Aras Innovator 12

Query Builder Guide

Document #: 12.0.02019054301 Last Modified: 6/10/2020

Copyright Information

Copyright © 2020 Aras Corporation. All Rights Reserved.

Aras Corporation 100 Brickstone Square Suite 100 Andover, MA 01810

Phone: 978-806-9400 Fax: 978-794-9826

E-mail: Support@aras.com

Website: https://www.aras.com

Notice of Rights

Copyright © 2020 by Aras Corporation. This material may be distributed only subject to the terms and conditions set forth in the Open Publication License, V1.0 or later (the latest version is presently available at http://www.opencontent.org/openpub/).

Distribution of substantively modified versions of this document is prohibited without the explicit permission of the copyright holder.

Distribution of the work or derivative of the work in any standard (paper) book form for commercial purposes is prohibited unless prior permission is obtained from the copyright holder.

Aras Innovator, Aras, and the Aras Corp "A" logo are registered trademarks of Aras Corporation in the United States and other countries.

All other trademarks referenced herein are the property of their respective owners.

Notice of Liability

The information contained in this document is distributed on an "As Is" basis, without warranty of any kind, express or implied, including, but not limited to, the implied warranties of merchantability and fitness for a particular purpose or a warranty of non-infringement. Aras shall have no liability to any person or entity with respect to any loss or damage caused or alleged to be caused directly or indirectly by the information contained in this document or by the software or hardware products described herein.



Table of Contents

Se	end U	s Your	Comments	4		
Do	ocum	ent Cor	nventions	5		
1	1 Overview					
2	Dev	eloping	Query Definitions	7		
	2.1 2.2	Defining Building	a Context Item a Relationship Structure	8 11		
		2.2.1 2.2.2 2.2.3	Building a Downward Relationship Structure Building an Upward Relationship Structure Adding a Recursive Call Structure			
	2.3 2.4	Adding Adding/	Search Conditions Editing Query Parameters	22 26		
		2.4.1 2.4.2	Structure Resolution Parameter Structure Resolution Configuration for Part BOM Structures	29 31		
3	Exe	cuting (Queries	33		
	3.1	Saving	Query Results to a Local File	35		
		3.1.1	Save Query Results			
4	Sam	ple Qu	ery with xProperties			
	4.1	Creating	g a Where Condition			
5	Usir	ng Max	and Min Aggregate Functions in a Query	40		
6	Sam	ple Qu	ery with Exists() Function			
7	Stru	cture R	esolution Parameters	50		
	7.1	OOTB E	Example: BOM Structure Resolution	50		
		7.1.1 7.1.2 7.1.3 7.1.4	Display Hidden Tabs Under Query Definition Check Parameter Tab Query Definition Events Tab Reference to Parameter in Join Condition in Query	51 51 		
	7.2	Enabling	g Structure Resolution in a Query Definition	53		



Send Us Your Comments

Aras Corporation welcomes your comments and suggestions on the quality and usefulness of this document. Your input is an important part of the information used for future revisions.

- Did you find any errors?
- Is the information clearly presented?
- o Do you need more information? If so, where and what level of detail?
- Are the examples correct? Do you need more examples?
- What features did you like most?

If you find any errors or have any other suggestions for improvement, indicate the document title, and the chapter, section, and page number (if available).

You can send comments to us in the following ways:

Email: <u>Support@aras.com</u> Subject: Aras Innovator Documentation

Or,

Postal service: Aras Corporation 100 Brickstone Square Suite 100 Andover, MA 01810 Attention: Aras Innovator Documentation

Or,

FAX: 978-794-9826 Attn: Aras Innovator Documentation

If you would like a reply, provide your name, email address, address, and telephone number.

If you have usage issues with the software, visit https://www.aras.com/support/



Document Conventions

The following table highlights the document conventions used in the document:

Table 1: Document Conventions

Convention	Description
Bold	Emphasizes the names of menu items, dialog boxes, dialog box elements, and commands. Example: Click OK .
Code	Code examples appear in courier font. It may represent text you type or data you read.
Yellow highlight	Code highlighted in yellow draws attention to the code that is being indicated in the content.
Yellow highlight with red text	Red text highlighted in yellow indicates the code parameter that needs to be changed or replaced.
Italics	Reference to other documents.
Note:	Notes contain additional useful information.
Warning	Warnings contain important information. Pay special attention to information highlighted this way.
Successive menu choices	Successive menu choices may appear with a greater than sign (>) between the items that you will select consecutively. Example: Navigate to File > Save > OK .



1 Overview

The Query Builder application enables administrators to build reusable AML queries using a graphical user interface (GUI). The intended use of the Query Builder application is to create queries that can be used in other applications, e.g. Tree Grid Views, Graph Navigation, Access Control, etc. Query Builder can be thought of as the common mechanism that allows users to define and execute queries against Aras Innovator data and data structures to provide useful results for any application needing it.

A Query can be defined to traverse a Relationship structure downward as in a sub-assembly, or upward in a "where-used" manner depending on business requirements. In addition, results can be refined using Where Conditions to produce highly selective output. User-defined Parameters allow input of search parameters to be deferred until runtime.

Specific property values to be returned from the Query can be defined, including xProperties. Output can be sorted according to user-specified order. (For more information about xProperties and xClasses, refer to the Aras Innovator Extended Classification document.)

The Query Builder User Interface includes an **Execute Query** function that displays query results to a subwindow to provide a preview of Query results. This allows end users to test their query, examine the results, and refine their search as needed.

A built-in **Exists()** function allows users to filter items based on the existence or absence of related items. Users can also duplicate Query Definition items and use them to create a Query Definition that is similar to the original.

A common usage of Query Builder results is to be formatted and displayed in a Tree Grid View. The Tree Grid View can generate a Relationship Tab for an ItemType where end users can access the structure display easily. (For more information on the Tree Grid View, refer to the *Aras Innovator 12.0 – Tree Grid View Administrator Guide.*)

	(*) 📢 📢	
Q	0	✓ ✿ Part (Part.[xp-cost] > 50)
		[Part.id = [Part 80M].source_id] Part BOM
		<pre>[[Part BOM].related_id = Part.id] Part (Part.[xp-cost] > 50)</pre>
		Qart.id = [Part CAD].source_id] Part CAD
		[[Part CAD].related_id = CAD.id] CAD (CAD.Name = 'CAD_1')
		[[Part Document].related_id = Document.id] Document (Document.[Document Number] LIKE 'A27%')
		[Part.created_by_id = [Part Created By].[Item ID]] Part Created By
		Part.id = [Simple MCO Part related_id].related_id] Simple MCO Part related_id
		[[Simple MCO Part related_id].source_id = [Simple M] Simple MCO

This guide walks through building the following sample query:

Figure 1.

©2020 Aras Corporation All Copyrights Reserved.



2 Developing Query Definitions

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



Figure 2.

When you click Create New Query Definition, the following screen appears:

Query Definition × Query Definition 2					
Save V Done Delete	Save Vone Delete				
Query Definition					
Name Description	Context Item Type				

Figure 3.

Each Query Definition starts with an initial **Context Item** before building it out to a more extensive structure. This specifies the "*root*" of the query, from where query of structures and property values will begin.

^{©2020} Aras Corporation All Copyrights Reserved.



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.

2.1 Defining a Context Item

The following example creates a Query Definition item for Part ItemType as Context Item. Create a new query definition and specify a unique Name and a Context Item Type.

F	Part Query 🏠 🖂	× • •
	A Query Definition Name Co Part Query Part	ntext Item Type rt
	Description	
	Figure 4.	

1. Click

. The Show Editor icon appears in the ribbon.

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



👼 Part A Query 🗙				
Part A Query	☆ 🗖			
Save V Don	e 😣 Discard	छ ् । * ~	· 🖻 · 📢 •	•••
😑 📢 🛤				
Ø	> 🔅 Part	Add Related Item 🕨]	
		Edit Query Element 🕨	Selected Properties	
		Change Alias	Condition	5
		Start Execution	Sort Order	

Figure 5.

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





Figure 6.

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.



Part Query X	
Part Query 🖄	
Save V Done	😣 Discard 🕢 🖉 🔆 - 💼 - 🛃 - 🛃 - 🖓
😑 💎 🖼 👯	
ه 🕥 😑 🟹	🔅 Part
Property	Order Direction
id 👻	± ∓ Descending ▼ ×
Part Number 👻	± ∓ Descending ▼ ×
Name 🔻	± ∓ Descending ▼ ×
State	± ∓ Descending ▼ ×
managed_by_id 🔻	± ∓ Descending ▼ ×
Add Property 🔹	× .



6. Save the Query.

2.2 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 *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.

2.2.1 Building a Downward Relationship Structure

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



Part Query ×				
👼 Part Query 😭 🖡				
Save Vone Siscard 6 (* 1) *** a v v 1 ····				
😑 💎 🖼 👯				
و ۱ 😑 💆	Part Add Related Item	Using Item Property		
	Edit Ouery Element >	Using Relationship		
	Change Alias	Using Referencing Item.		
	Start Execution			

Figure 8.

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



Figure 9.



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 10.

 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 11.



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**.

Add Related Item Using Item Property	\times
Find property to add:	
☆ config_id	^
created_by_id	
o current_state	
🔅 id	
locked_by_id	
∑ managed_by_id	~
Add	

Figure 12.

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.	Ī

- 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.

👼 Part Query 🗙				
Part Query 🗘 K				
Save V Done	😣 Discard 💋 🔗 🔆 🖬 📩 💼 🗸 📢 📫			
😑 💎 🛤 👯				
ه 😑 🧕	V 🔅 Part			
	✓ S [Part.id = [Part BOM].source_id] Part BOM			
	<pre>[[Part BOM].related_id = Part_1.id] Part_1</pre>			
	Part.id = [Part Document].source_id] Part Document			
	[Part Document].related_id = Document.id] Document			
a	[Part.created_by_id = User.[Item ID]] User			
	> 🚱 [Part.id = [Part Document_1].source_id] Part Document_1			
۵	Part.id = [Part CAD].source_id] Part CAD Add Related Item			
a	[[Part CAD].related_id = CAD.id] (Edit Query Element			
	Change Alias			
	Reuse Query Element Definition			
	Start Execution			
	Remove Item			

Figure 13.

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



Part Query ×				
Q	Part Query 🕁 K			
4	Save V Done	😢 Discard 🧭 🦿 👫 🗸 💼 🗸 📢 👘		
=	(▼) [■] {\$}			
Q	e ()	~ 🔅 Part		
		Part.id = [Part BOM].source_id] Part BOM		
		<pre>[[Part BOM].related_id = Part_1.id] Part_1</pre>		
		Section 2017 Se		
		[[Part Document].related_id = Document.id] Document		
	٥	Part.created_by_id = User.[Item ID]] User		
		> 🍪 [Part.id = [Part Document_1].source_id] Part Document_1		
	٥	[Part.id = [Part CAD].source_id] Part CAD		
	0	[[Part CAD].related_id = CAP Change Alias Part CAD ✓		

Figure 14.

- 7. Use the same procedure to change **Documents** Relationship to **Part Document**.
- 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.

2.2.2 Building an Upward Relationship Structure

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



👼 Part Query 🗙				
👼 Part Query 🕁 📕				
Save V Done S I	Discard 🕢 🕫	[★•] = · · · · · · · · · · · · · · · · · ·		
😑 🚺 🛤				
ب 📵 🗧 🖉	C. Dart			
	Add Related Item	Using Item Property		
	Edit Query Element 🕨	Using Relationship		
	Change Alias	Using Referencing Item rt_1		
	Start Execution	Document].source_id] Part Document		
	[[Part Do	ocument].related_id = Document.id]		
۵	[Part.created]	by_id = User.[Item ID]] User		
	> 🚫 [Part.id = [Par	t Document_1].source_id] Part Document_1		
٥	✓ S [Part.id = [Part CAD].source_id] Part CAD			
a	[[Part CA	D].related_id = CAD.id] CAD		



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





Figure 16.

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



👳 Part Query 🗙	
👼 Part Query 🏠	
Save Vone	8 Discard 🔗 😤 🔆 × 💼 × 💽 •••
😑 📢 🛤 👯	
ه 😑 🧕	~ 🔅 Part
	Part.id = [Part BOM].source_id] Part BOM
	<pre>[[Part BOM].related_id = Part_1.id] Part_1</pre>
	✓ S [Part.id = [Part Document].source_id] Part Document
	[Part Document].related_id = Document.id] Document
a	[Part.created_by_id = User.[Item ID]] User
	> 🚫 [Part.id = [Part Document_1].source_id] Part Document_1
(1)	✓ ♥ [Part.id = [Part CAD].source_id] Part CAD
٩	[[Part CAD].related_id = CAD.id] CAD
	Part.id = [Simple MCO Part related_id].related_id] Simple MCO Part related_id
	[[Simple MCO Part related_id].source_id = [Simple M] Simple MCO

Figure 17.

- 4. Set Selected Properties on the Simple MCO element to:
 - id
 - MCO Number
 - Status
 - Date Originated
- 5. Save the Query Definition.

2.2.3 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.

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



👼 Part Query 🗙	
Part Query 🟠	
Save V Done	🗴 Discard 🔗 🌮 🔆 🗸 🗟 🗸 🔩 🖓 🚥
😑 📢 🛤	
ه 😑 🧕	✓ ♥ Part
	Part.id = [Part BOM].source_id] Part BOM
	<pre>[[Part BOM].related_id = Part_1.id] Part_1</pre>
	Part.id = [Part Document].source_id] Part Document
	[Part Document].related_id = Document.id] Document
a	[Part.created_by_id = User.[Item ID]] User
	> 🚱 [Part.id = [Part Document_1].source_id] Part Document_1
a	Section 2017 Se
ા	[[Part CAD].related_id = CAD.id] CAD
	Part.id = [Simple MCO Part related_id].related_id] Simple MCO Part related_id
a	[[Simple MCO Part related_id].source_id = [Simple M] Simple MCO

Figure 18.

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



Dart Query X	
👼 Part Query 🏠	
Save V Done	😣 Discard 🚱 🜮 🔆 🗸 💼 🗸 📢 🖬 🗸
😑 📢 🛤	
ه 😑 👩	-> 🔅 Part
aj	Part.id = [Part BOM].source_id] Part BOM
000	[[Part BOM].related_id = Part_1.id] Part_1
	Image: Ward with a state of the state of
a	Part.created_by_id = Us Reuse Query Element Definition
	> 🌀 [Part.id = [Part Documen Start Execution
٩	✓ ✓ ✓ ✓ ✓ Part.id = [Part CAD].sou Remove Item
0	[[Part CAD].related_id = CAD.id] CAD
a	 [Part.id = [Simple MCO Part related_id].related_id] Simple MCO Part related_id [Simple MCO Part related_id].source_id = [Simple M] Simple MCO

Figure 19.

4. Select the top-level Part element and click Add.







5. 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.

2.3 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
=	AND	Count()
! =	OR	
>	NOT	
>=		
<		
<=		
like		
is null		

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:

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

👼 Part Query 🗙						
👼 Part Query 🏠 📕						
Save V Done	😢 Discard 🚱 🦿 🔆 🔹 💼 🗸 🔩 🖓 🚥					
😑 💔 📢						
ه 😑 🙍	V 🔅 Part					
٥	✓ S [Part.id = [Part BOM].source_id] Part BOM					
	[[Part BOM].related_id = Part_1.id] Part_1					
	✓ S [Part.id = [Part Document].source_id] Part Document					
	[Part Document].related_id = Document.id] Document					
a	[Part.created by id = User.[Item ID]] User					
	> S [Part.id = [Part Document 1].source id] Part Document 1					
a	[Part id = [Part CAD] source id] Part CAD					
8 0 0						
	Add Related Item					
	V Part.id = L Edit Query Element Selected Properties					
a)	[[Sim Change Alias Condition D					
	Reuse Query Element Definition Sort Order					
	Start Execution					
	Remove Item					



©2020 Aras Corporation All Copyrights Reserved.



2. Enter a search condition such as [Name] = 'CAD_1'Na

Part Query ×		
Where Condition	Join Condition	
Name = 'CAD_1'		
	Save Cancel	
	Save Cancel	

Figure 22.

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.

3. Right-click on the Document element and select Edit Query Element --> Condition.

Dart Query X			
Dart Where Co	ndition Join Condition	& II ! ×	
Save [Docum	ent Number] LIKE 'A27%'		
	Save Cancel		
😑 🕤 🔍	[[Part Document].related_id = Do	cument.id] Document {Document.[Doc	ument Number] LIKE 'A27%'}

4. Enter a search condition such as [Document Number] like 'A27%'

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.

 $\ensuremath{\textcircled{\sc 0}}$ 2020 Aras Corporation All Copyrights Reserved.



Figure 23.

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

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

Directory X	
👼 Part Query 🕁 📕	
Save Vone Siscard 6 (* 1) *** 1 ***	•
Image: Second state Image: Second state<	}
Where Condition (Count([[Relationship]] Part CAD) >= 1)AND (Count([[Relationship]] Part Document) >= 1)	

Figure 24.

7. Save the Query Definition.



2.4 Adding/Editing Query Parameters

The following example walks you through the process of defining query parameters. It uses Test Query as an example:

🔊 Test 🗙		
👼 Test 🕁 🏳		
Edit 🚱 🦿	*	
🚍 (7) 📢 👯		
Q	v 🛟 Part	
	✓ 🍪 [Part.id = [F	Part Document].source_id] Part Document
	[[Part	Document].related_id = Document.id] Document
	Figure 25.	
. Click the Edit Paramete Query Parameters	rs icon 👫 . The Query Pa	arameters dialog box appears.
Actions • No Related •	🔅 🍭 🗙 🚳 🚺	≗ ≝ ℰ Q ⊠ ⊘ 🔽 👻
Name	Label	Default Value []

Figure 26.

©2020 Aras Corporation All Copyrights Reserved.



2. Click the **New** icon to add a parameter. Enter the following information:

Name = doc_state Label = Document State Default Value = *

ctions - No Related - 🏠	🔅 🗙 🖾 🖬 🔒 🔐	ć° Q 🛛 🖉 🔽	
ime	Label	Default Value []	
c_state	Document State	*	

Figure 27.

- 3. Close the dialog box to save the parameter information.
- 4. Select the **Document** level of the relationship structure and click the associated Filter button to filter using the parameter you just created.
- 5. Set any other filters and sort orders that you need.
- 6. Save the Query definition.



The parameters that you add to the Query definition appear in the Map Parameters dialog box in the Tree Grid View:

Мар	Parameters						×
< >	¢						
Actions	s 🔻 No Related	New relation	onship 🕥 Pick/Re	place Related Item	🗙 Delete relations	hip	*
Visible	Name	Label	Default Value	Data Type	Data Source []	Pattern	
	doc_state	Document State	*				

Figure 28

Once mapped, the parameters become available in the Tree Grid View under the "Modify Parameters" command:

A BOM BOM Structure	Altern	ates	AML Docun	nents CAD	Docur
🖳 🙀 🏞 🏹 🔳	ÌC,	Ģ	×		
Part Number		R	State	Seque	Quan
🖃 – 🏠 Part1		А	Released	128	1
Part1_1		С	Released	128	1
Part1_3		А	Preliminary	256	2
🖃 🏠 Part2		С	Released	256	2
Part2_1		А	Released	128	2
🖃 🔅 Part4		А	Released	384	4
Part4_1		А	Released	128	1

©2020 Aras Corporation All Copyrights Reserved.



This allows the entry or selection of the configured parameters from the Tree Grid View, setting query parameters in the underlying Query Definition from the Tree Grid User Interface at runtime.

Parameters can also be used for previewing query results when Executing a Query in the Query Builder. See Section 3.0 "Executing Queries" for an example of this.

2.4.1 Structure Resolution Parameter

For structures of Versionable items (like Part/Part BOM Structures) it is useful to examine query results in a Tree Grid View by either the latest versions, or by released versions only, or a combination where if no released components exist then the latest (un-released) version is displayed in the overall structure. The default is to display the structure as it is stored in the database without any optional resolution.

These Resolution Modes are provided to Part BOM Query Definitions (and Tree Grid Views) by a special Parameter which is pre-configured OOTB. Structure Resolution can be applied to any Item Relationship structure of Versionable items that follows the same rules as Part BOM structures – Life Cycle State "Released" (is_released Boolean), incremental Revisions with interim versioned generations, etc.

2.4.1.1 Default Resolution Mode

The **Default** Resolution Mode uses the versions of related child Parts as stored on the BOM Relationships. This is the standard resolution in Aras Innovator.

Part Number	Revision	State
⊡ 🛟 Child A	В	Preliminary
Child A.1	А	Preliminary
Child A.2	С	Released
Child A.3	С	Preliminary
Er Child B	В	Released
Child B.1	А	Released
Child B.2	В	Released
Erro Child C	А	Preliminary
Child C.1	А	Preliminary
Child C.2	В	Released

Figure 1. **Default** resolution mode

2.4.1.2 Latest Resolution Mode

When the **Latest** resolution mode is selected, the structure is resolved starting from the top using the latest versions of child items. When a child item is resolved to its latest version, then its child items are resolved using this resolution mode. This is done recursively.

The Latest resolution mode does not consider the released state of the child Parts.



Part Number	Revision	State
⊡ 🏠 Child A	В	Preliminary
Child A.1	А	Preliminary
Child A.2	С	Released
Child A.3	С	Preliminary
⊡… 🏠 Child B	В	Released
Child B.1	А	Released
Child B.2	D	Preliminary
⊡ 🛟 Child C	А	Preliminary
Child C.1	А	Preliminary
Child C.2	В	Released

Figure 2. Latest resolution mode

The latest revision **D** of **Child B.2** (Preliminary) is shown where this Part version is in the **Preliminary** state, although the default structure under revision **B** of **Child A** (this structure) refers to revision **B** of **Child B.2** (Released) as stored in the database. In other words, the Latest Resolution Mode dynamically provides a view to in-work changes to released parts that are not yet committed to the released structure.

2.4.1.3 Latest Released Resolution Mode

When the **Latest Released** resolution mode is selected, the structure is resolved starting from the top using the latest versions of child items that are in Released state. When the child item is resolved to its latest Released version, then its child items are resolved using this resolution mode. This is done recursively.

If the related child item has no versions that are in Released state, then a blank row is displayed for that item. There is no further processing on that item.

Part Number	Revision	State
⊡ 🛟 Child A	А	Released
Child A.2	С	Released
Child A.3	В	Released
⊡ 🛟 Child B	В	Released
Child B.1	А	Released
Child B.2	C	Released
¢		

Figure 3. Latest Released resolution mode

The latest Released versions of child parts are resolved. A blank row is displayed for **Child C** because it has no version that is in the Released state.



2.4.1.4 Latest Released Or Latest Resolution Mode

When the **Latest Released or Latest** resolution mode is selected, the structure is resolved starting from the top using the latest versions of child items that are in Released state. If there are no versions that are in Released state, then the latest version is used. When the child item is resolved to a version, then its child items are resolved using this resolution mode. This is done recursively.

Part Number	Revision	State
⊡ 🛟 Child A	А	Released
Child A.2	С	Released
Child A.3	В	Released
⊡ 🛟 Child B	В	Released
Child B.1	A	Released
Child B.2	С	Released
🖃 🔆 Child C	А	Preliminary
Child C.1	А	Preliminary
Child C.2	В	Released

Figure 4. Latest Released or Latest resolution mode

The latest Released versions of child parts are resolved. **Child C** does not have any version that is in the Released state. Therefore, its latest Preliminary version is resolved.

2.4.2 Structure Resolution Configuration for Part BOM Structures

In Aras Innovator, the BOM Structure grid (a TGV under the "BOM Structure" tab in the Part form) is preconfigured with a Structure Resolution Parameter, thus it supports *Latest*, *Released*, and *Released or Latest* modes OOTB:

A BOM BOM Stru	Parameters ✓ ★	×
🖳 👯 🐤 🟹	Property	Value
Part Number	StructureResolutionMode	Latest Released Or Latest
Part1_1		Default Latest Latest Released Latest Released Or Latest
Part2		·

In the event you need to enable Structure Resolution for your Query Definition, you will need to add a Parameter that has a default value of "*Aras.Resolution.EntryPoint;Default*", and then also follow the additional detailed steps for configuring Structure Resolution Parameters described in **Section 7.0**.



Query Parameters			×
Actions 🗸 No Related 🔽 🔅 🗙 🚺		° Q 🛛 🖉 🔽	•
Name	Label	Default Value []	
StructureResolutionMode		Aras.Resolution.EntryPoint;Default	

For more details and extensive examples of how Resolution Modes are applied in BOM Structures, Effectivity, and 3D Visualization please refer to the following documents:

- Aras Innovator 12 Dynamic Visualization
- Aras Innovator 12.0R2 Product Engineering User Guide



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. In the case of a table row that contains relationship and related_item information you must have "Get" permissions for the related item, otherwise the cells in the row that correspond to the related_item will appear as blank/empty.

Query Builder/Tree Grid View can be configured to use the "keyed_name" associated with the related_item. If there is no "Get" access to the related_item, the keyed_name value will appear as "Restricted."

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:

Dert Dert	t Query 🗙			
Ø F	Part Qu	iery 🕁		
٩	Save	/ Done	S Discard 6 C * * i * * I Start Execution ×	
₩ {	(*) 🖬 🕴	Ŷ	Max Count	
Q	8 0	0	V V Part {(Count([[Relationship]] Part CAD) >= 1) AND (Cou	
		0	> (Signature_id) Part BOM Conditions	
			> S [Part.id = [Part Document].source_id] Part Docume	
		0	[Part.created_by_id = User.[Item ID]] User	
			> 😘 [Part.id = [Part Document_1].source_id] Part Docu	
		0	> 🍪 [Part.id = [Part CAD].source_id] Part CAD	
			> D [Part.id = [Simple MCO Part related_id].related_id]	
			Parameter Parameter Value Customication	
			Select parameter	
			Execute	

Figure 5.

The window offers an opportunity to add conditions on top of the ones defined by the search, enter parameter values, and set **Max Count**. If Conditions and/or Parameters are unspecified the query is executed without any filtering via conditions or parameters. The Max Count is intended to limit the number of top-level items returned by the search. These settings are recommended during development because the number of elements returned by the query could potentially be very large.

©2020 Aras Corporation All Copyrights Reserved.



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</pre>
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>2019-03-21T15:59:26</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.



3.1 Saving Query Results to a Local File

Using the Query Execution Dialog, users can opt to save the results returned to a local file instead of having the results displayed in a pop-up dialog. Saving the results in this manner is for convenience when users would like to review query results using a separate editor.

3.1.1 Save Query Results

To save the results from executing a Query Definition to a local file, select the check box labeled '**Result** as **File**' prior to executing the query.

Parameter	P	arameter V	alue Custor	nisation	
Select parameter	•				
Result as File					
	Execu	te			

Figure 6. Save Query Results to a File

Once the results are returned to the client, the web browser will provide an interface for the user to select a folder and file name to store the results. The default file name uses the Query Definition name with an '.xml' extension. For example, MyQueryDefinition.xml.



4 Sample Query with xProperties

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



Figure 7.

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 8.

4.1 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:



	Part ((Count([[Relationship]]	Part CAD) >= 1) AND (Count)	
Where	: Condition	& III × >= 1) AND (Count)	
		ument Number] LDKE 1	A27%
0	Save	Cancel related_id].source_id = [Simple M] Simple MCO	

Figure 9.

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

Where Condition [xp-cost] >	• 0		>= 1) AND (Count)
			ument Number] LIKE 'A27%')
0	Save Cancel	d = [Simple M] Simple MCO	

Figure 10.

- 2. Click Save.
- 3. Right click Part and select Start Execution:





Figure 11.

The Query Result dialog appears:

Query Result	\times
<aml> <item alias="Part"> <id keyed_name="part_xp_cost_100">FB4823513745485DA904EC944F6C3911</id> <id keyed_name="part_xp_cost_100">FB4823513745485DA904EC944F6C3911</id> <thumbnail is_null="1"></thumbnail> <item_number>part_xp_cost_100</item_number> <xp=cost>100</xp=cost>100<!--/xp=cost-->100 <kelationships> <item alias="User"> <id keyed_name="Innovator Admin">308991F927274FA3829655F50C99472E</id></item></kelationships></item></aml>	

Figure 12.



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:

AML>	
<item alias="Part"></item>	
- <id keyed_name="part_xp_cost_10">64D2D3886E734862AC393273D846E437</id>	
<tbumbneil>/Images/a1pettern.svg</tbumbneil>	
<item_number>part_xp_cost_10</item_number>	
<xp-cost>10</xp-cost>	
<relationships></relationships>	
<item alias="Part BOM"></item>	
<sort_order>1</sort_order>	
<relationships></relationships>	
<item alias="Part_1"></item>	
<id keyed_name="part_xp_cost_10.1">210F4F2240D44F7A9D81A9DBC6083D15</id>	
<thumbnail is_null="1"></thumbnail>	
<item_number>part_xp_cost_10.1</item_number>	
<relationships></relationships>	
<item alias="User_1"></item>	
<id keyed_name="Imovator Admin">30B991F927274FA3829655F50C99472E</id>	
<item alias="User"></item>	
<id keyed_name="Innovator Admin">308991F927274FA3829655F50C99472E</id>	
<item alias="Part"></item>	
<id keyed_name="part_xp_cost_100">FB4823513745485DA904EC944F6C3911</id>	
<thumbnail is_null="1"></thumbnail>	
<item_number>part_xp_cost_100</item_number>	
<xp-cost>100</xp-cost>	
<relationships></relationships>	
<item alias="User"></item>	
<id keyed_name="Innovator Admin">30B991F927274FA3829655F50C39472E</id>	

Figure 13.

Both parts contain the xProperty xp-cost.



5 Using Max and Min Aggregate Functions in a Query

The following example demonstrates how to use the Max() aggregate function in a Where clause to identify the latest related part using the generation property. The Max() aggregate function returns the maximum value of a specified property. Similarly, the Min() aggregate function returns the minimum value of a specified property. You can use a statement like this in the conditional logic applied to either a Join or Where clause when defining a Query Definition.

1. In the Navigation pane, select Administration>Configuration>Query Definitions. The following menu appears:



Figure 14.

2. Click Create New Query Definition. A blank query definition window appears:



Query Definition × Query Definition 2	
Save V Done Delete	
Cuery Definition	
Name	Context Item Type
Description	



- 3. Enter the Query title in the Name field.
- 4. Click the ellipses in the **Context Item Type** field and select **Part**.
- Click Save
 The Show Editor icon appears in the ribbon.



D Late	_{stReleased} × .atestReleasedExan	nple 🕁 🖂
	Edit 🕑 🔗 🔆 ~	i ∽
	∧ Query Definition	
	Name	Context Item Type
	Description	<u>r'all</u>

Figure 16.

6. Click the Show Editor icon. The following window appears displaying the context item.

反 LatestReleased 🗙	
LatestRelease	edExample 🕁 🖂
🖍 Edit 🖌 🌮	× · ∎ · •••
🔚 (7) 📢 👯	
	Part Part



7. Right-click on the context item and select Add Related Item>Using Item Property.



Discrete LatestReleased ×		
🖻 LatestReleasedExample ☆ 💻		
Save Vone Siscard 6 (* 1990) Save		
	Dart	
	Add Related Item	Using Item Property
	Edit Query Element 🕨	Using Relationship
	Change Alias	Using Referencing Item
	Start Execution	

Figure 18.

The following dialog box appears:

Config_id	
created_by_id	
O current_state	
🗘 id	
locked_by_id	
ğ managed_by_id	

8. Select config_id and click Add.



👳 LatestReleased 🗙	
LatestRelease	edExample 🕁 📕
Save V Done	😢 Discard 🕢 🌮 🔆 🗸 🗟 <
😑 💎 🖬 👯	
Ø	V 🔅 Part
	<pre>[Part.config_id = Part_1.id] Part_1</pre>



9. Right-click the Where condition icon and select **Change Alias** from the context menu:

Discrete LatestReleased ×		
🖻 LatestReleasedExample 😭 💻		
Save Vone Siscard 6 (* 1) *** in * * 1 ***		
🔚 🚺 📢		
Ø	✓ ✿ Part	
	<pre>[Part.config_id = Part_1.id] Part Add Related Item</pre>	Þ
	Edit Query Element	►
	Change Alias	
	Reuse Query Element Definitio	n
	Start Execution	
	Remove Item	

Figure 20.



The Change Alias dialog appears.

👮 LatestReleased 🗙	
DatestRelease	edExample ☆ 📕
Save V Done	😢 Discard 🕢 🖉 🔆 - 💼 - 💽 - 💼 -
😑 💎 ы 👯	
Q	V 🔅 Part
8 9 0	[Part.config_id = Part_1.id] Part_1
	Change Alias Part_1

Figure 21.

- 10. Change the alias to Part_LatestReleased_Filter and click \checkmark .
- 11. Click the **Where** condition icon to create the following join condition for the child part:





Figure 22.

12. Click the **Where** condition icon to create the following where condition for the query:

DiatestRelea	sed ×
👼 Lates	stReleasedExample ☆ 📕
Save	✓ Done S Discard \checkmark ($^{\circ}$ $\overset{\bullet}{}$ $\overset{\bullet}{}$ $\overset{\bullet}{}$ $\overset{\bullet}{}$ $\overset{\bullet}{}$ $\overset{\bullet}{}$
😑 (T) 📢	\$8î
Ø	V 🔅 Part
0	[Part.config_id = Part_LatestReleased_Filter.id] Part_LatestReleased_Filter {Part_LatestReleased_Filter.Released = 1}
	Where Condition Released = 1 Join Condition Part.config_id = Part_LatestReleased_Filter.id

Figure 23.

13. Click Save.

^{©2020} Aras Corporation All Copyrights Reserved.



14. Select the parent Part and click the Where condition icon to create the following expression:

🙍 LatestReleased 🗙	
👳 LatestReleasedExample 🏠 📕	
Save Vone Siscard 6 C 🔆 View	
😑 🕶 ы	
De Part	
Where Condition	:_LatestReleased_Filter.Released = 1}
<pre>generation = max(Part_LatestReleased_Filter.generation)</pre>	
Save Cancel	



15. Click **Save**. The Show Editor window displays the parent and child parts.

👳 LatestReleased 🗙	
LatestRelease	edExample 🚖 💻
Save V Done	😒 Discard 🕑 🦿 🖾 👻 🚅 🗸 🖬 🗠 📢 🚅
😑 (7) 💽 👯	
ø 😶	Part {Part.generation = max(Part_LatestReleased_Filter.g}
•	Part_config_id = Part_LatestReleased_Filter.id] Part_LatestReleased_Filter {Part_LatestReleased_Filter.Released = 1}

Figure 25.

The query returns the most recent version of released parts associated with a particular config_id.



6 Sample Query with Exists() Function

The Exists() Function can be applied in a WHERE condition to filter Items based on the *existence* of related items. In essence, it's similar to a LEFT INNER JOIN in a SQL statement. To explain the use of such a feature, refer to the basic Query Definition shown here:

(*) 🛤 👯	
	🔅 Part
	Figure 26.

This query will result in returning all Parts in the system. It's equivalent to the SQL statement: SELECT FROM Part

The following addition to this Query Definition will add related Parts through the Part BOM Relationship:





This query will return all Parts, including those Parts that have a Part BOM Relationship along with the related 'child' Part. It's equivalent to the SQL statement:

SELECT FROM Part p LEFT JOIN Part BOM pb on ... LEFT JOIN Part child on...

The following addition to this Query Definition will add the Condition Exists([[Relationship]] Part BOM) on the Context Part Query Item:





SELECT FROM Part p

^{©2020} Aras Corporation All Copyrights Reserved.



```
INNER JOIN Part BOM pb on ...
LEFT JOIN Part child on...
```

The effect is to restrict the Part Items returned for the Context Query Item to only those that have related Parts using the Part BOM relationship. Any related Query item can also have conditions applied to it. For example, using the previous example, the Part BOM Query item can have a condition that restricts Items based on a Quantity > 2. In this case, the entire Query Definition will only return Parts that have related Part BOM Items that have a Quantity of at least 3.



The Exists() function can be used on any Query item with related content. Note that the Condition Editor will automatically offer valid Exists() statements. For Example:

W	/here Condition		& 11 !
×	Exi		
	Exists([[Relationship]] Part BOM)	item	
	External ID	item	
	<pre>max([Part BOM].external_id)</pre>	item	
	<pre>min([Part BOM].external_id)</pre>	item	
	<pre>max([Part BOM].eff_expression_strip</pre>	ng_notat	
	<pre>min([Part BOM].eff_expression_strip</pre>	ng_notat	

Figure 30.

In this case, beginning to type the 'Exi' term will isolate the list of valid statements and include all valid ways to use 'Exists()' for the selected Query item.



7 Structure Resolution Parameters

We mentioned Structure Resolution briefly in the Parameters section above. The Structure Resolution parameter sets query criteria to return a specific configuration of items with respect to Released state and version (generation).

Structure Resolution Modes:

- **Default**: Structure as physically stored in the database. Technically speaking, results will contain Items with ID values in related_id Property of Relationships in the structure without interpretation of LifeCycle State or availability of new versions (generations).
- Latest: Uses the generation of the Item with the highest value.
- Latest Released: Uses the generation of the Item with the highest value which is also Released.
- Latest Released or Latest: Uses the generation of the Item with the highest value which is also Released; otherwise the Latest if there are no Released generations.

By default, the structure is only displayed as stored in the database. Without dynamic view parameters users would need to spend more time examining work-in-progress changes to components in a structure. Dynamic structure resolution is important feature for BOM structure, Effectivity, 3D Visualization, and other Aras Innovator features.

In this section we describe in detail how Structure Resolution can be configured for a Query Definition using the Query Builder.

Note: In Aras Innovator 12.0 SP2 and SP3, there was an error in process Structure Resolution Queries for the Dynamic Viewer in that it would assign the Structure Resolution filter to the root CAD Item. Thus, if a Structure Resolution query was meant to include all Latest Released CAD Items (and the root CAD Item was not Released, the query would return nothing. This has been fixed in 12.0 SP4 and later releases.

7.1 OOTB Example: BOM Structure Resolution

Structure Resolution comes pre-configured for Part BOM Structures OOTB. If you wish to enable Structure Resolution for your own Query Definition its useful to examine the OOTB configuration used in Query Definition PE_BomStructure for reference. Other than naming conventions and base query structure you will follow the same procedure as used for the OOTB implementation when adding Structure Resolution to your Query.



7.1.1 Display Hidden Tabs Under Query Definition

By default, Tabs under the Query Definition itemtype are hidden. Query Definition Itemtype have Relationships to Parameters and Event handlers. We will need them displayed to examine and configure Structure Resolution.

As administrator, edit the Itemtype qry_QueryDefinition and change the Default Structure View to "Tabs On" as shown on right. If the tabs do not appear for Query Definition form, then you may need to exit the Query Definition and re-open to refresh the form.

Name	History Template
qry_QueryDefinition	
Singular Label	Plural Label
Query Definition	Query Definitions
Show Parameters Tab	Default Structure View
When Populated 🔽	Tabs On 🔻

7.1.2 Check Parameter Tab

Once displayed, you will have access to relationships under the Query Definition as Tabs. Recall from **Section 2.4** that Parameters are essentially runtime variables configured in the Query Builder to provide runtime variables (i.e. search criteria) to the underlying query.

Under qry_QueryParameter tab, you will see all Query Parameters defined for this query. For Bom Structure OOTB, Parameter "StructureResolutionMode" is pre-configured with the default value "Aras.Resolution.EntryPoint.Default" as shown below.

If adding Structure Resolution to your custom query, you will need to add a Parameter and assign the Default Value "Aras.Resolution.EntryPoint.Default" as shown below. This procedure is detailed in section 7.2.

Query Definition						
Name Context Item Type PE_BomStructure Part BOM						
Description						
<pre>^ qry_QueryItem qry_QueryReference qry</pre>	QueryParameter qry_QueryDefin	itionEvent				
● qry_QueryParameter v ☆						
$\Box = \Box =$						
Name	Label	Default Value []				
StructureResolutionMode		Aras.Resolution.EntryPoint;Default				



7.1.3 Query Definition Events Tab

Beside the Parameters tab in the Query Definition form is the *qry_QueryDefinitionEvent* tab. In the OOTB Bom Structure Query Definition you will see an Event Handler that tells the query to run a method named "*qry_ResolveStructureEntryPoint*" before query execution (event "OnBeforeExecute"). This method is provided OOTB, providing the logic to resolve structures according to the selected Parameter value (Latest, Latest Released, Latest Released or Latest).

When adding Structure Resolution to your custom query, you will need to add this event handler and define the method and event name exactly as set in the BOM Structure query definition.



In Summary, the Parameter and Event Handler tells the Query Definition to (a) allow runtime entry of Latest/Released criteria and (b) specifies method logic to execute to enforce Structure Resolution behavior on execution of a query.

7.1.4 Reference to Parameter in Join Condition in Query

Once the Parameter is added to the Query Definition and the Event Handler set, the Parameter can be dereferenced in the Join Condition of the query item to be resolved using the \$Name syntax below. In the BOM Structure example, the resolved query item is the Part node, therefore the Join condition is edited to use '\$StructureResolutionMode.parent.related_id' = Part.id.

	U J	- 1 9	1						
ø	€	0	a	0	Part BOM {\$EffectivityResolutionConfigurationQueryDefinition}				
	Θ	0	a		✓ ♦ ['\$StructureResolutionM	lode;parent.related_id' = Par] Part			
	₿	Wh	ere C	onditi	on	OM].source_id] Part BOM {\$EffectivityResolutionConfigurationQueryDefinition}			
		Joi '\$S	n Co tructu	nditior	n olutionMode;parent.related_id' = Part.id				

Sections 7.1.1 - 7.1.4 describe the same configuration settings you would use to enable Structure Resolution on your custom query, or another Query Definition applied to a structure of related versionable items such as CAD Structures. The next section covers this procedure in detail.

©2020 Aras Corporation All Copyrights Reserved.



7.2 Enabling Structure Resolution in a Query Definition

Structure Resolution can be applied to any Item Relationship structure of Versionable items that follows the same rules as Part BOM structures – Life Cycle State "Released" (is_released Boolean), incremental Revisions with interim versioned generations, etc.

- CAD Structure is an example of such a structure, and Structure Resolution can be applied to Query Definitions against this data model in the same manner as Part BOM structures. We use CAD Structure as an example of this procedure which is documented in greater detail in *Aras Innovator 12 Dynamic Visualization*

Per the Part BOM Query Definition we examined in Section 7.1, Structure Resolution requires a Parameter with the Resolution Entrypoint, an Event Handler must be added to the Query, and the Parameter must be referenced in the Join Condition for the resolved itemtype.

Adding the Structure Resolution configuration to a CAD Structure Query will involve all of these steps as well.



Step 1: Create Structure Resolution Mode Parameter

Applying a specific Structure Resolution Mode ('Default', 'Latest', etc.) is done using a Query Parameter. Using the Query Parameters Dialog, add a Parameter to be used for Structure Resolution Mode. For example 'StructureResolutionMode' and assign the default value 'Aras.Resolution.EntryPoint; Default'

Note: Be sure to check the syntax of the Default Value for the Parameter. It is used to select a specific List Item as set in the Tree Grid View Definition and described below in Step 4.

Step 2: Incorporate the Query Parameter in Join Condition

The added Query Parameter must be incorporated into the Query Definition in order for it to be used/exposed to the end-user. Add it as part of the Join Condition of the Child CAD Query Item as shown in **Error! Reference source not found.** Replace the default Join Condition – e.g. '[Part B OM].related id = Part.id' with the following –

''\$StructureResolutionMode;parent.related id' = Part.id'



Part BOM and Part are used for this example, but it could just as well be '[myLinkage].related_id =
myNode.id' being replaced by ''\$StructureResolutionMode;parent.related_id' =
myNode.id'.

Note: Be sure to check the syntax of the Join Condition. It must be as specified above (without the outer quotes). This assumes the value StructureResolutionMode was used for the Query Parameter Name. If you named the Parameter differently, e.g. myParam, then you would naturally need to dereference it as \$myParam in the expression.

Note: The Structure Resolution Mode Query Parameter can only be inserted once in a Query Definition. It cannot be used/applied to more than one Query Item.

Step 3: Add Structure Resolution Method to Query Event Handler

Show the Relationship Tabs for the Query Definition in the Form View if they are not shown. Tabs On/Off is set in the Query Definition itemtype (qry_QueryDefinition).

Select the qry_QueryDefinitionEvent Relationship Tab. A Method needs to be added to the OnBeforeExecute event. To do this, select the Select Items Icon to create a Relationship and select the qry_ResolveStructureEntryPoint Method and assign it to the OnBeforeExecute event as shown in Figure 31.

qry_QueryItem qry_QueryReference qry_QueryParameter qry_QueryDefinitionEvent							
😘 Methods 🗸 🏠							
🔕 🖨 🔚 🔀 Q 🚳 Hidden 🗸 💽 ~ 🖳 ×							
Name 🕇	Method T	V	execution_all	Comments	event		
qry_ResolveStructureEntryPoint	CSharp	1	Administrators	(OnBeforeExecute		

Figure 31. Assigning Structure Resolution OnBeforeExecute Method

Step 4: Map the Query Parameter in the Tree Grid View Definition

In order for the Query Parameter to be used in a **Tree Grid View**, it must be enabled (made visible) in the Tree Grid View Definition.

Open the Map Parameters Dialog in the Tree Grid View Definition within the Editor View. All Parameters defined in the Query Definition used by the Tree Grid View will be listed.

Select the Visible check box for the StructureResolutionMode row. The Data Type is a List and the Data Source is the List qry_StructureResolution.

Мар	Parameters						×
~ :	×						
Action	ns - No Related -) 💿 🗙 🖾 🖬	🔒 🔐 🖑 🔍 🚳 ⊘	-			*
Visi	Name Label		Default Value	Data Type	Data Source []	Pattern	
	StructureResolutionMode	Structure Resolution	Aras.Resolution.EntryPoint;Default	list	<u>gry_StructureResolution</u>		



^{©2020} Aras Corporation All Copyrights Reserved.



With this Configuration in place, users will be able to select specific Resolution Modes by opening the Query Parameters Dialog in the Tree Grid View used within the 3D Viewer.

Model Views		
🖳 👯 🏹 💈		
CAD / Part Structure	Parameters	×
🕀 👩 Cylinder Head 1	¹ √ ×	
	Property	Value
	Structure Resolution	Default
		Default
		Latest
		Latest Released
		Latest Released Or Latest

Figure 33. Tree Grid View Parameter Dialog

7.2.1.1 Applying a Where Condition Against Resolved Items

In Aras Innovator 12.0 SP4 and later releases it is possible to add a Where Condition to a Query Item that is used for Structure Resolution (e.g., CAD in this example). Doing so allows the Administrator to apply an additional condition (filter) when resolving the Structure of Items with fixed Relationships (such as CAD / CAD Structure). For example, assuming the CAD Structure in the following diagram:



Figure 34. Structure Resolution Logic with Condition

If the Administrator would like to use Structure Resolution of 'Latest' and apply an additional condition to only include CAD Items that were *In Review* (that is, with a CAD.State = 'In Review') then Where and Join conditions like the following could be added to the Query Item:



0		6	 CAD {CAD.State = 'In Review'} (CAD.id = [CAD Structure].source_id] CAD Structure 	
0	a	¢,	['\$StructureResolutionMode;parent.related_id;ApplyC] CAD {CAD.State = 'In Review	ı'}
	Whe	ere Co '\$stru =R	Image: Save Cancel	

Figure 35. Where/Join Condition Example for Structure Resolution

Note the addition of the left-side of the Join Condition:

`\$StructureResolutionMode;parent.related_id;ApplyChildWhereMode=ResolutionTar
getResolved' = CAD.id]

Executing this Query Definition, with Structure Resolution of 'Latest Released' with return:



Figure 36. Structure Resolution Logic with Condition Result

In this example, only the CAD Item – CS3, version 5 is returned because of the added Where Condition of `CAD.State = `In Review' applied to the resolved results of the Structure Resolution Join Condition using the selected Mode parameter.



For more details and extensive examples of how Resolution Modes are applied in BOM Structures, Effectivity, and 3D Visualization please refer to the following documents:

- Aras Innovator 12 Dynamic Visualization
- Aras Innovator 12.0R2 Product Engineering User Guide

