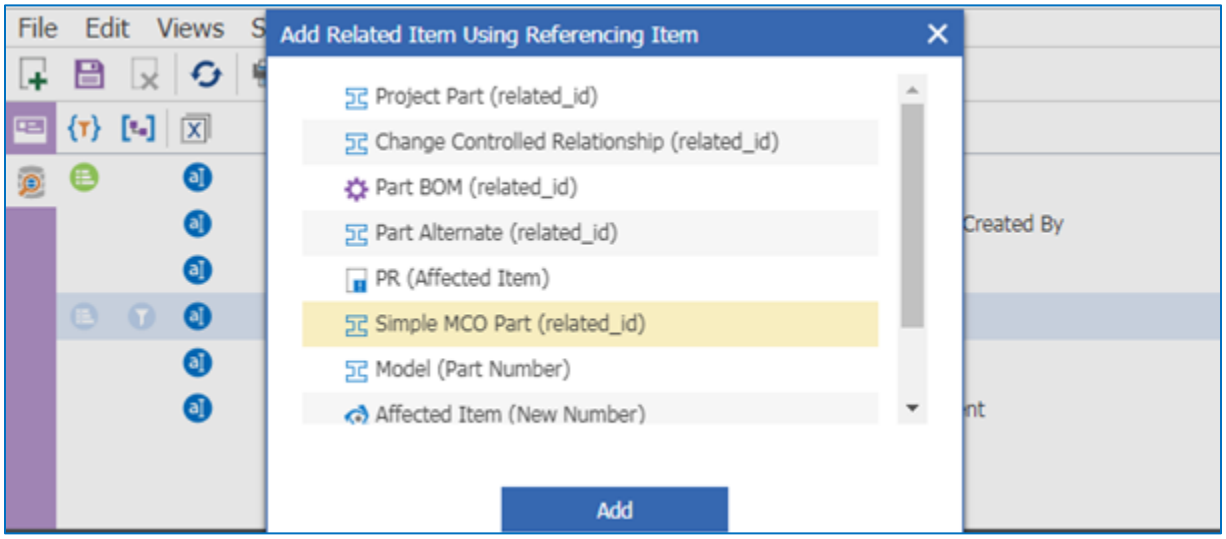

Figure 15.

3. Right-click on the Simple MCO Part element and select Add Related Item --> Using Item Property to select source_id.

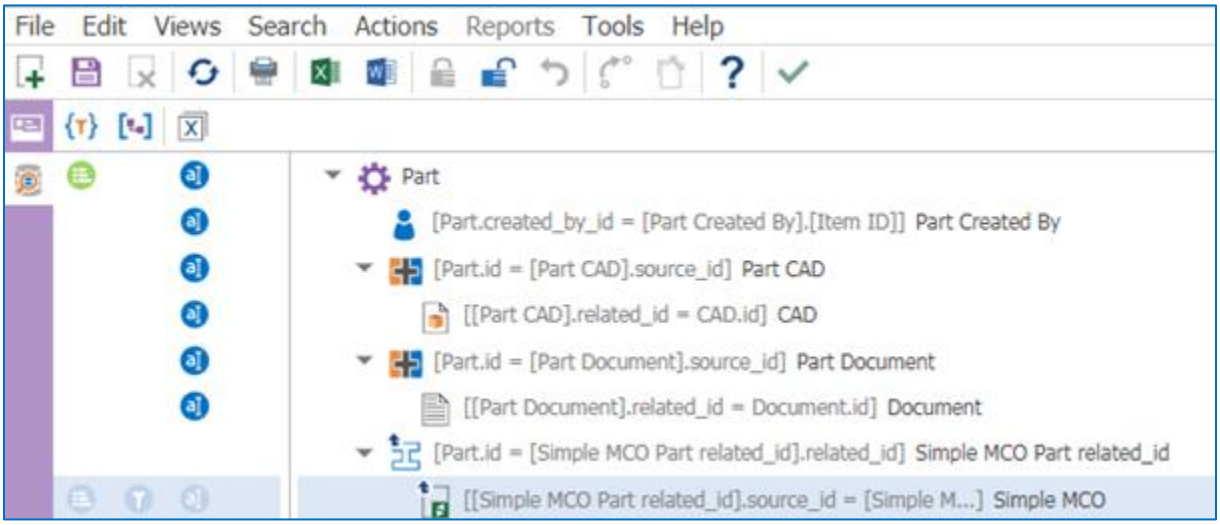

Figure 16.

4. Set Selected Properties on the Simple MCO element to:
   - `id`
   - `MCO Number`
   - `Status`

- Date Originated

5. Save the Query Definition.

1.1.1 Adding a Recursive Call Structure

Recursive structures can reuse sections of the already defined queries. The following example shows how to set up a recursive call for the Part --> Part BOM --> Part relationship structure.

6. Right-click on the top-level **Part** element and add the **Part BOM --> Part** structure as shown.

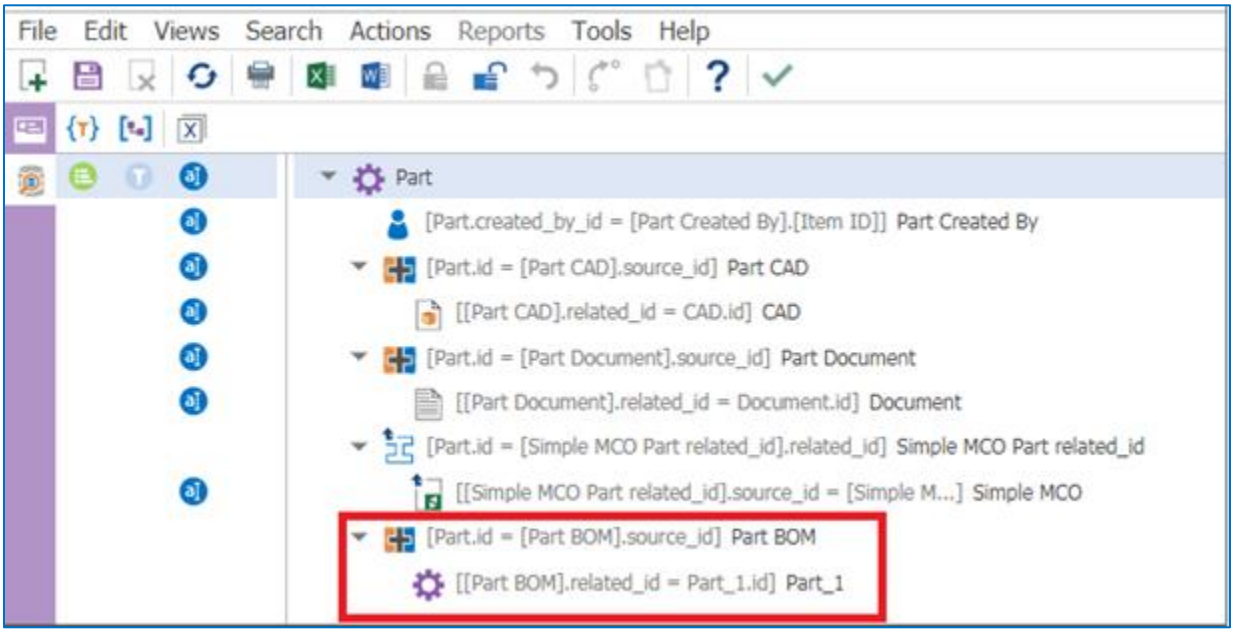

Figure 17.

7. For the BOM element, select the following properties:
   - Sequence
   - Quantity
   - related_id
8. Right-click on the child Part element and select **Reuse Query Element Definition**.

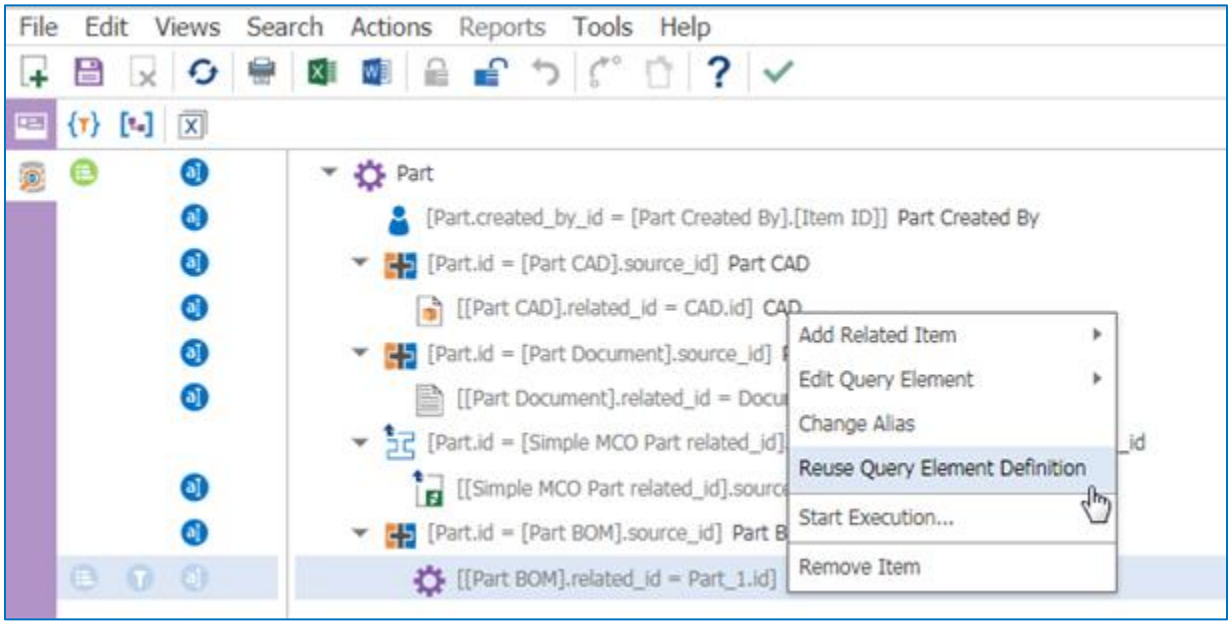

Figure 18.

9. Select the top-level Part element and click **Add**.

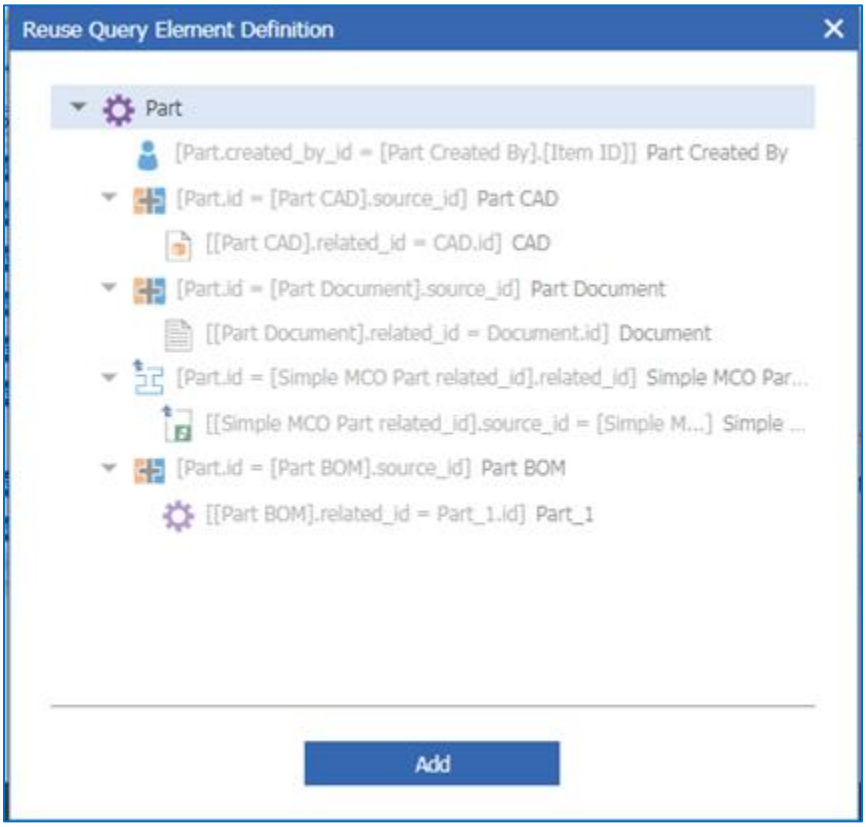

Figure 19.

10. Save the Query Definition item.

The child Part item now includes a search for the same elements as the top-level Part, including CADs, Documents, other Parts, etc. It also includes the same Properties and Order By logic as specified above.

Adding Search Conditions

You can specify Search conditions for every level of the Query Definition Structure. These search conditions can refine the set of data the Query returns. The query logic can include the components shown in the following table:

Comparison Operators	Boolean Operators	Aggregate Functions
= != > >= < <= like is null	AND OR NOT	Count()

Note: Query Builder does not support comparison of string and integer values. For example, you can use logic like `Count(…) = 3`, but you cannot use `Count(…) = "3"`.

Use the following process to add Conditions to the **Query Definition** item:

11. Right-click on the CAD element and select **Edit Query Element --> Condition**.

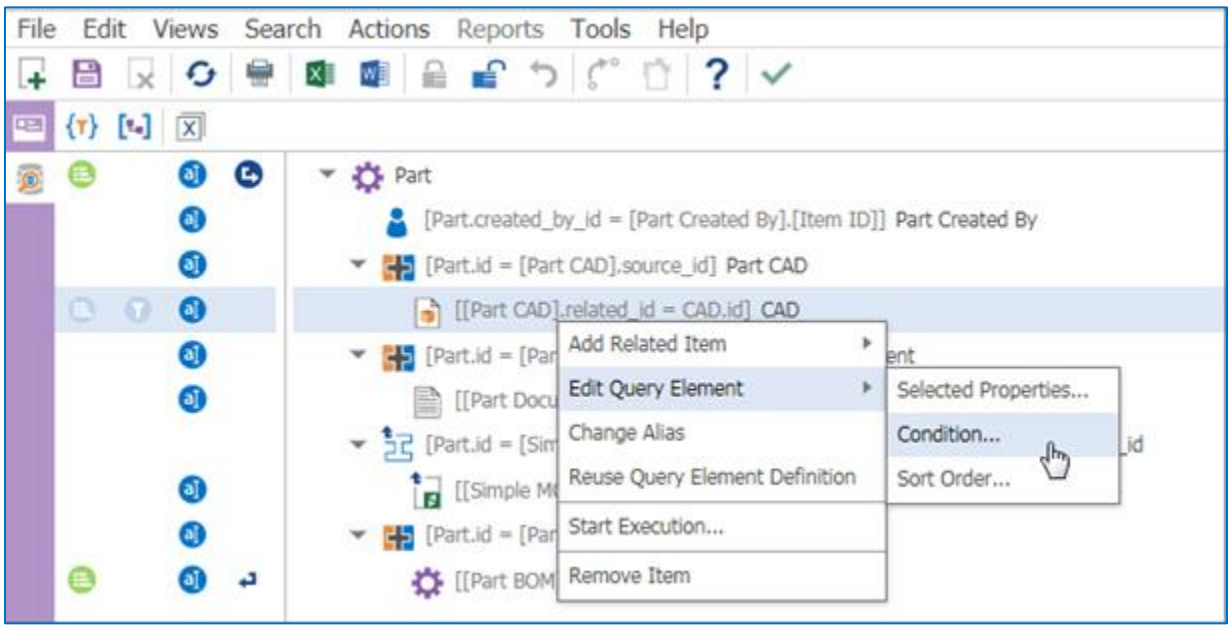

Figure 20.

12. Enter a search condition such as `[Name] = 'CAD_1'`

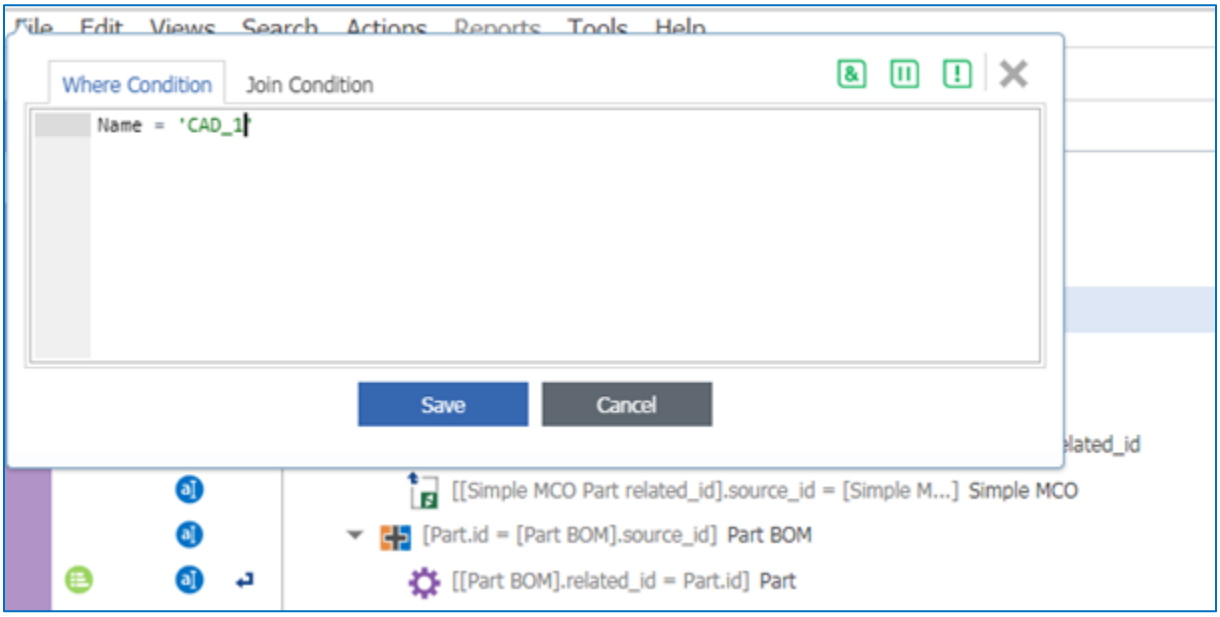

Figure 21.

Note: Conditions written in Query Builder must be separated by white space. For example, you can use the condition `Cost >= 0` but not `Cost>=0`.

13. Right-click on the Document element and select **Edit Query Element --> Condition**.
14. Enter a search condition such as `[Document Number] like 'A27%'`

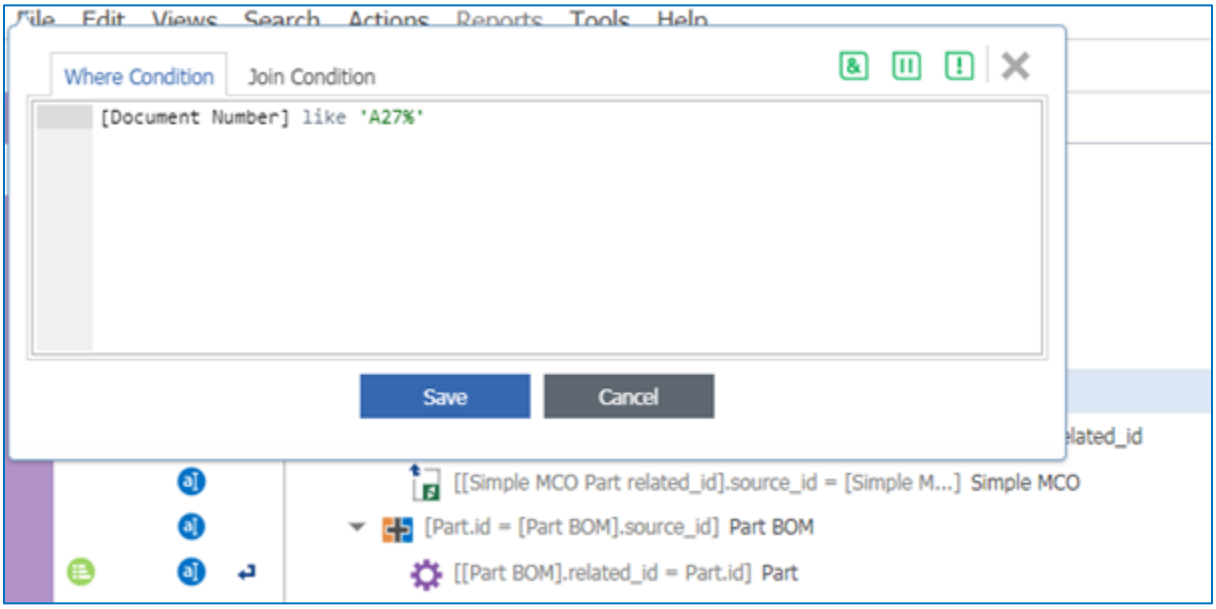

Figure 22.

Note: Brackets are placed around multi-word properties in the condition window of Query Builder. You may notice that these brackets automatically disappear for single-word properties.

15. Right-click on the top Part element and select **Edit Query Element --> Condition**.
16. Enter a search condition that enforces the CAD and Document Relationships to exist:

```
(Count([[Relationship]] Part CAD) >= 1) AND (Count([[Relationship]]
Part Document) >= 1)
```

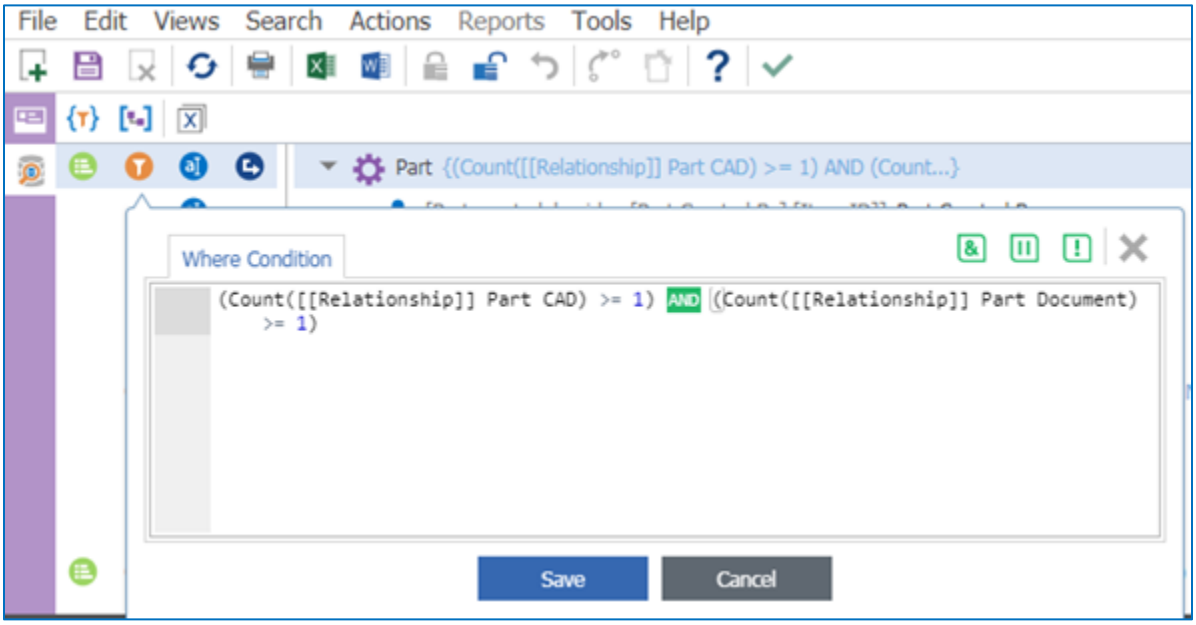

Figure 23.

17. Save the Query Definition.

Chapter 3. Executing Queries

The query can be executed at any time by right-clicking on the Query structure or by selecting the option from the **Actions** menu.

Note: The results of the query are limited by the User's standard "Get" permissions.

Executing the query from the top-level element or from the **Actions** menu executes the entire query structure. However, executing the query from one of the child elements returns the results for that substructure only.

Running the **Start Execution** command results in a window that looks similar to the following:

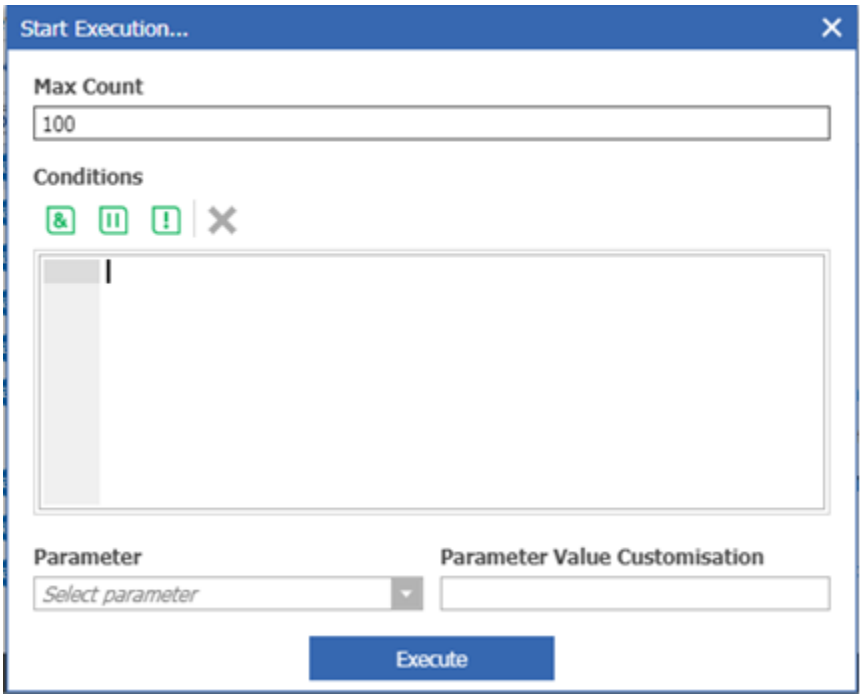

Figure 24.

The window offers an opportunity to add conditions on top of the ones defined by the search and to set **Max Count**. The **Max Count** is intended to limit the number of top-level items returned by the search. This setting is recommended for development because the number of elements returned by the query could potentially be very large.

When ready, press the **Execute** command to run the search.

The search for the **Part Query** defined in Section 2 Developing Query Definitions would return a structure similar to the following:

```
<AML>
	<Item type="Part">
		<id keyed_name="part_7">82A0B82AAD1C43309075640D21117E90</id>
		<modified_by_id keyed_name="Innovator
Admin">30B991F927274FA3829655F50C99472E</modified_by_id>
		<name>part_7</name>
		<state>Preliminary</state>
		<item_number>part_7</item_number>
		<Relationships>
			<Item type="Part CAD">
				<sort_order>128</sort_order>
				<Relationships>
					<Item type="CAD">
						<current_state
keyed_name="Preliminary">A29C9720FF0D4950A140A90BEEA903DA</current_state>
						<id
keyed_name="cad_1">C94B5097E5894EDA8C39C310F735832B</id>
						<name is_null="1" />
						<item_number>cad_1</item_number>
					</Item>
				</Relationships>
			</Item>
			<Item type="Part Documents">
				<sort_order>1</sort_order>
				<Relationships>
					<Item type="Document">
						<id
keyed_name="Doc_01">FFB47BECC6E64E8BBB8254F0A61E9EEC</id>
						<name>User Guide</name>
						<state>Preliminary</state>
						<item_number>Doc_01</item_number>
					</Item>
				</Relationships>
			</Item>
			<Item type="Part Created By">
				<id keyed_name="Super User">AD30A6D8D3B642F5A2AFED1A4B02BEFA</id>
				<login_name>root</login_name>
				<first_name>Super</first_name>
				<last_name>User</last_name>
			</Item>
			<Item type="Simple MCO Part (related_id)">
				<Relationships>
					<Item type="Simple MCO">
						<created_on>2018-03-21T15:59:26</created_on>
						<id keyed_name="MCO-
100001">A5DCE160DE28465A8ECDFF37CEA6B75C</id>
						<state>New</state>
						<item_number>MCO-100001</item_number>
					</Item>
				</Relationships>
			</Item>
		</Relationships>
	</Item>
```

```
</AML>
```

Note: All child elements of the top-level Part item are returned as `Relationships`, including **created_by_id** and **Simple MCO**.

Chapter 4. Sample Query with xProperties

This section describes using Query builder to create a query using a Part item containing an xProperty.

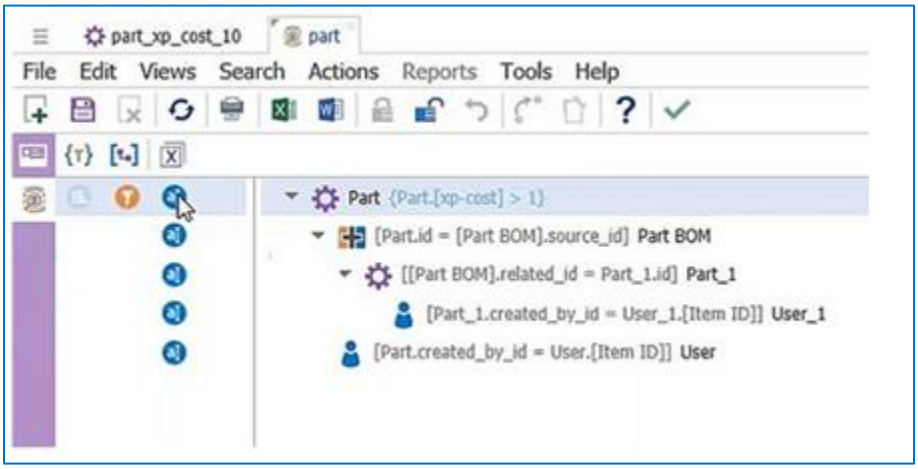

Figure 25.

The first level of the query contains a Part item that has the **xp-cost** property associated with it. Click the Part Properties button to view this xproperty:

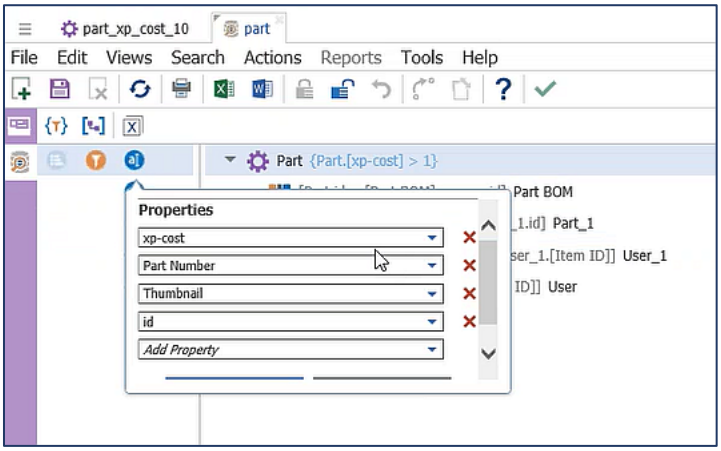

Figure 26.

Creating a Where Condition

You can create a Where condition using the xpcost property by clicking on the Show Condition button. The Condition box appears:

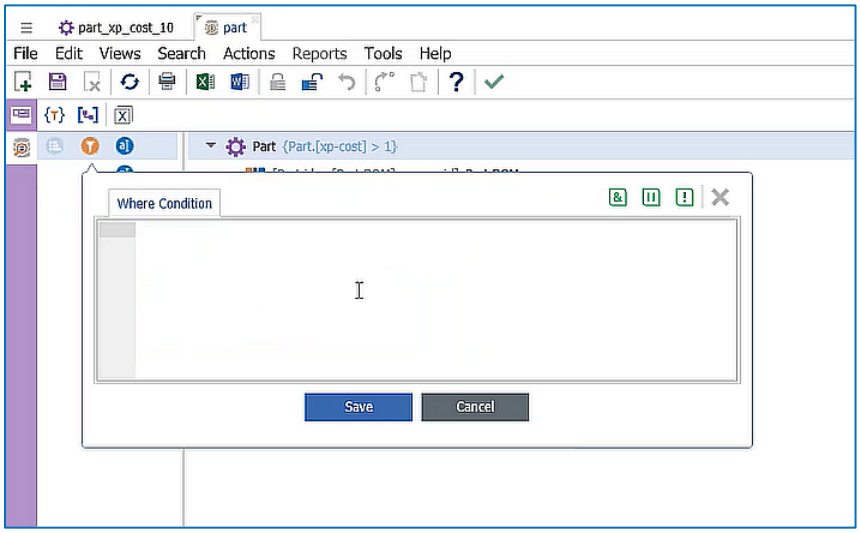

Figure 27.

18. Enter the following in the Where Condition dialog to return the number of items where the cost is greater than 50:

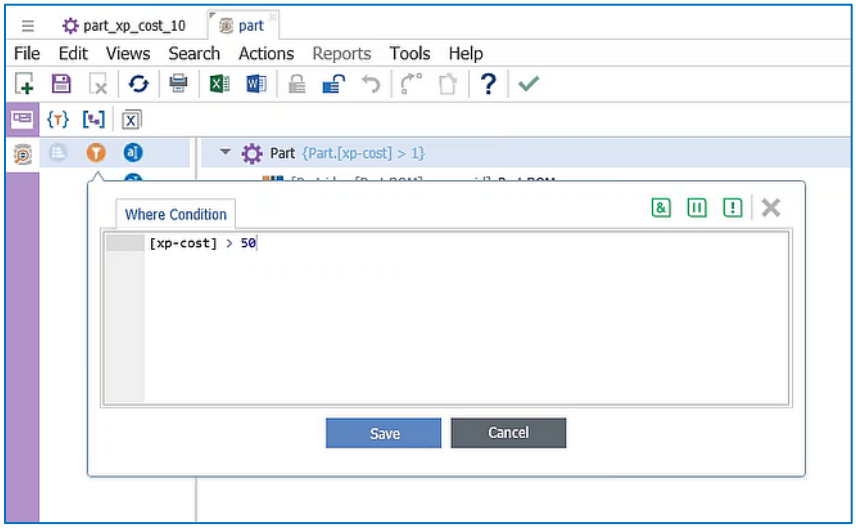

Figure 28.

19. Click **Save**.
20. Right click Part and select Start Execution:

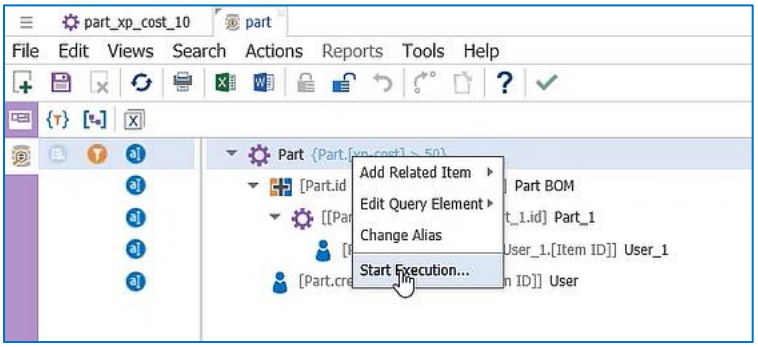

Figure 29.

The Query Result dialog appears:

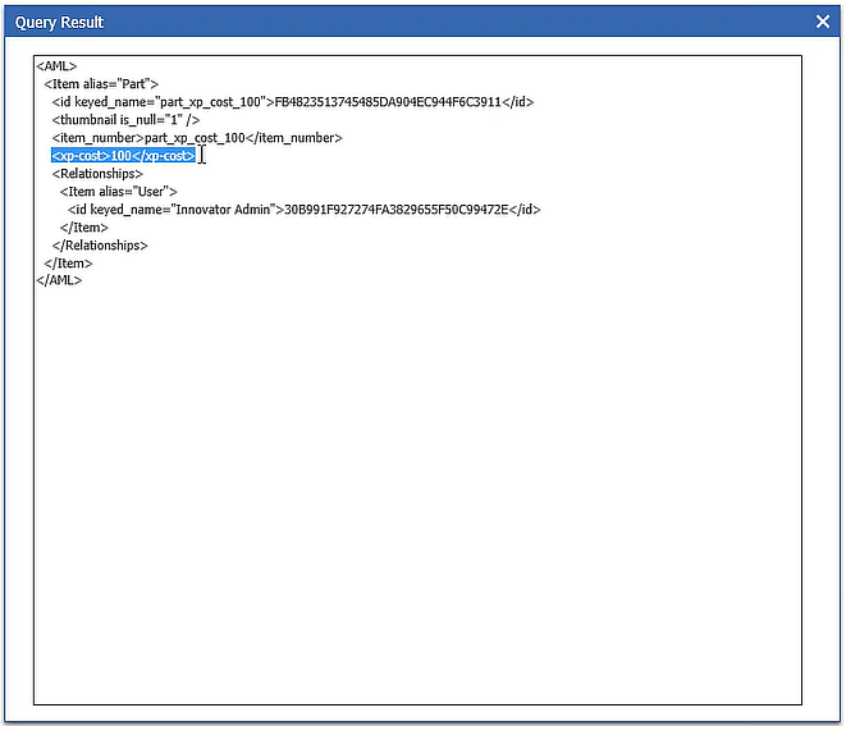

Figure 30.

As you can see, this query only contains one part item where the cost is greater than 50. You can modify the Where condition so xp-cost > 100. This returns two parts in the result:

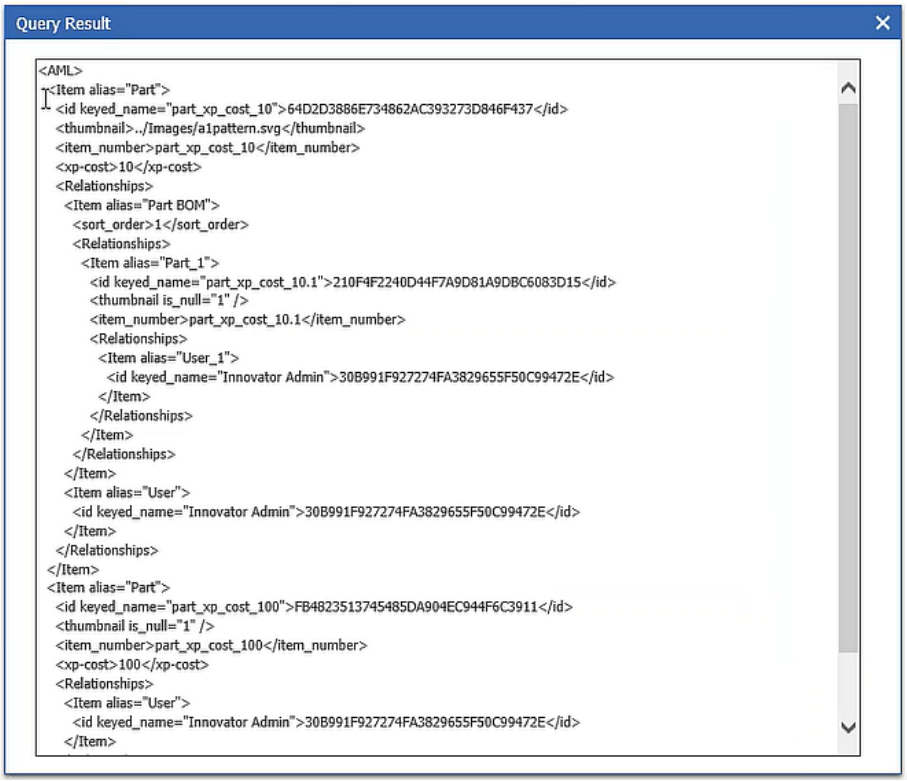

Figure 31.

Both parts contain the xProperty xp-cost.