cubewise code

Arc for TM1

HANDS-ON

Table of Contents

TRAINING ENVIRONMENT

Environment

To go through this training, you will have to have Arc running. Download the latest version of Arc from here: https://code.cubewise.com/arc-download, there is a quick video to show you how to setup Arc.

Once Arc is running, if you want to follow the steps in this training, you will need to set up the training TM1 instance. You can download the TM1 objects from here.

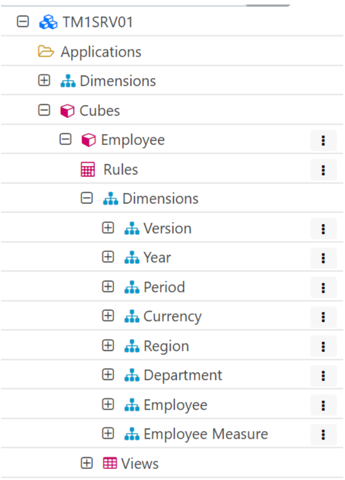
Once setup to connect to the TM1 instance just use login with user **Admin** and no password.

How to access Arc

Go to http://localhost:7070/ and login to the instance (instance name provided by your instructor) using admin and no password

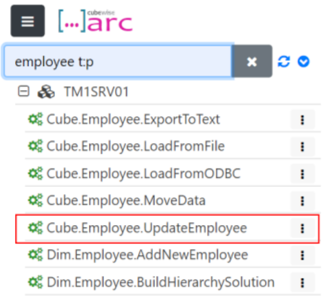
Objective of the training

The objective of this session is to learn how hierarchies can help us to deal with slow-changing dimension. For example, if employees move from one department to another, how can you compare the same dimension between two years. We will focus on the Employee dimension in the Employee cube:



- TM1SRV01
  - Applications
  - Dimensions
  - Cubes
    - Employee
      - Rules
      - Dimensions
        - Version
        - Year
        - Period
        - Currency
        - Region
        - Department
        - Employee
        - Employee Measure
      - Views

During this hands-on we are going to use Arc to build new Employee hierarchies and then execute the process **Cube.Employee.UpdateEmployee**. To find this process, you can type in the search bar **employee t:p**:



employee t:p
TM1SRV01
Cube.Employee.ExportToText
Cube.Employee.LoadFromFile
Cube.Employee.LoadFromODBC
Cube.Employee.MoveData
Cube.Employee.UpdateEmployee
Dim.Employee.AddNewEmployee
Dim.Employee.BuildHierarchySolution

In the Arc search bar, you can filter the search using **t:<type>**, for example, you can use **t:d** to get all the dimensions or **t:p** to get only the processes.

TOPIC 1: ARC FUNDAMENTALS

What is it?

Arc is an integrated development environment (IDE) for IBM Planning Analytics (TM1). It provides a centralized platform for all your TM1 development, access all your on-premise and cloud environments from one location. Built from the ground up to take advantage of the new features in Planning Analytics (TM1 11+), it includes:

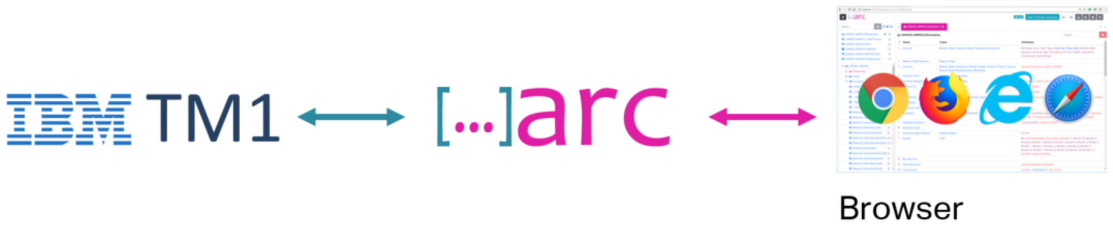
- Single file, 1-minute install.
- Web interface access it from anywhere
- Dimension / Hierarchy editor
- Cube and rule editor
- Turbo Integrator editor
- Chore editor
- Cube viewer
- Syntax highlighting
- Dynamic auto-complete (intellisense) for rules and TI
- Debugging of Turbo Integrator processes
- Quick access to all your favourite TM1 objects
- User-defined snippets
- User-defined plugins and pages

What is it built with?

- Arc is written mainly in JavaScript with a GO server. All operations use the [TM1 REST API](#).
- It is a single executable that has everything included.

Architecture

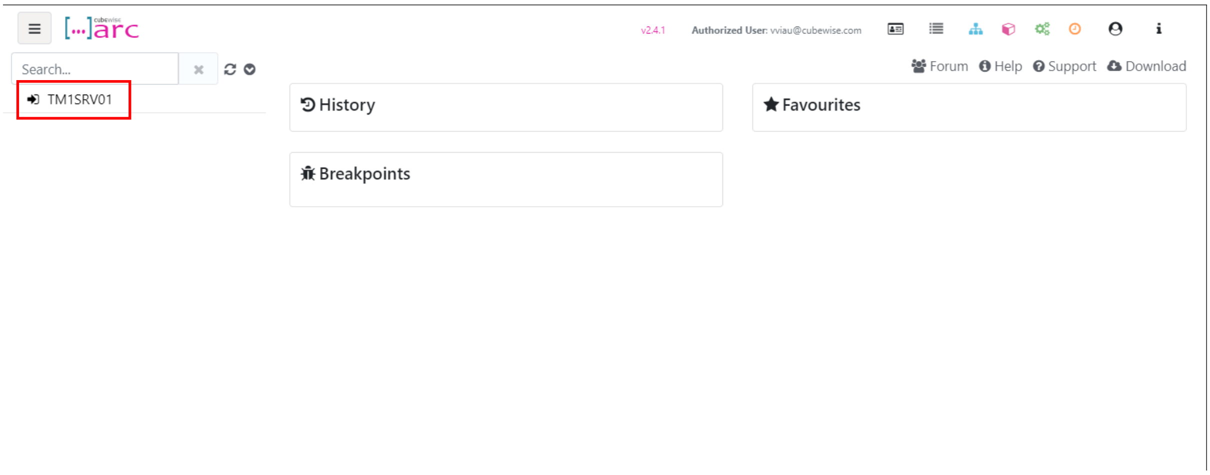
Arc is a single executable which access your TM1 instance using the TM1 REST API. You can access Arc from any web browser:



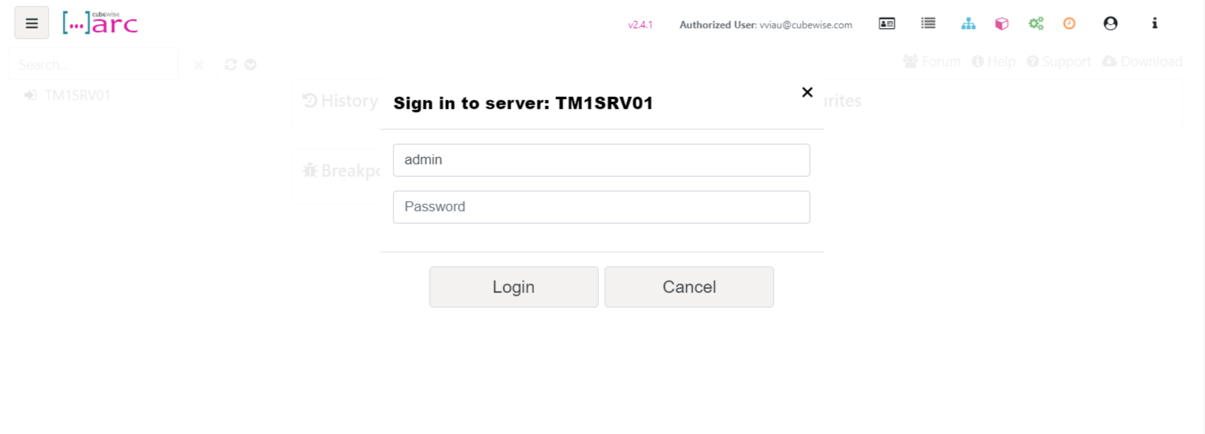
How to access it?

Just run the executable and then go to the Arc URL:

- http://localhost:7070/



Then click on the instance name TM1SRV01 and enter the credentials: **admin** and **apple**

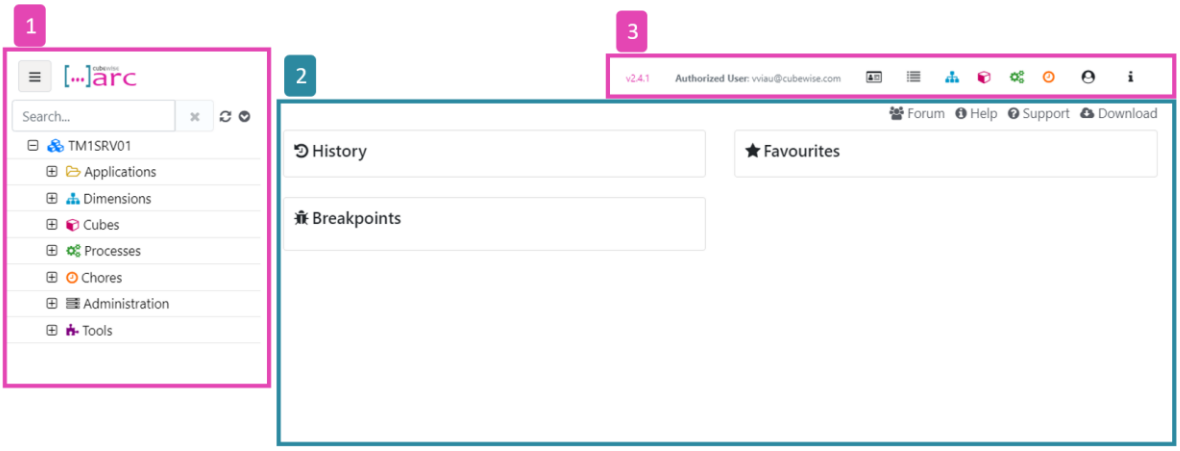

Sign in to server: TM1SRV01

admin

Password

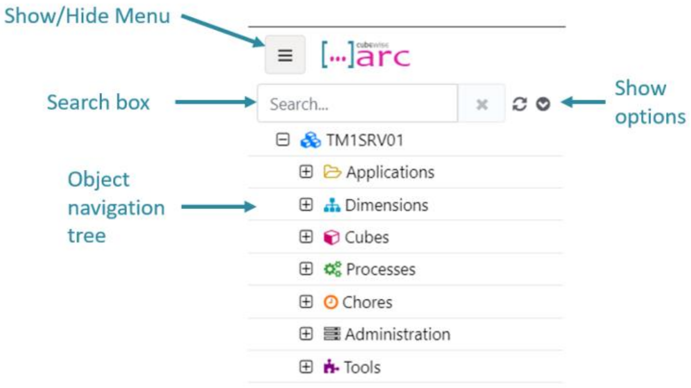
Arc can be split into three main sections:

1. Menu
2. Workbench
3. Top bar

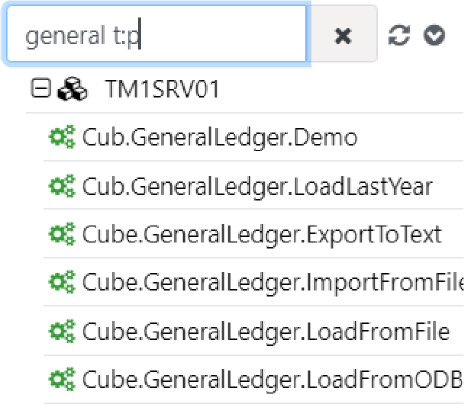


Arc Menu

On the top left, you will see the menu to access all TM1 objects:



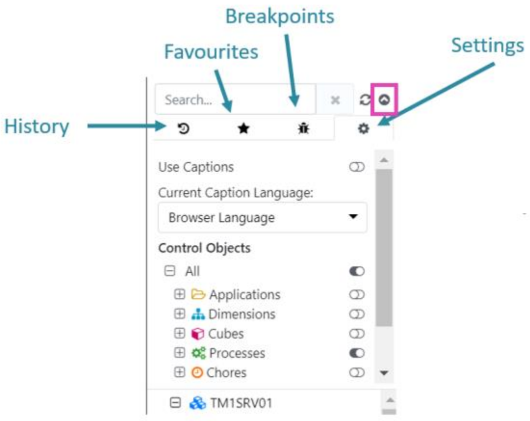
The search feature will give you a quicker access to your TM1 instances. Arc searches in all your TM1 instances. Filter the search results by using **t:p** to see only processes. **t:** works for all object types such as **t:d** to see dimensions, **t:cube** to see cubes and **t:cho** to see chores.



general t:p

TM1SRV01
- Cub.GeneralLedger.Demo
- Cub.GeneralLedger.LoadLastYear
- Cube.GeneralLedger.ExportToText
- Cube.GeneralLedger.ImportFromFil
- Cube.GeneralLedger.LoadFromFile
- Cube.GeneralLedger.LoadFromODB

Inside the options, you will find your history (latest objects open, your favourites, your breakpoints and the settings (browser language, show/hide control objects and toggle adminhosts).



Use Captions

Current Caption Language: Browser Language

Control Objects
- All
- Applications
- Dimensions
- Cubes
- Processes
- Chores

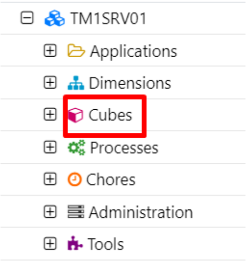
TM1SRV01

Arc enables control objects to be toggled on/off by type. For example, showing only control objects for processes and not showing the dimensions.

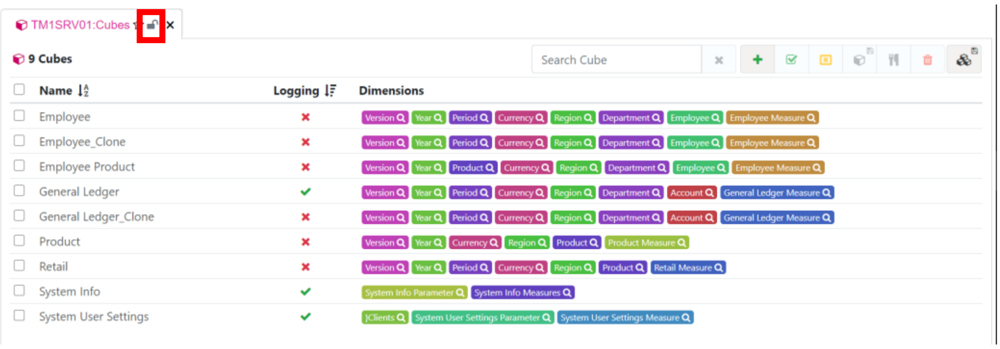
Arc Workbench

By clicking on Dimensions, Cubes, Processes and Chores, you will find a summary page of the TM1 objects list.

Just click on Cubes:

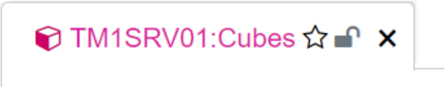


and it will open the Cubes summary page:

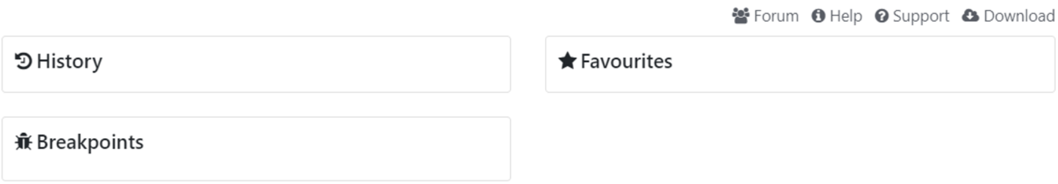


□	Name	Logging	Dimensions
□	Employee	✗	Version, Year, Period, Currency, Region, Department, Employee, Employee Measure
□	Employee_Clone	✗	Version, Year, Period, Currency, Region, Department, Employee, Employee Measure
□	Employee Product	✗	Version, Year, Product, Currency, Region, Department, Employee, Employee Measure
□	General Ledger	✓	Version, Year, Period, Currency, Region, Department, Account, General Ledger Measure
□	General Ledger_Clone	✗	Version, Year, Period, Currency, Region, Department, Account, General Ledger Measure
□	Product	✗	Version, Year, Currency, Region, Product, Product Measure
□	Retail	✗	Version, Year, Period, Currency, Region, Product, Retail Measure
□	System Info	✓	System Info Parameter, System Info Measures
□	System User Settings	✓	}Clients, System User Settings Parameter, System User Settings Measure

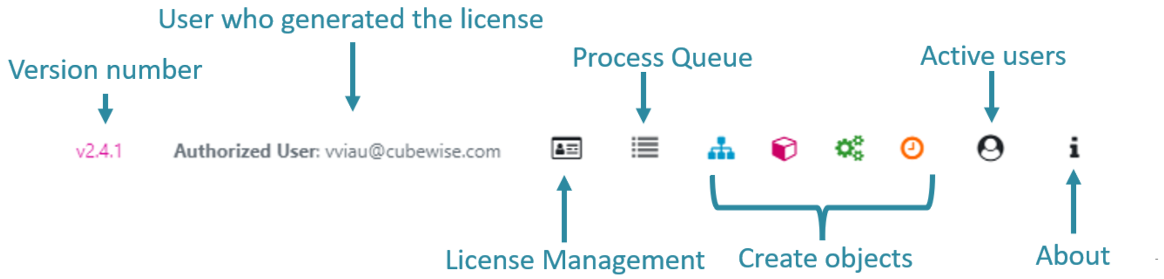
In the tab header, you can click the star to add to favourites or click the lock icon to lock it (if you refresh the browser, it is going to keep the tab):



The new tab will open on the workbench and therefore hide the homepage. To show the homepage again, you will have to close all tabs.



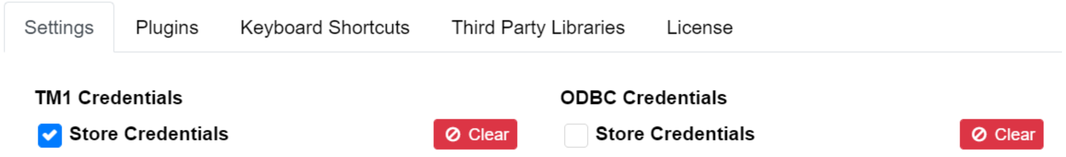
Arc top bar



In the top bar you will find:

- Arc version number
- Authorized User: The person who generated the license
- License Management: To request new licenses
- Process Queue: List of running processes
- Create objects: A shortcut to create dimension, cube, process and chore
- Active users: See who is active in Arc, Logout to a TM1 instance
- About: To open the About page.

After your first login to TM1, in the Arc About page, you can tick the option to store your TM1 credentials so you won't have to input again your TM1 password:



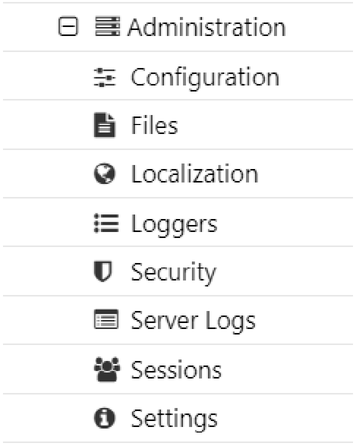
TM1 Credentials

☑ Store Credentials ⊘ Clear

ODBC Credentials

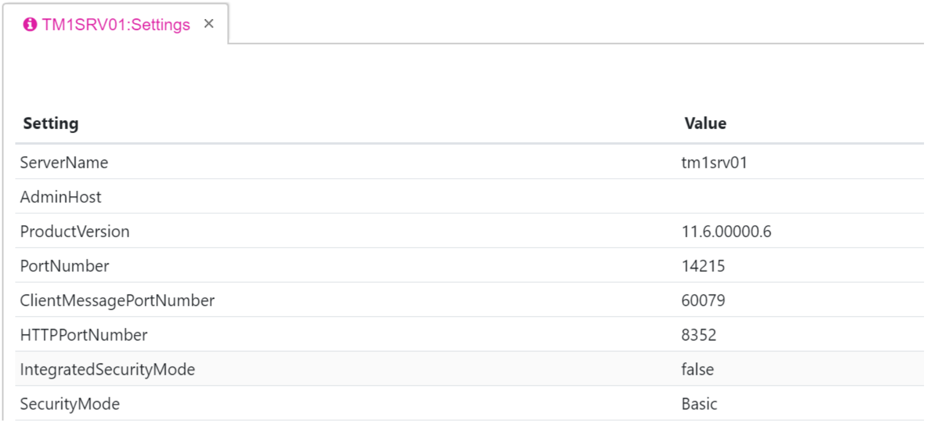
☐ Store Credentials ⊘ Clear

Administration



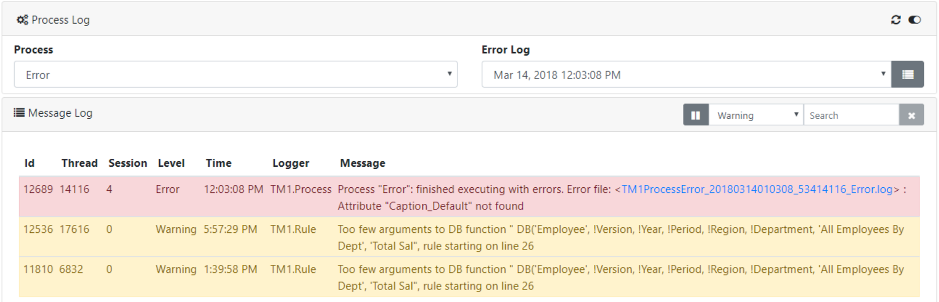
Arc comes with a set of common administration tools to allow you to access the message log, process error logs and active user sessions:

Settings:



Setting	Value
ServerName	tm1srv01
AdminHost	
ProductVersion	11.6.00000.6
PortNumber	14215
ClientMessagePortNumber	60079
HTTPPortNumber	8352
IntegratedSecurityMode	false
SecurityMode	Basic

Server Logs:

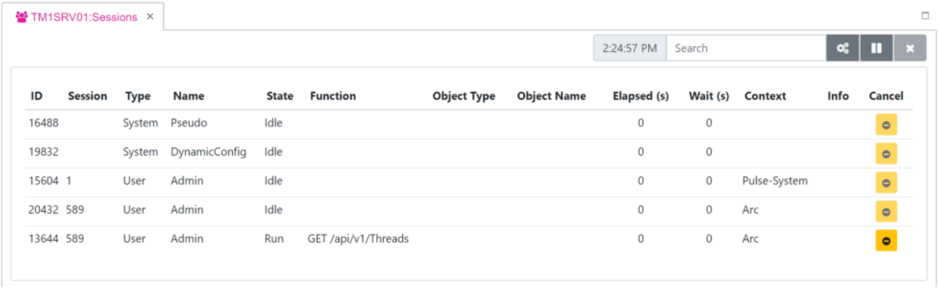


Process: Error

Error Log: Mar 14, 2018 12:03:08 PM

Id	Thread	Session	Level	Time	Logger	Message
12689	14116	4	Error	12:03:08 PM	TM1.Process	Process "Error": finished executing with errors. Error file: <TM1ProcessError_20180314010308_53414116_Error.log> : Attribute "Caption_Default" not found
12536	17616	0	Warning	5:57:29 PM	TM1.Rule	Too few arguments to DB function " DB('Employee', !Version, !Year, !Period, !Region, !Department, 'All Employees By Dept', 'Total Sal", rule starting on line 26
11810	6832	0	Warning	1:39:58 PM	TM1.Rule	Too few arguments to DB function " DB('Employee', !Version, !Year, !Period, !Region, !Department, 'All Employees By Dept', 'Total Sal", rule starting on line 26

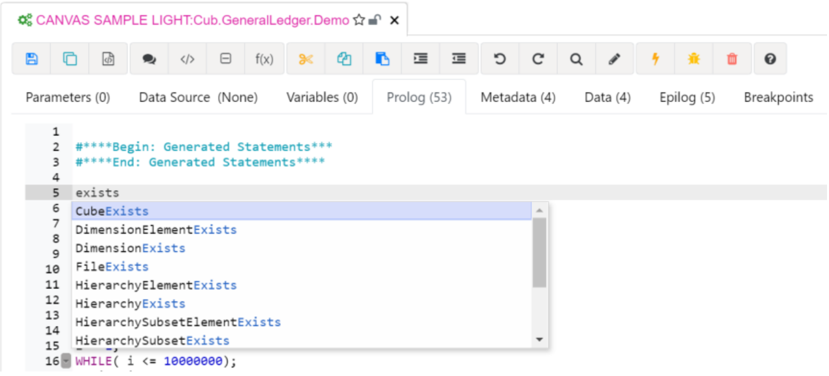
Threads:



ID	Session	Type	Name	State	Function	Object Type	Object Name	Elapsed (s)	Wait (s)	Context	Info	Cancel
16488		System	Pseudo	Idle				0	0			⊖
19832		System	DynamicConfig	Idle				0	0			⊖
15604	1	User	Admin	Idle				0	0	Pulse-System		⊖
20432	589	User	Admin	Idle				0	0	Arc		⊖
13644	589	User	Admin	Run	GET /api/v1/Threads			0	0	Arc		⊖

Snippets

Arc delivers modern IDE features such as snippets, hit **CTRL+SPACE** in a TM1 process or TM1 rule and the snippets will appear:



```

#****Begin: Generated Statements***
#****End: Generated Statements****

exists
CubeExists
DimensionElementExists
DimensionExists
FileExists
HierarchyElementExists
HierarchyExists
HierarchySubsetElementExists
HierarchySubsetExists
WHILE( i <= 10000000);
```

More information about all features can be found on our website:

- Arc for TM1 features

TOPIC 2: WORKING WITH HIERARCHIES

Objectives

In this section, students will learn:

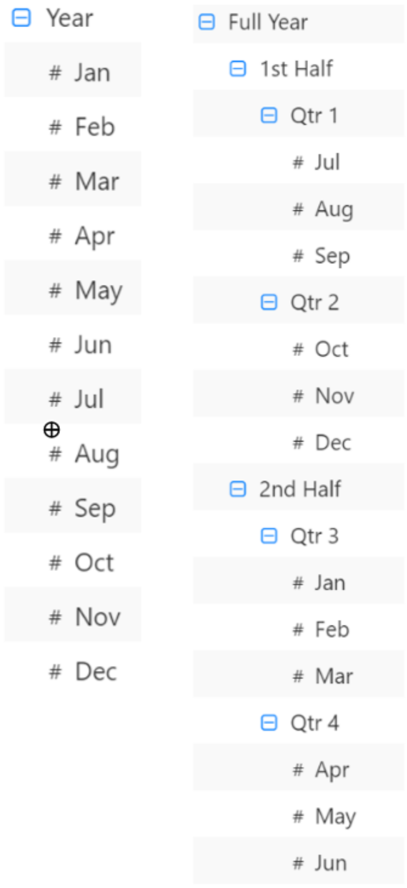
- The basics about hierarchies.
- How to build a new hierarchy manually.
- How to build a new hierarchy using new TM1 process.
- Introduction to Bedrock 4

Introduction to Hierarchies

Hierarchies vs Rollups

Let's begin by answering this question: **"What is a Hierarchy?"**

Until the arrival of Planning Analytics, what was commonly called a "hierarchy" in TM1 was simply a specific roll-up of C and N level elements in a dimension. For example, in a "Period" dimension, the same N-level elements could roll up to multiple C-level elements, with the roll-ups having names like "Full Year" and "Jun YTD":



Cube Architecture with Hierarchies

TM1's basic cube structure has not changed since TM1's invention in 1984: a cube is made of two or more dimensions, a cell's value is attached to the N-level elements in those dimensions, and each dimension can have multiple consolidation paths that roll up N and C elements. In other words, there was a direct relationship between a dimension and its elements.

Planning Analytics has introduced "real" hierarchies to TM1. Instead of a straight path from a dimension to its elements, there is now the option of inserting an intermediate level. This container object is called a "Hierarchy", and a dimension can have as many hierarchies as you wish.

By default, the new Hierarchy feature is not turned on – it must be enabled by adding the line **EnableNewHierarchyCreation=T** in the **tm1s.cfg** file. When you turn on Hierarchies:

An extra level can be added between dimensions and elements in TM1's object model

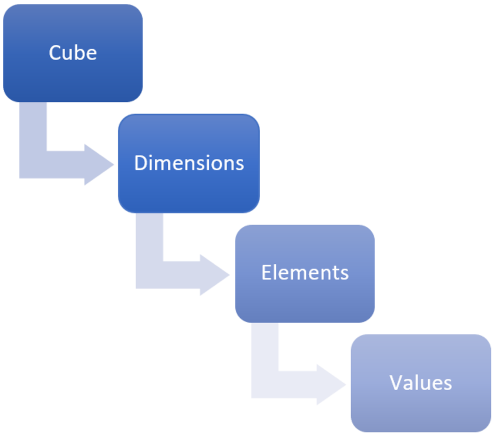
Dimensions are no longer a container of elements, they are a container of Hierarchies.

A default Hierarchy is created, which has the exact same name as the dimension itself (this is to maintain backward compatibility)

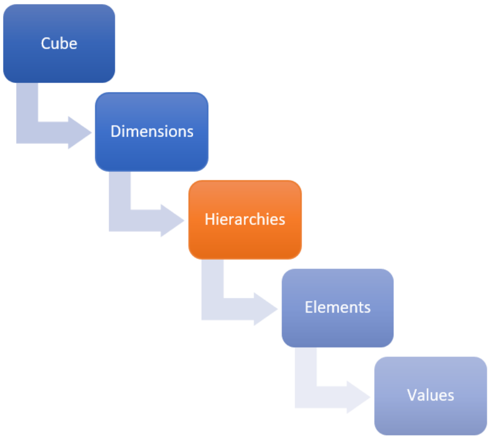
Each dimension can now have multiple Hierarchies, with each Hierarchy containing its own set of consolidations and can include one or more of the leaf elements that are shared across Hierarchies.

The "before" and "after" of TM1's object model looks like this:

No hierarchies



Hierarchies



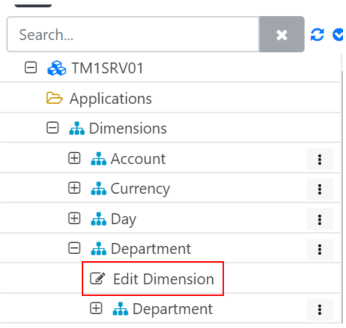
To learn more about Hierarchies, you can check the **Mastering Hierarchies** blog article from the Cubewise CODE website:

- code.cubewise.com/blog/mastering-hierarchies-in-ibm-tm1-and-planning-analytics

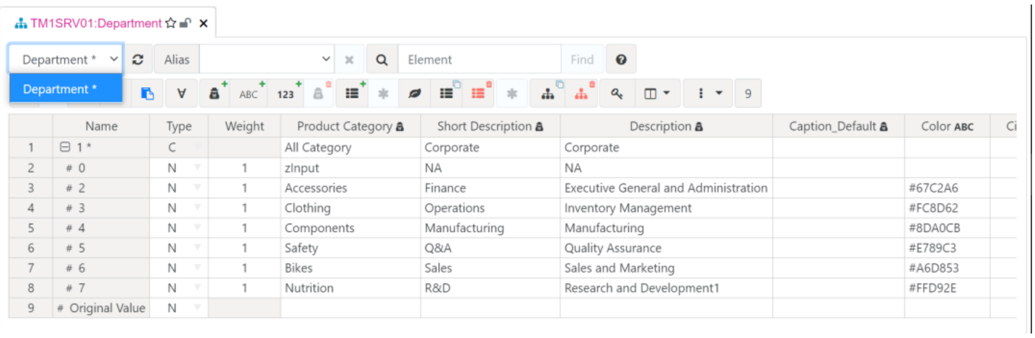
Build hierarchies manually

Let's start by building manually a new hierarchy, we want to create a new hierarchy called **Type** which groups **Bikes**, **Components** and **Clothing** under a new consolidation **High Value** and then **Accessories**, **Safety** and **Nutrition** under **Low Value**.

First, expand the **Dimensions** from the menu, expand the **Department** and finally click on the **Edit** button:

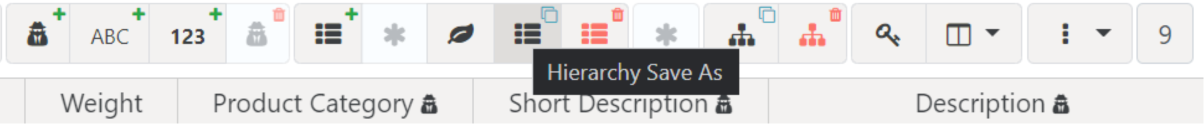


By clicking on the dimension (Edit Dimension) in the left-hand side menu, it is going to open the **hierarchy editor**. The dimension has currently only one hierarchy (the default hierarchy which has the same name as the dimension name):

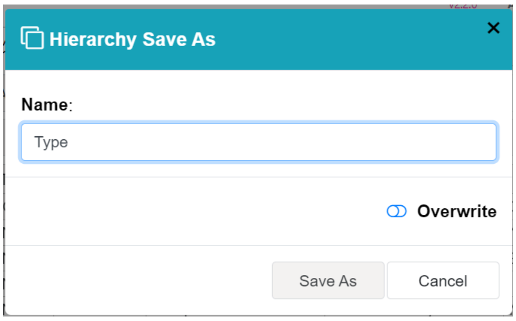


	Name	Type	Weight	Product Category	Short Description	Description	Caption_Default	Color ABC	Ci
1	⊟ 1 *	C		All Category	Corporate	Corporate			
2	# 0	N	1	zInput	NA	NA			
3	# 2	N	1	Accessories	Finance	Executive General and Administration		#67C2A6	
4	# 3	N	1	Clothing	Operations	Inventory Management		#FC8D62	
5	# 4	N	1	Components	Manufacturing	Manufacturing		#8DA0CB	
6	# 5	N	1	Safety	Q&A	Quality Assurance		#E789C3	
7	# 6	N	1	Bikes	Sales	Sales and Marketing		#A6D853	
8	# 7	N	1	Nutrition	R&D	Research and Development1		#FFD92E	
9	# Original Value	N							

To create a new hierarchy manually, you can either click "**New Hierarchy**" to start from a blank hierarchy or like in our case we want to create a new hierarchy including all leaf elements so we are going to click on, "**Hierarchy Save As**" :

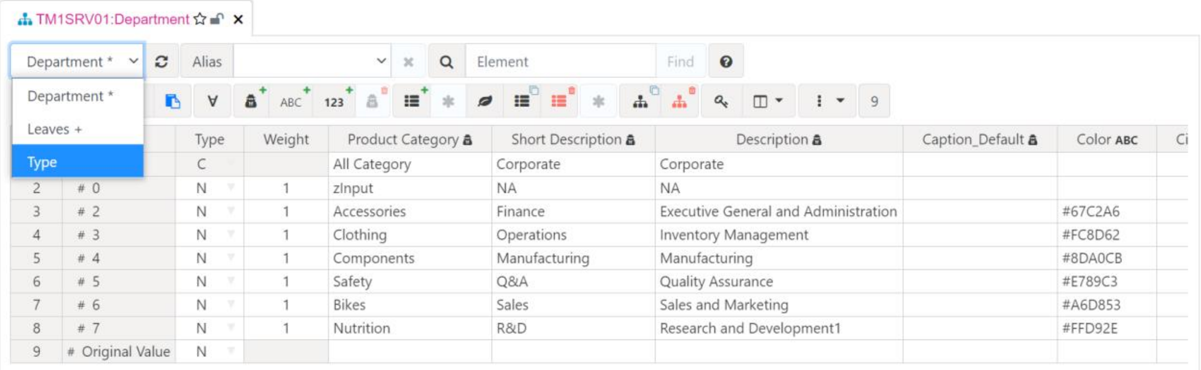


The name of the new hierarchy is **Type** and then click **Save As** button:



Name:
Type

Now in the hierarchy list, we can choose between **Default**, **Leaves** and our new hierarchy, **Type**:



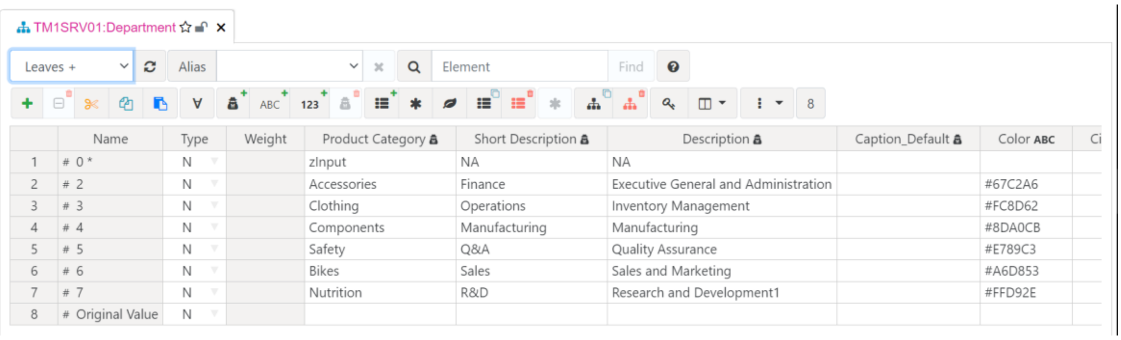
TM1SRV01:Department

Department * | Alias | Element | Find

Department *
Leaves +
Type

		Type	Weight	Product Category	Short Description	Description	Caption_Default	Color ABC	Ci
		C		All Category	Corporate	Corporate			
2	# 0	N	1	zInput	NA	NA			
3	# 2	N	1	Accessories	Finance	Executive General and Administration		#67C2A6	
4	# 3	N	1	Clothing	Operations	Inventory Management		#FC8D62	
5	# 4	N	1	Components	Manufacturing	Manufacturing		#8DA0CB	
6	# 5	N	1	Safety	Q&A	Quality Assurance		#E789C3	
7	# 6	N	1	Bikes	Sales	Sales and Marketing		#A6D853	
8	# 7	N	1	Nutrition	R&D	Research and Development1		#FFD92E	
9	# Original Value	N							

The **Leaves** hierarchy is created automatically by TM1 when the first hierarchy is created. It is a default hierarchy which contains all leaf elements:

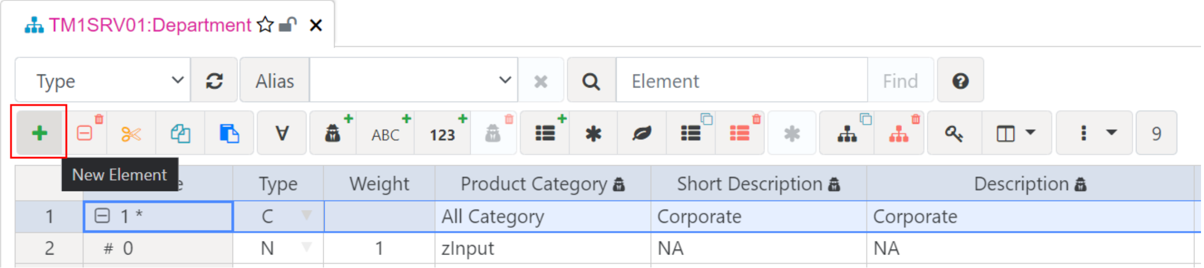


TM1SRV01:Department

Leaves + | Alias | Element | Find

	Name	Type	Weight	Product Category	Short Description	Description	Caption_Default	Color ABC	Ci
1	# 0 *	N		zInput	NA	NA			
2	# 2	N		Accessories	Finance	Executive General and Administration		#67C2A6	
3	# 3	N		Clothing	Operations	Inventory Management		#FC8D62	
4	# 4	N		Components	Manufacturing	Manufacturing		#8DA0CB	
5	# 5	N		Safety	Q&A	Quality Assurance		#E789C3	
6	# 6	N		Bikes	Sales	Sales and Marketing		#A6D853	
7	# 7	N		Nutrition	R&D	Research and Development1		#FFD92E	
8	# Original Value	N							

Let's go back to the **Type** hierarchy, we want to add two new consolidations to the top consolidation "**1**", select the top consolidation and then click the **+** button to add our new consolidations:

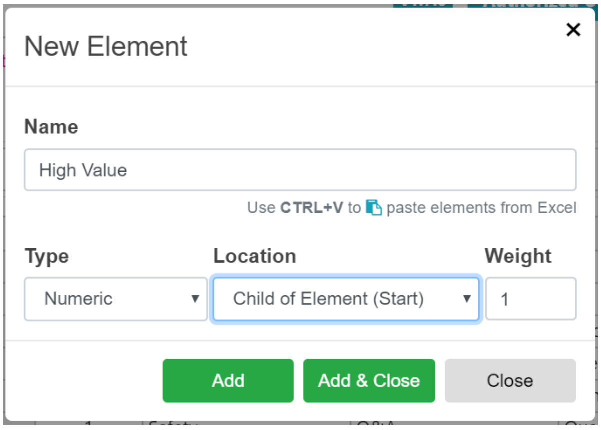


TM1SRV01:Department

Type | Alias | Element | Find

	e	Type	Weight	Product Category	Short Description	Description
1	⊟ 1 *	C		All Category	Corporate	Corporate
2	# 0	N	1	zInput	NA	NA

Enter 'High Value' as the **name** for the new consolidation, set the **type** to 'Consolidated' and select 'Child of Element (Start)' to add as a first child of the consolidation:



New Element

Name: High Value

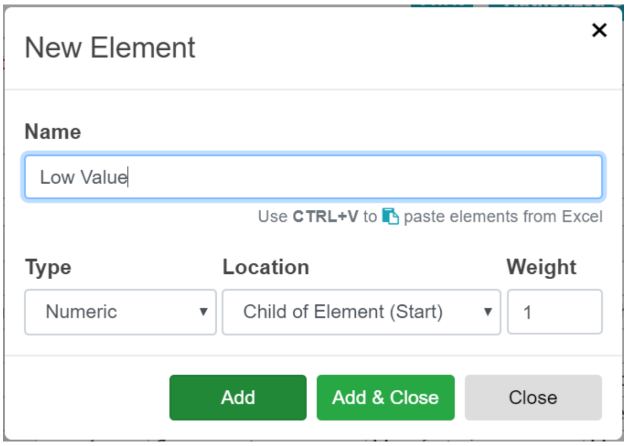
Use CTRL+V to paste elements from Excel

Type: Numeric

Location: Child of Element (Start)

Weight: 1

Click **Add** button and then **Add** the **Low Value** Consolidation, click the **Add & Close** button to close the pop-up window:



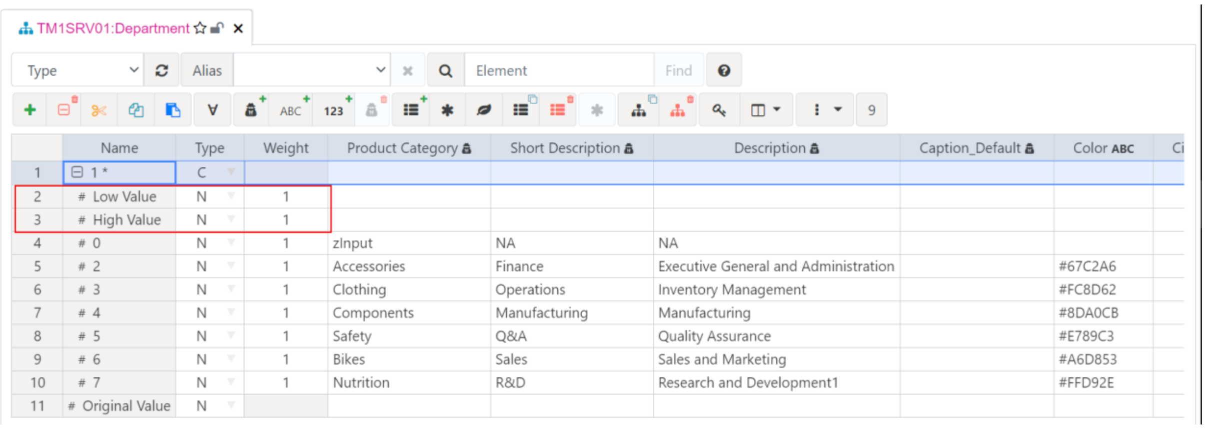
Name: Low Value

Type: Numeric

Location: Child of Element (Start)

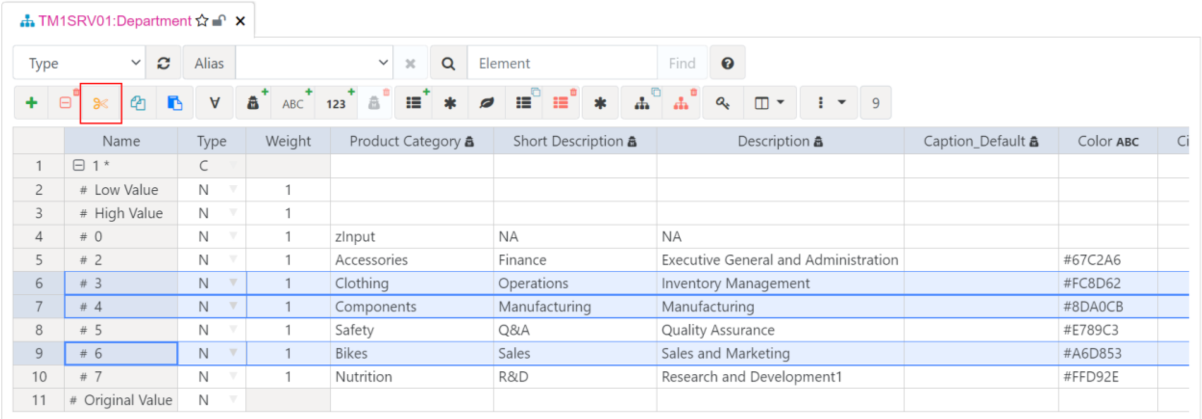
Weight: 1

Changes are immediately applied to TM1. If you look at the Type Hierarchy, you will see the two new elements:



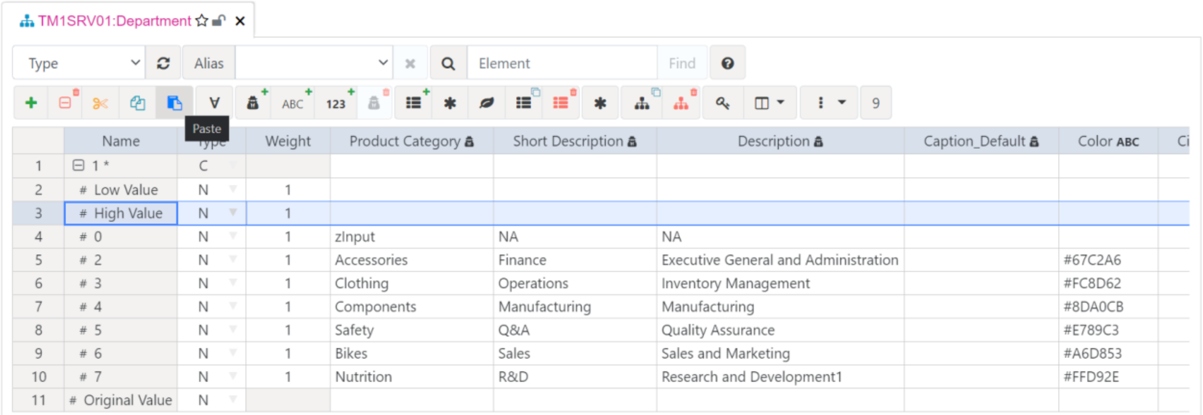
	Name	Type	Weight	Product Category	Short Description	Description	Caption_Default	Color ABC	Ci
1	⊟ 1 *	C							
2	# Low Value	N	1						
3	# High Value	N	1						
4	# 0	N	1	zInput	NA	NA			
5	# 2	N	1	Accessories	Finance	Executive General and Administration		#67C2A6	
6	# 3	N	1	Clothing	Operations	Inventory Management		#FC8D62	
7	# 4	N	1	Components	Manufacturing	Manufacturing		#8DA0CB	
8	# 5	N	1	Safety	Q&A	Quality Assurance		#E789C3	
9	# 6	N	1	Bikes	Sales	Sales and Marketing		#A6D853	
10	# 7	N	1	Nutrition	R&D	Research and Development1		#FFD92E	
11	# Original Value	N							

We want to move the elements 3,4 and 6 as child of **High Value**. To do that, first select the three elements as below and then click on the cut button:



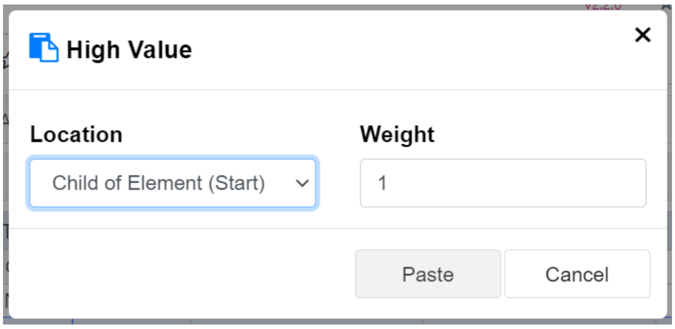
	Name	Type	Weight	Product Category	Short Description	Description	Caption_Default	Color ABC	Ci
1	⊟ 1 *	C							
2	# Low Value	N	1						
3	# High Value	N	1						
4	# 0	N	1	zInput	NA	NA			
5	# 2	N	1	Accessories	Finance	Executive General and Administration		#67C2A6	
6	# 3	N	1	Clothing	Operations	Inventory Management		#FC8D62	
7	# 4	N	1	Components	Manufacturing	Manufacturing		#8DA0CB	
8	# 5	N	1	Safety	Q&A	Quality Assurance		#E789C3	
9	# 6	N	1	Bikes	Sales	Sales and Marketing		#A6D853	
10	# 7	N	1	Nutrition	R&D	Research and Development1		#FFD92E	
11	# Original Value	N							

Once the cut button selected, you will be able to see the paste button, just click the paste button for the element **High Value** as below:



	Name	Type	Weight	Product Category	Short Description	Description	Caption_Default	Color ABC	Ci
1	⊟ 1 *	C							
2	# Low Value	N	1						
3	# High Value	N	1						
4	# 0	N	1	zInput	NA	NA			
5	# 2	N	1	Accessories	Finance	Executive General and Administration		#67C2A6	
6	# 3	N	1	Clothing	Operations	Inventory Management		#FC8D62	
7	# 4	N	1	Components	Manufacturing	Manufacturing		#8DA0CB	
8	# 5	N	1	Safety	Q&A	Quality Assurance		#E789C3	
9	# 6	N	1	Bikes	Sales	Sales and Marketing		#A6D853	
10	# 7	N	1	Nutrition	R&D	Research and Development1		#FFD92E	
11	# Original Value	N							

We want to add the elements as child so select **Child of Element (Start)** and then press **Paste**:

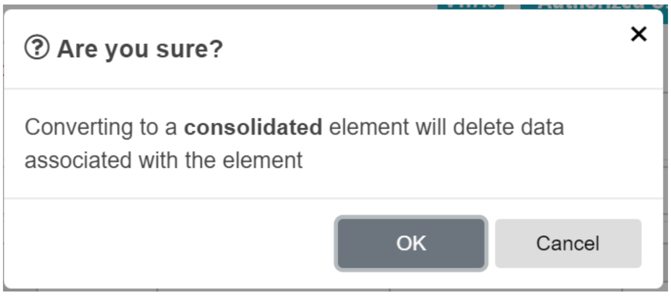


High Value

Location: Child of Element (Start)

Weight: 1

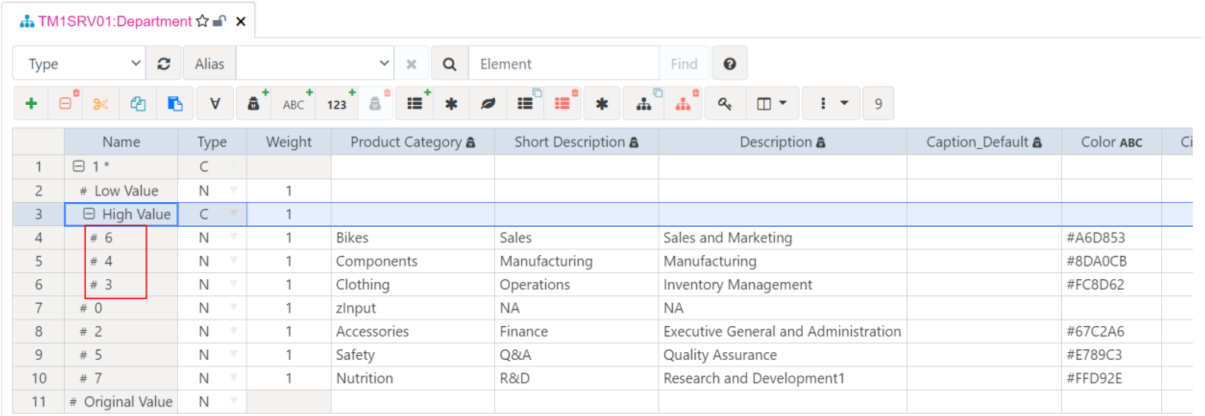
If the **High Value** was a leaf element you will the warning message below saying that the element will become a consolidation, just click **OK**:



Are you sure?

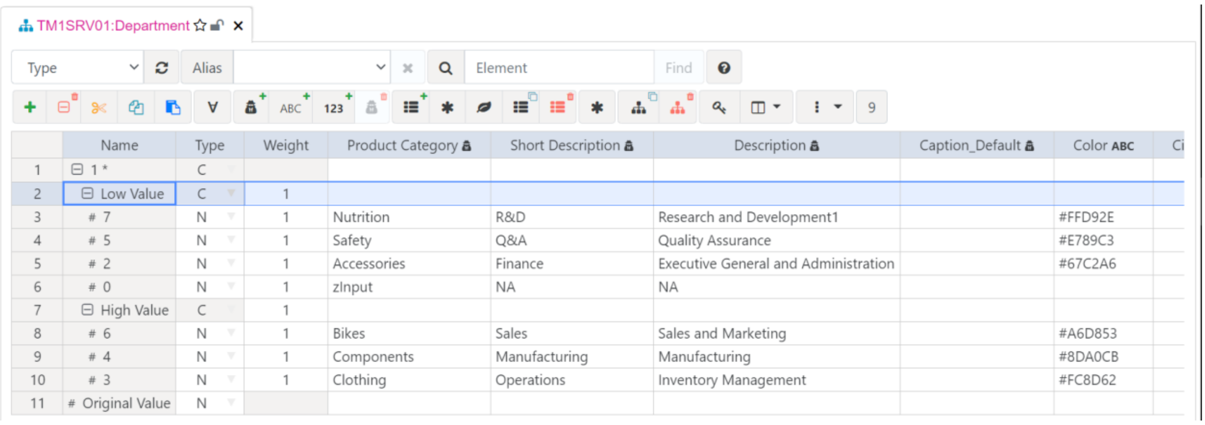
Converting to a **consolidated** element will delete data associated with the element

You should be able to see now



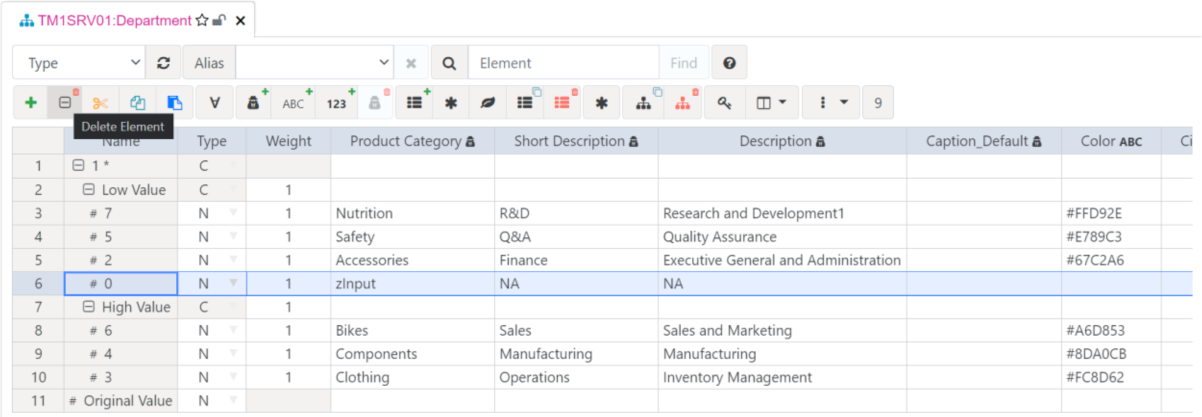
	Name	Type	Weight	Product Category	Short Description	Description	Caption_Default	Color ABC	Ci
1	⊟ 1 *	C							
2	# Low Value	N	1						
3	⊟ High Value	C	1						
4	# 6	N	1	Bikes	Sales	Sales and Marketing		#A6D853	
5	# 4	N	1	Components	Manufacturing	Manufacturing		#8DA0CB	
6	# 3	N	1	Clothing	Operations	Inventory Management		#FC8D62	
7	# 0	N	1	zInput	NA	NA			
8	# 2	N	1	Accessories	Finance	Executive General and Administration		#67C2A6	
9	# 5	N	1	Safety	Q&A	Quality Assurance		#E789C3	
10	# 7	N	1	Nutrition	R&D	Research and Development1		#FFD92E	
11	# Original Value	N							

Let's replicate these steps to move elements 0,2,5 and 7 as child of **Low Value**:



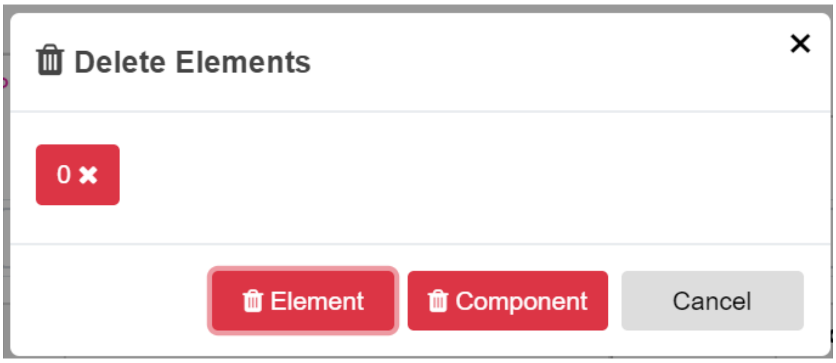
	Name	Type	Weight	Product Category	Short Description	Description	Caption_Default	Color ABC	Ci
1	⊟ 1 *	C							
2	⊟ Low Value	C	1						
3	# 7	N	1	Nutrition	R&D	Research and Development1		#FFD92E	
4	# 5	N	1	Safety	Q&A	Quality Assurance		#E789C3	
5	# 2	N	1	Accessories	Finance	Executive General and Administration		#67C2A6	
6	# 0	N	1	zInput	NA	NA			
7	⊟ High Value	C	1						
8	# 6	N	1	Bikes	Sales	Sales and Marketing		#A6D853	
9	# 4	N	1	Components	Manufacturing	Manufacturing		#8DA0CB	
10	# 3	N	1	Clothing	Operations	Inventory Management		#FC8D62	
11	# Original Value	N							

In this hierarchy we do not need the **0** element, click the delete button and you will then have to click the Confirm button to delete it:



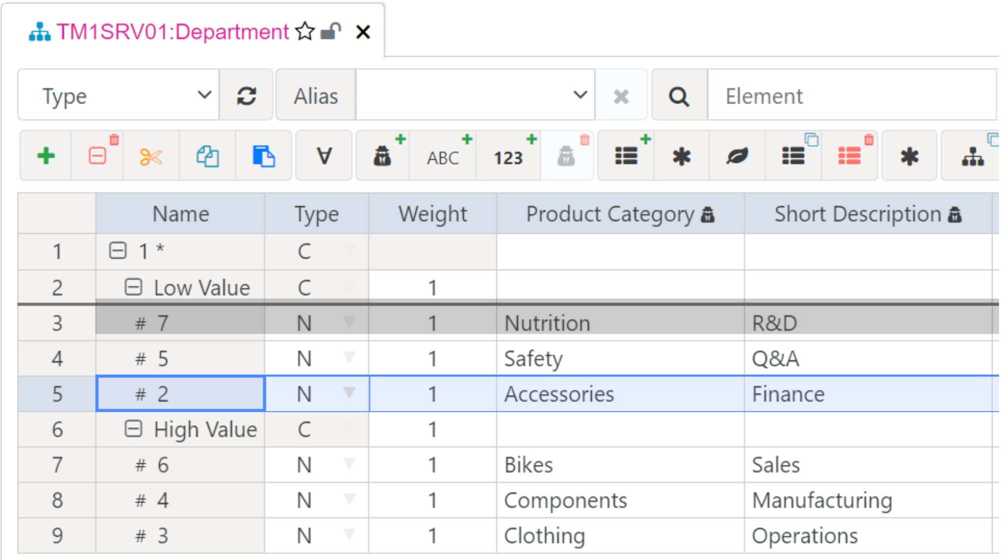
	Name	Type	Weight	Product Category	Short Description	Description	Caption_Default	Color ABC	Ci
1	⊟ 1 *	C							
2	⊟ Low Value	C	1						
3	# 7	N	1	Nutrition	R&D	Research and Development1		#FFD92E	
4	# 5	N	1	Safety	Q&A	Quality Assurance		#E789C3	
5	# 2	N	1	Accessories	Finance	Executive General and Administration		#67C2A6	
6	# 0	N	1	zInput	NA	NA			
7	⊟ High Value	C	1						
8	# 6	N	1	Bikes	Sales	Sales and Marketing		#A6D853	
9	# 4	N	1	Components	Manufacturing	Manufacturing		#8DA0CB	
10	# 3	N	1	Clothing	Operations	Inventory Management		#FC8D62	
11	# Original Value	N							

Click **Element** to delete the element from the Hierarchy:



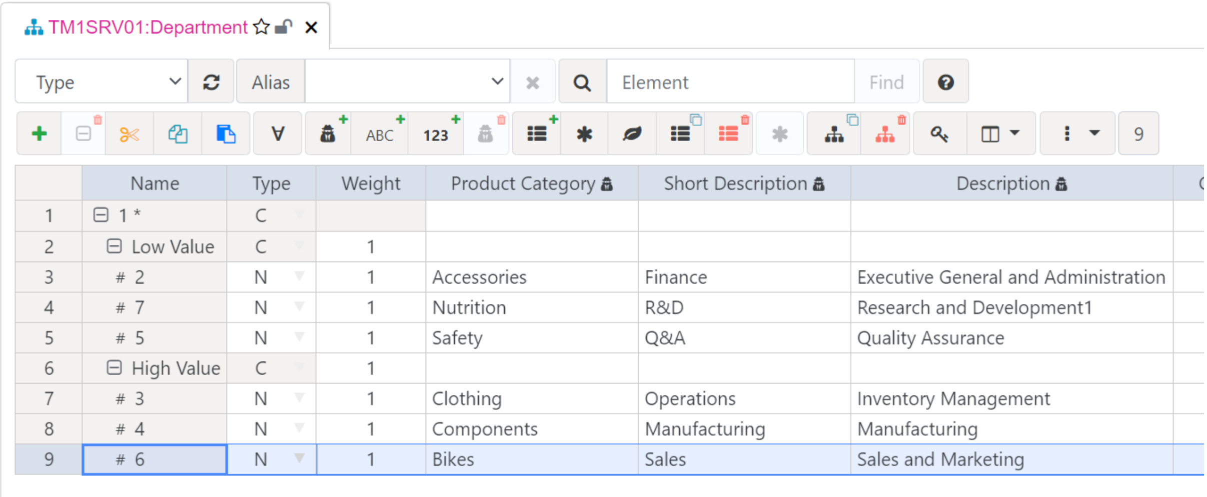
Note: The delete **Component** button would remove the element only from the consolidation. We don't need the **Original Value** element, just delete it as well.

You can then use drag and drop to reorder the child, for example click on the element 2 row number and drag it before the element 7.



	Name	Type	Weight	Product Category	Short Description
1	⊟ 1 *	C			
2	⊟ Low Value	C	1		
3	# 7	N	1	Nutrition	R&D
4	# 5	N	1	Safety	Q&A
5	# 2	N	1	Accessories	Finance
6	⊟ High Value	C	1		
7	# 6	N	1	Bikes	Sales
8	# 4	N	1	Components	Manufacturing
9	# 3	N	1	Clothing	Operations

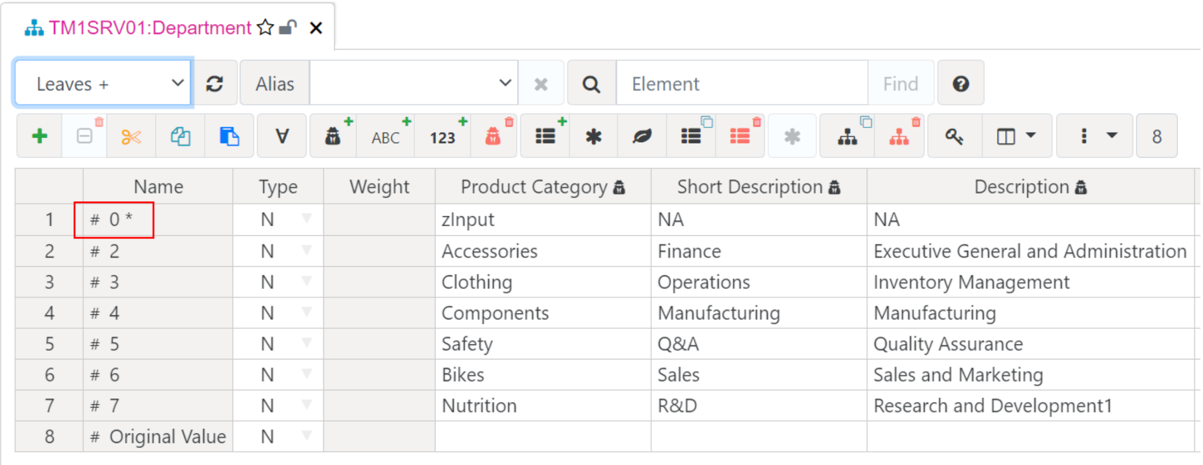
Replicate these steps to order all elements as below, the **Type** hierarchy should look like this:



	Name	Type	Weight	Product Category	Short Description	Description
1	⊟ 1 *	C				
2	⊟ Low Value	C	1			
3	# 2	N	1	Accessories	Finance	Executive General and Administration
4	# 7	N	1	Nutrition	R&D	Research and Development1
5	# 5	N	1	Safety	Q&A	Quality Assurance
6	⊟ High Value	C	1			
7	# 3	N	1	Clothing	Operations	Inventory Management
8	# 4	N	1	Components	Manufacturing	Manufacturing
9	# 6	N	1	Bikes	Sales	Sales and Marketing

Congratulations you built your first Hierarchy manually!

It is important to be aware that deleting a leaf element from a hierarchy is not going to delete it from the default hierarchy (Department and Leaves), you can see now that these two elements have been removed from the **Type** hierarchy but they are still available in the **Department** and **Leaves** Hierarchy:



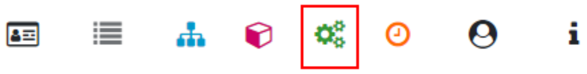
	Name	Type	Weight	Product Category	Short Description	Description
1	# 0 *	N		zInput	NA	NA
2	# 2	N		Accessories	Finance	Executive General and Administration
3	# 3	N		Clothing	Operations	Inventory Management
4	# 4	N		Components	Manufacturing	Manufacturing
5	# 5	N		Safety	Q&A	Quality Assurance
6	# 6	N		Bikes	Sales	Sales and Marketing
7	# 7	N		Nutrition	R&D	Research and Development1
8	# Original Value	N				

Build hierarchies with TM1 process

Objectives

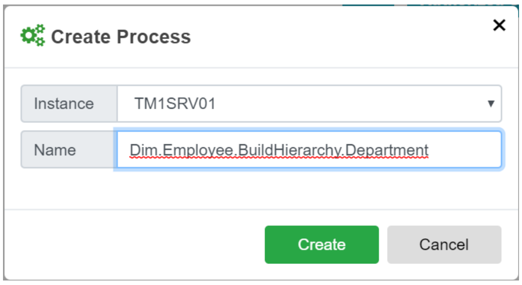
Each employee in the **Employee** dimension has a **Department Code** attribute value. We want to create a new hierarchy called **Actual 2018 Department** which contains all employees with their current department value. To do this we are going to create a new process.

To create new process, just click on the **New Process** button on the top right corner:



Then type the new process name:

- **Dim.Employee.BuildHierarchy.Department**



Create Process

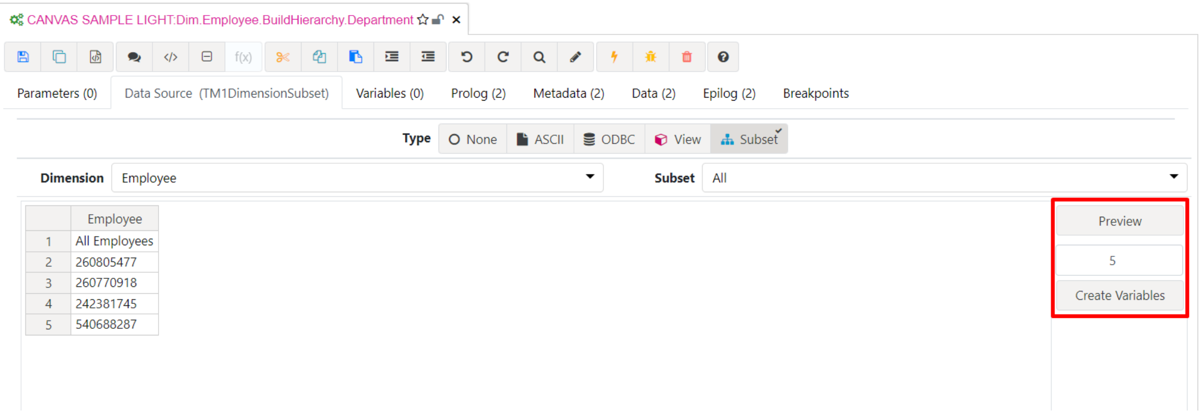
Instance: TM1SRV01

Name: Dim.Employee.BuildHierarchy.Department

Click **Create** button, you should now see a new tab with the process name. In the **Data Source** tab, select:

- Type: **Dimension Subset**
- Dimension: **Employee**
- Subset: **All**

Click **Preview** and then **Create Variables**.



CANVAS SAMPLE LIGHT:Dim.Employee.BuildHierarchy.Department

Type: Subset

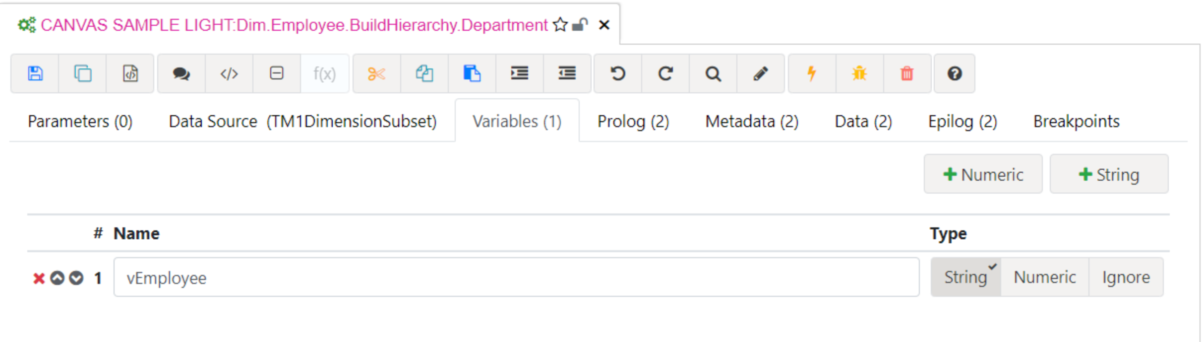
Dimension: Employee

Subset: All

5

	Employee
1	All Employees
2	260805477
3	260770918
4	242381745
5	540688287

In the **variables** tab, notice that Arc has already prefixed the variable with a "**v**" (this can be configured in the **settings.yml**):

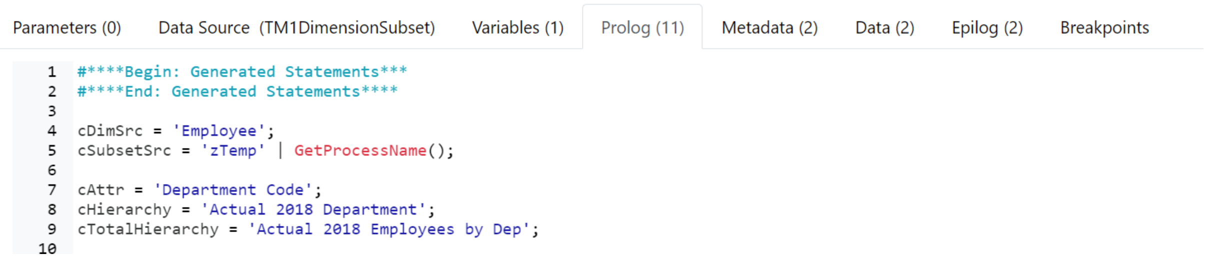


#	Name	Type
1	vEmployee	String ✓ Numeric Ignore

Let's start by defining the constants in the **Prolog** tab, to build the hierarchy:

- **cDimSrc**: our dimension source which is also our target dimension.
- **cSubsetSrc**: The subset source that we are going to loop through.
- **cAttr**: The attribute we need to build the consolidation.
- **cHierarchy**: Hierarchy name.
- **cTotalHierarchy**: The top consolidation of our new hierarchy.

Enter the code in the screenshot then click the save button or hit CTRL+S:



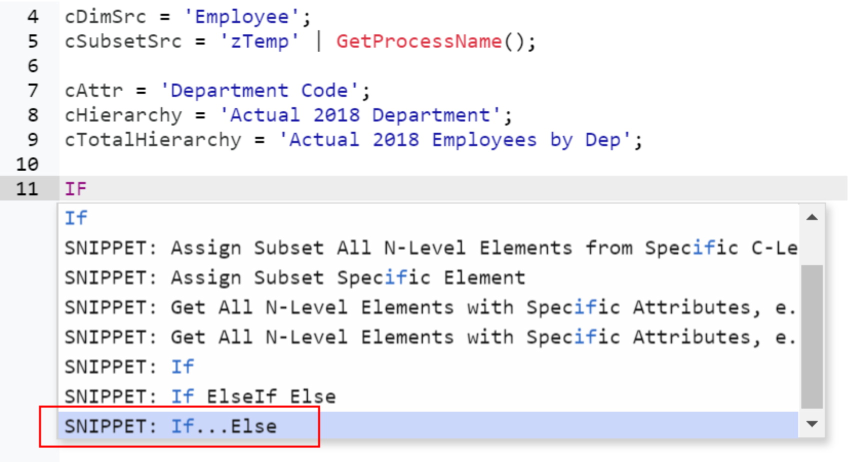
```
#****Begin: Generated Statements***
#****End: Generated Statements****

cDimSrc = 'Employee';
cSubsetSrc = 'zTemp' | GetProcessName();

cAttr = 'Department Code';
cHierarchy = 'Actual 2018 Department';
cTotalHierarchy = 'Actual 2018 Employees by Dep';

```

Hierarchies with processes

To work with hierarchies in the Turbo Integrator process (TI), a lot of new functions were introduced in IBM Planning Analytics. Thankfully all new functions to work with hierarchies are very similar to the dimension TI functions. For hierarchies, you will almost find a one to one match with the dimension functions such as **HierarchyExists** instead of **DimensionExists. HierarchyExists is used** to check if a hierarchy exists.

Arc facilitates this transition through snippets. To open snippets, press **CTRL + SPACE. Use this command in the Prolog after the above code** and then start typing "exists" and then you will find the function we need.

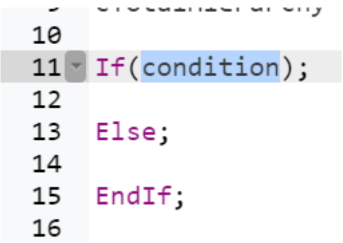
We want to check if the **Hierarchy exists**, we need first a **IF ELSE** condition. To add a IF ELSE, open the snippets with CTRL + SPACE and stat typing IF



```
cDimSrc = 'Employee';
cSubsetSrc = 'zTemp' | GetProcessName();

cAttr = 'Department Code';
cHierarchy = 'Actual 2018 Department';
cTotalHierarchy = 'Actual 2018 Employees by Dep';

IF
```

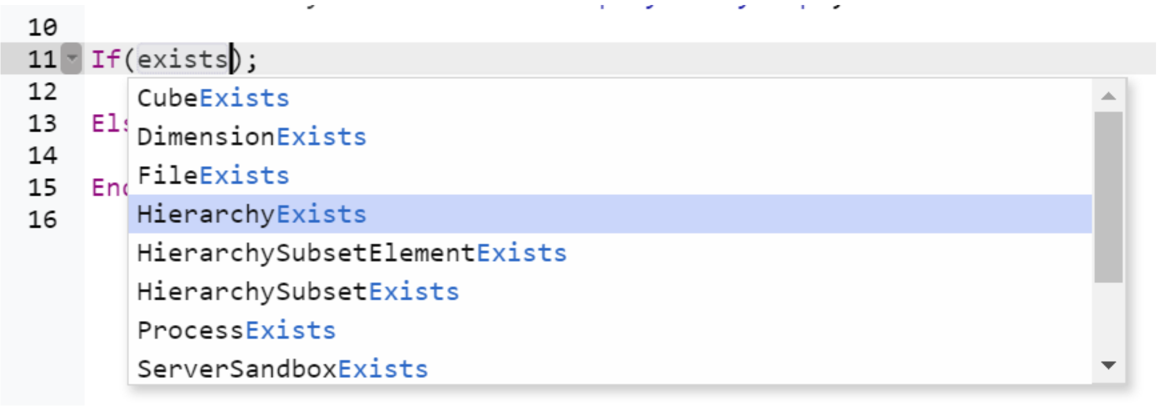
If
SNIPPET: Assign Subset All N-Level Elements from Specific C-Le
SNIPPET: Assign Subset Specific Element
SNIPPET: Get All N-Level Elements with Specific Attributes, e.
SNIPPET: Get All N-Level Elements with Specific Attributes, e.
SNIPPET: If
SNIPPET: If ElseIf Else
SNIPPET: If...Else

By clicking on **SNIPPET: If…Else**, Arc is going to prepopulate the code for you:



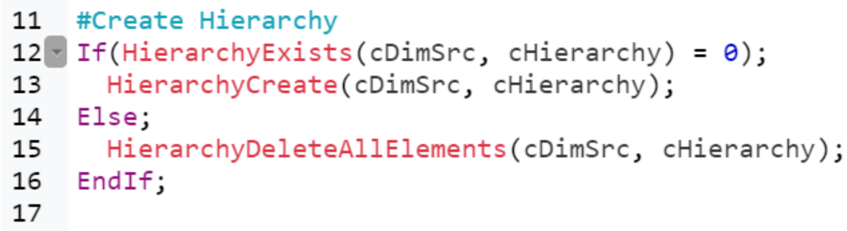
```

If(condition);

Else;

EndIf;

```

Now we can replace the **condition** with the **HierarchyExists**, open the snippet and search for the **exists** text:



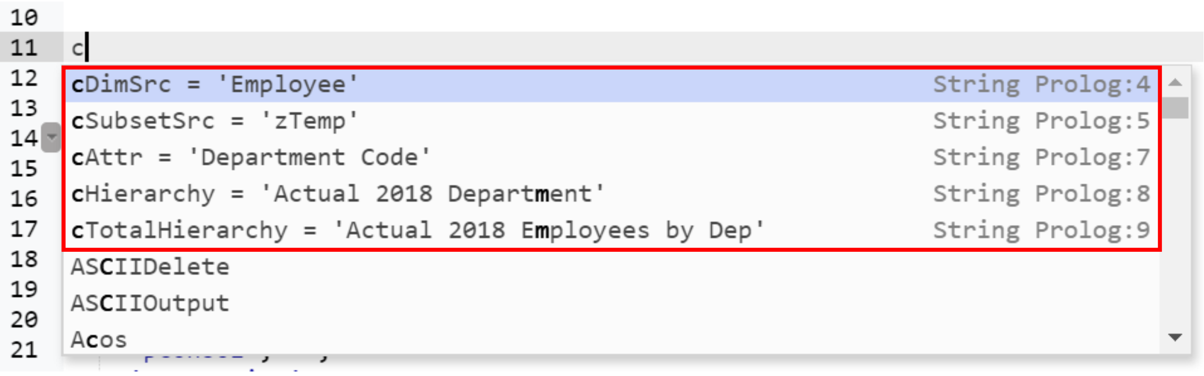
```

If(exists);

Else;

EndIf;

```

To create a dimension or a hierarchy, we need to check first if the hierarchy exists, if it exists we delete all elements if it does not exist we create the new hierarchy. Copy the code.



```
#Create Hierarchy
If(HierarchyExists(cDimSrc, cHierarchy) = 0);
  HierarchyCreate(cDimSrc, cHierarchy);
Else;
  HierarchyDeleteAllElements(cDimSrc, cHierarchy);
EndIf;

```

In the snippets, all IBM TM1 and Planning Analytics functions plus all the variables defined in your process. For example, if you click **CTRL+SPAC**E and then ‘**c**’, you will see all variables starting with a ‘**c**’.

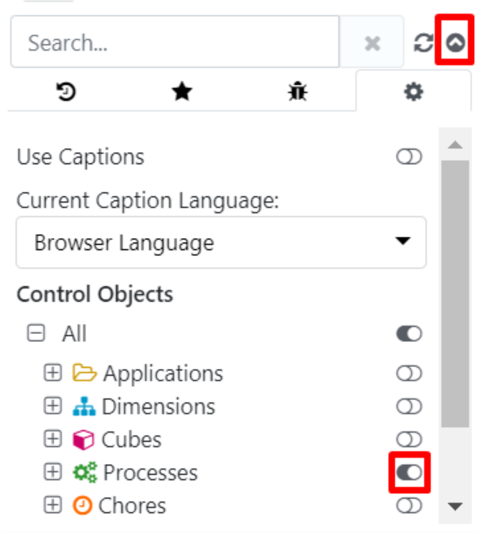


```
c
cDimSrc = 'Employee'                                              String Prolog:4
cSubsetSrc = 'zTemp'                                              String Prolog:5
cAttr = 'Department Code'                                         String Prolog:7
cHierarchy = 'Actual 2018 Department'                             String Prolog:8
cTotalHierarchy = 'Actual 2018 Employees by Dep'                  String Prolog:9
ASCIIDelete
ASCIIOutput
Acos
```

Using snippets will significantly reduce your typing and therefore speed up your development.

It can be noted that instead of deleting all elements in a hierarchy using **HierarchyDeleteAllElements**, you should **unwind** the hierarchy to avoid the risk of losing data.

To unwind a Hierarchy, the Bedrock 4 process **}bedrock.hier.unwind** can be used. As all Bedrock 4 processes are control objects, to find them from the snippets, you will have first to check if the **Show/Hide Control Objects** is turned on. This setting is available under the search (Click the arrow icon to show/hide this setting):



Search...

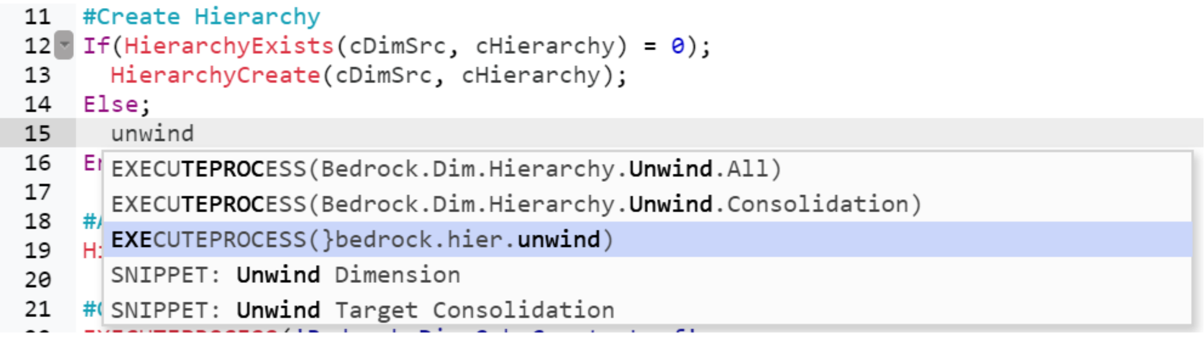
Use Captions ○

Current Caption Language: Browser Language

Control Objects

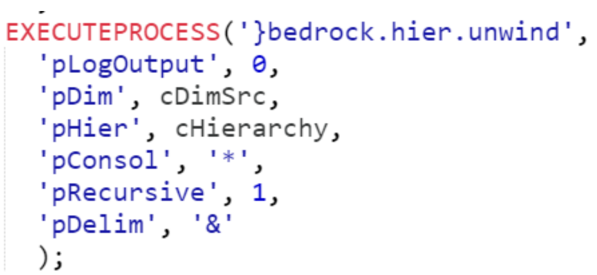
- All ●
  - Applications ○
  - Dimensions ○
  - Cubes ○
  - Processes ●
  - Chores ○

Press CTRL+SPACE in the **Prolog** tab and type **unwind** and select **EXECUTEPROCESS(}bedrock.hier.unwind)**:



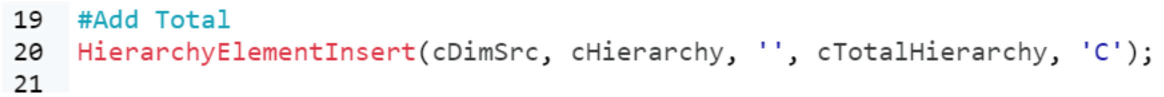
```
#Create Hierarchy
If(HierarchyExists(cDimSrc, cHierarchy) = 0);
  HierarchyCreate(cDimSrc, cHierarchy);
Else;
  unwind
EXECUTEPROCESS(Bedrock.Dim.Hierarchy.Unwind.All)
EXECUTEPROCESS(Bedrock.Dim.Hierarchy.Unwind.Consolidation)
EXECUTEPROCESS(}bedrock.hier.unwind)
SNIPPET: Unwind Dimension
SNIPPET: Unwind Target Consolidation
```

Arc will prepopulate the code with all the process's parameter. Add **cDimSrc** for **pDim** and **cHierarchy** for **pHier** and for **pConsol** we set to * because we want to unwind all consolidations:



```
EXECUTEPROCESS('}bedrock.hier.unwind',
  'pLogOutput', 0,
  'pDim', cDimSrc,
  'pHier', cHierarchy,
  'pConsol', '*',
  'pRecursive', 1,
  'pDelim', '&'
  );
```

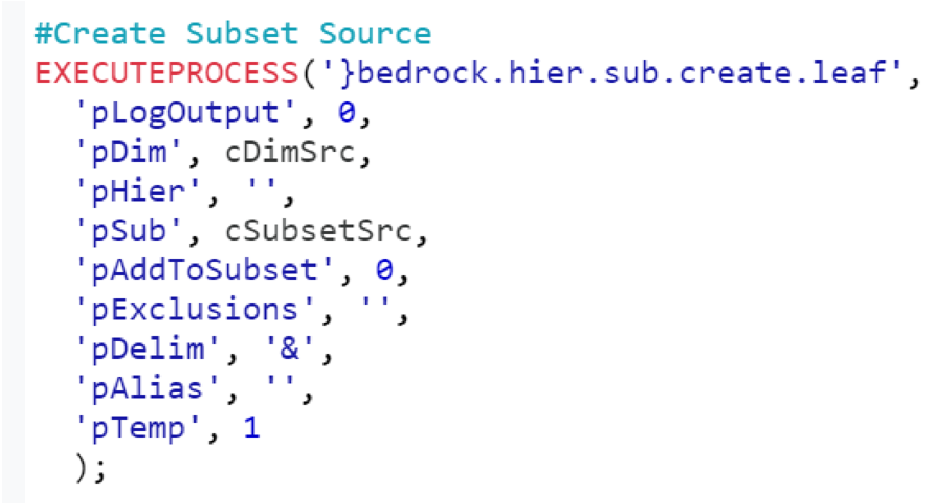
Then we insert the new hierarchy total using the function **HierarchyElementInsert**:



```
#Add Total
HierarchyElementInsert(cDimSrc, cHierarchy, '', cTotalHierarchy, 'C');

```

Currently the process's DataSource is set to the **Default** subset from the Employee dimension. This subset is static, which means that if new employees are added to the dimension, they will not show up automatically in the **Default** subset. Since we want to loop through all leaf elements in the Employee dimension, a static subset won't be useful.

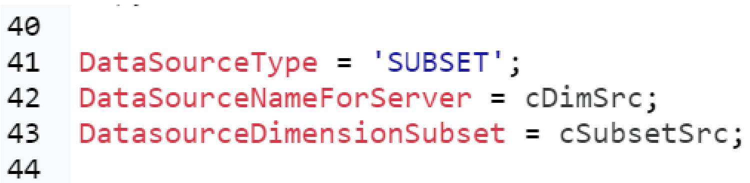
Luckily Bedrock 4 has a process that can easily create a subset with all leaf elements. This process is called, **}bedrock.hier.sub.create.leaf**:



```
#Create Subset Source
EXECUTEPROCESS('}bedrock.hier.sub.create.leaf',
  'pLogOutput', 0,
  'pDim', cDimSrc,
  'pHier', '',
  'pSub', cSubsetSrc,
  'pAddToSubset', 0,
  'pExclusions', '',
  'pDelim', '&',
  'pAlias', '',
  'pTemp', 1
  );
```

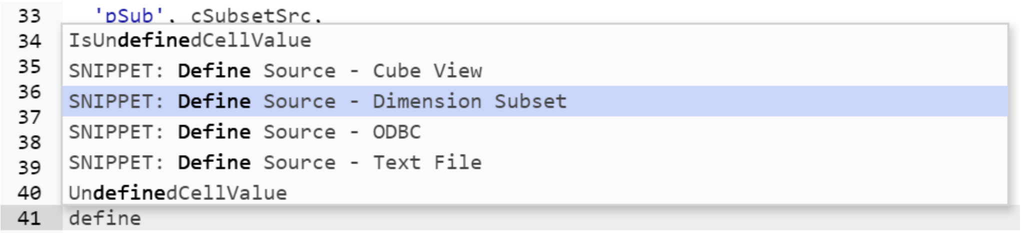
Add the **cDimSrc** to **pDim** and **cSubsetSrc** to **pSub**. The parameter **pTemp** specifies that the new subset created will be a temporary subset. This is one of the new features in IBM Planning Analytics, where a temporary subset will be deleted automatically at the end of the process execution. You don't need to use the **subsetDestroy** function in the **Epilog** anymore.

The last step in the Prolog is to override the DataSource definition with the new subset **cSubsetSrc**. To define the new data source as a subset you will have to add the following lines to your process:



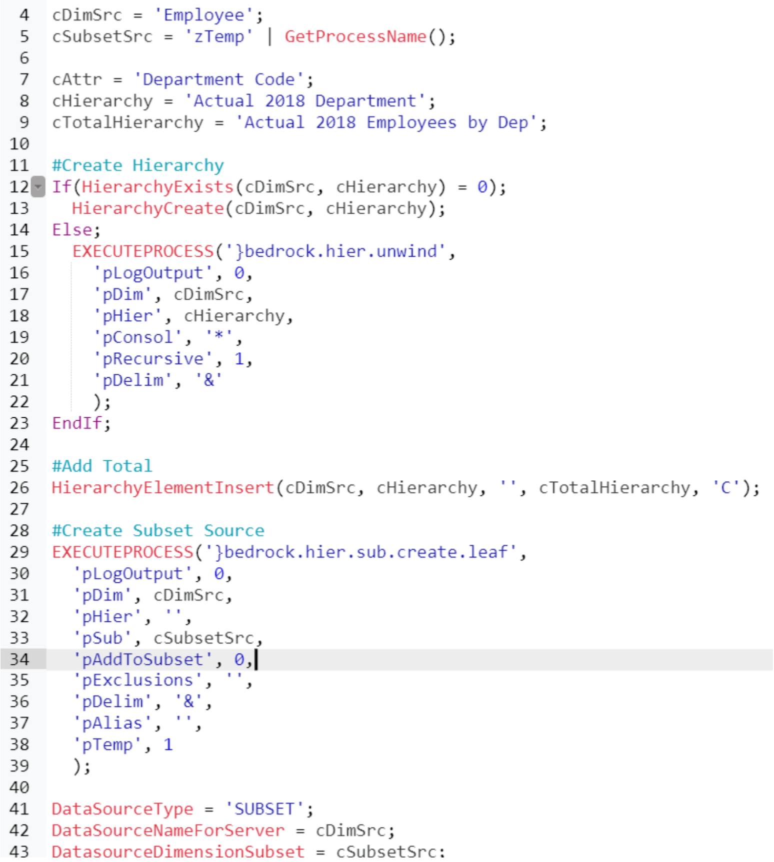
```

DataSourceType = 'SUBSET';
DataSourceNameForServer = cDimSrc;
DatasourceDimensionSubset = cSubsetSrc;

```

Thanks to Arc you don't have to type all of this, you could just use the snippets to prepopulate these lines, if you start typing **define** and then hit CTRL+SPACE, you will see all the different ways to define the source, the one we need for this example is **Dimension Subset**:



```
  'pSub', cSubsetSrc,
IsUndefinedCellValue
SNIPPET: Define Source - Cube View
SNIPPET: Define Source - Dimension Subset
SNIPPET: Define Source - ODBC
SNIPPET: Define Source - Text File
UndefinedCellValue
define
```

The **Prolog** tab will look like this



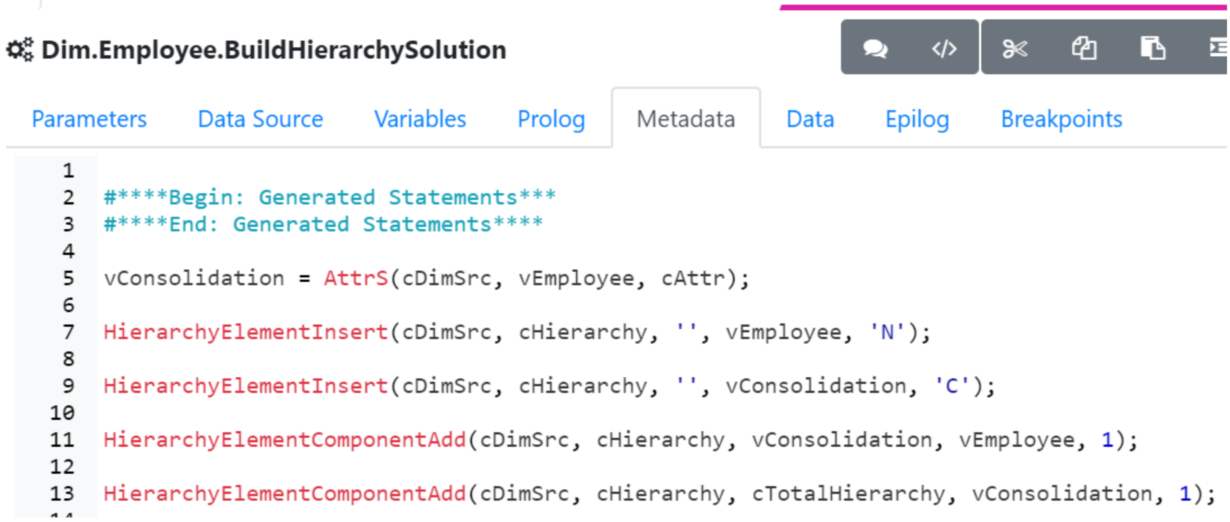
```
cDimSrc = 'Employee';
cSubsetSrc = 'zTemp' | GetProcessName();

cAttr = 'Department Code';
cHierarchy = 'Actual 2018 Department';
cTotalHierarchy = 'Actual 2018 Employees by Dep';

#Create Hierarchy
If(HierarchyExists(cDimSrc, cHierarchy) = 0);
  HierarchyCreate(cDimSrc, cHierarchy);
Else;
  EXECUTEPROCESS('}bedrock.hier.unwind',
    'pLogOutput', 0,
    'pDim', cDimSrc,
    'pHier', cHierarchy,
    'pConsol', '*',
    'pRecursive', 1,
    'pDelim', '&'
    );
EndIf;

#Add Total
HierarchyElementInsert(cDimSrc, cHierarchy, '', cTotalHierarchy, 'C');

#Create Subset Source
EXECUTEPROCESS('}bedrock.hier.sub.create.leaf',
  'pLogOutput', 0,
  'pDim', cDimSrc,
  'pHier', '',
  'pSub', cSubsetSrc,
  'pAddToSubset', 0,
  'pExclusions', '',
  'pDelim', '&',
  'pAlias', '',
  'pTemp', 1
  );

DataSourceType = 'SUBSET';
DataSourceNameForServer = cDimSrc;
DatasourceDimensionSubset = cSubsetSrc;
```

Now let's move to the **Metadata** tab. In this tab, we are going to setup the hierarchy:

- Total Department
  - Department
    - Employee

First, we need to insert new elements using **HierarchyElementInsert** and then we add the components to the consolidations using **HierarchyElementComponentAdd**.

Copy the code in the screenshot.

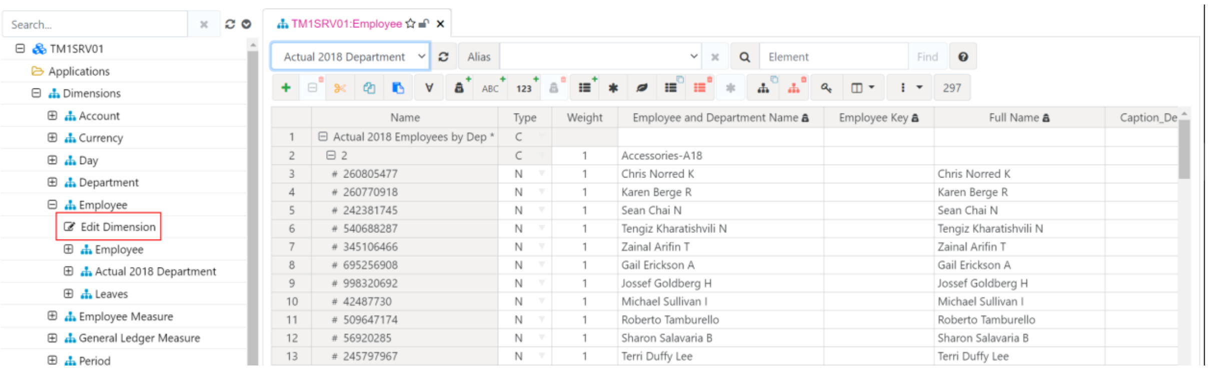


```

#****Begin: Generated Statements***
#****End: Generated Statements****

vConsolidation = AttrS(cDimSrc, vEmployee, cAttr);

HierarchyElementInsert(cDimSrc, cHierarchy, '', vEmployee, 'N');

HierarchyElementInsert(cDimSrc, cHierarchy, '', vConsolidation, 'C');

HierarchyElementComponentAdd(cDimSrc, cHierarchy, vConsolidation, vEmployee, 1);

HierarchyElementComponentAdd(cDimSrc, cHierarchy, cTotalHierarchy, vConsolidation, 1);
```

Usually the **Epilog** tab is used to delete any temporary objects created during the process, but because the subset Bedrock process will take care of this, we won't need to code any clean up functionality.

Save and then execute the process.

Open the Employee dimension. If it was already open, refresh your web browser to get the last changes. In the hierarchy dropdown, you should be able to see the new hierarchy 'Actual 2018 Department':

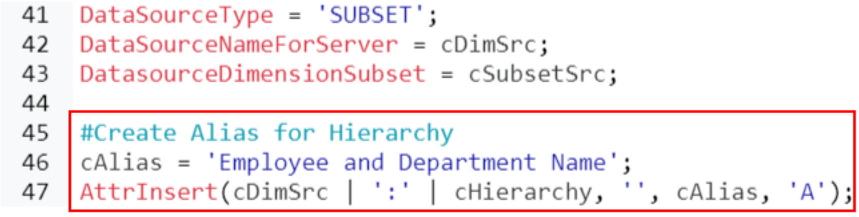


	Name	Type	Weight	Employee and Department Name	Employee Key	Full Name	Caption_De
1	⊟ Actual 2018 Employees by Dep *	C					
2	⊟ 2	C	1	Accessories-A18			
3	# 260805477	N	1	Chris Norred K		Chris Norred K	
4	# 260770918	N	1	Karen Berge R		Karen Berge R	
5	# 242381745	N	1	Sean Chai N		Sean Chai N	
6	# 540688287	N	1	Tengiz Kharatishvili N		Tengiz Kharatishvili N	
7	# 345106466	N	1	Zainal Arifin T		Zainal Arifin T	
8	# 695256908	N	1	Gail Erickson A		Gail Erickson A	
9	# 998320692	N	1	Jossef Goldberg H		Jossef Goldberg H	
10	# 42487730	N	1	Michael Sullivan I		Michael Sullivan I	
11	# 509647174	N	1	Roberto Tamburello		Roberto Tamburello	
12	# 56920285	N	1	Sharon Salavaria B		Sharon Salavaria B	
13	# 245797967	N	1	Terri Duffy Lee		Terri Duffy Lee	

We used the department code to create the consolidation, which is not very meaningful. Let's create an alias on this hierarchy to make it friendlier to read.

Open the process we had created to set up this hierarchy.

Add the alias in the **Prolog** tab with the following lines of code:

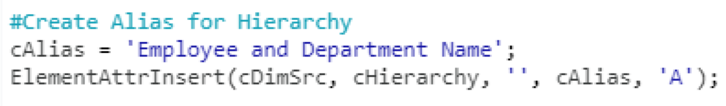


```
DataSourceType = 'SUBSET';
DataSourceNameForServer = cDimSrc;
DatasourceDimensionSubset = cSubsetSrc;

#Create Alias for Hierarchy
cAlias = 'Employee and Department Name';
AttrInsert(cDimSrc | ':' | cHierarchy, '', cAlias, 'A');
```

To avoid duplicate alias when creating a new attribute for a hierarchy only, use the new function **ElementAttrInsert** instead of **AttrInsert.** The syntax will look like this:

- ElementAttrInsert(cDimSrc, cHierarchy, '', cAlias, 'A');

Replace **AttrInsert** with **ElementAttrInsert**.

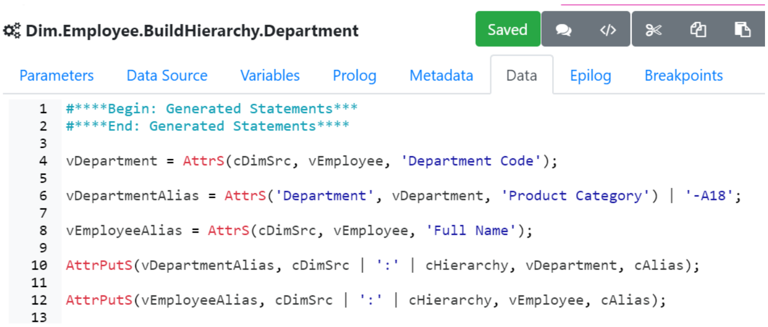


```
#Create Alias for Hierarchy
cAlias = 'Employee and Department Name';
ElementAttrInsert(cDimSrc, cHierarchy, '', cAlias, 'A');
```

The **ElementAttrInsert** will avoid duplicate alias on one consolidation which appears in two hierarchies in the same dimension.

Setting up the alias will be coded in the Data tab. We are going to retrieve the department's alias from the **Product Category** attribute in the **Department** dimension. For the employee, we will retrieve the attribute **Full Name** from the **Employee** dimension.

Copy the code below.



```
#****Begin: Generated Statements***
#****End: Generated Statements****

vDepartment = AttrS(cDimSrc, vEmployee, 'Department Code');

vDepartmentAlias = AttrS('Department', vDepartment, 'Product Category') | '-A18';

vEmployeeAlias = AttrS(cDimSrc, vEmployee, 'Full Name');

AttrPutS(vDepartmentAlias, cDimSrc | ':' | cHierarchy, vDepartment, cAlias);

AttrPutS(vEmployeeAlias, cDimSrc | ':' | cHierarchy, vEmployee, cAlias);

```

To add an attribute value to a consolidation which exists only in the hierarchy, the syntax for **ATTRPUTS** will be:

- AttrPutS(vDepartmentAlias, **cDimSrc | ':' | cHierarchy**, vDepartment, cAlias)

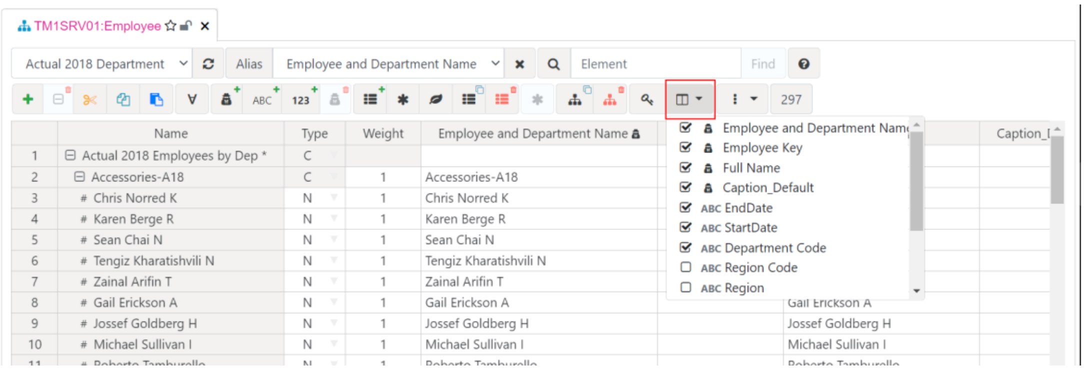
Instead of

- AttrPutS(vDepartmentAlias, **cDimSrc**, **vDepartment**, cAlias)

When using Hierarchies, the attribute value can be stored in different locations, either in the default hierarchy or in a specified hierarchy. If the hierarchy name is not specified in ATTRPUTS, the target element **vDepartment** will be the one in the default hierarchy (Department). To send the value to the **vDepartment** in our new hierarchy (cHierarchy), we use **cDimSrc | ':' | cHierarchy** instead of just **cDimSrc.**

Save and **Execute** the process.

Open the Employee dimension, if you don't see the alias just refresh your browser and make sure 'Employee and Department Name' is selected as the alias:



☑ Employee and Department Name
☑ Employee Key
☑ Full Name
☑ Caption_Default
☑ EndDate
☑ StartDate
☑ Department Code
☐ Region Code
☐ Region

	Name	Type	Weight	Employee and Department Name		Caption_[
1	⊟ Actual 2018 Employees by Dep *	C				
2	⊟ Accessories-A18	C	1	Accessories-A18		
3	# Chris Norred K	N	1	Chris Norred K		
4	# Karen Berge R	N	1	Karen Berge R		
5	# Sean Chai N	N	1	Sean Chai N		
6	# Tengiz Kharatishvili N	N	1	Tengiz Kharatishvili N		
7	# Zainal Arifin T	N	1	Zainal Arifin T		
8	# Gail Erickson A	N	1	Gail Erickson A	Gail Erickson A	
9	# Jossef Goldberg H	N	1	Jossef Goldberg H	Jossef Goldberg H	
10	# Michael Sullivan I	N	1	Michael Sullivan I	Michael Sullivan I	
11	# Roberto Tamburello	N	1	Roberto Tamburello	Roberto Tamburello	

Build hierarchies with TM1 process solution

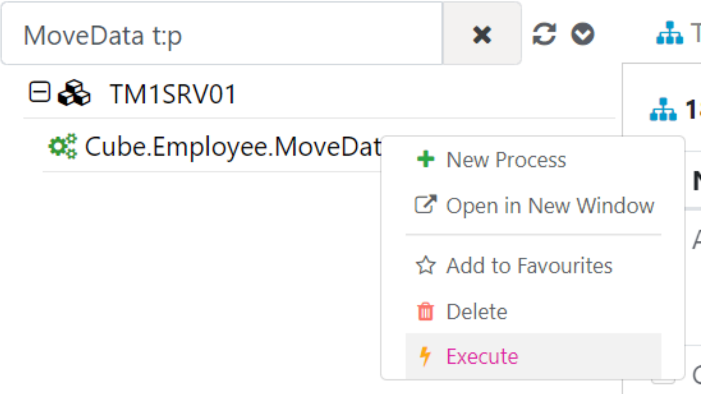
If you are struggling to make your process executed successfully, you can check the solution in the following process:

- **Dim.Employee.BuildHierarchySolution**

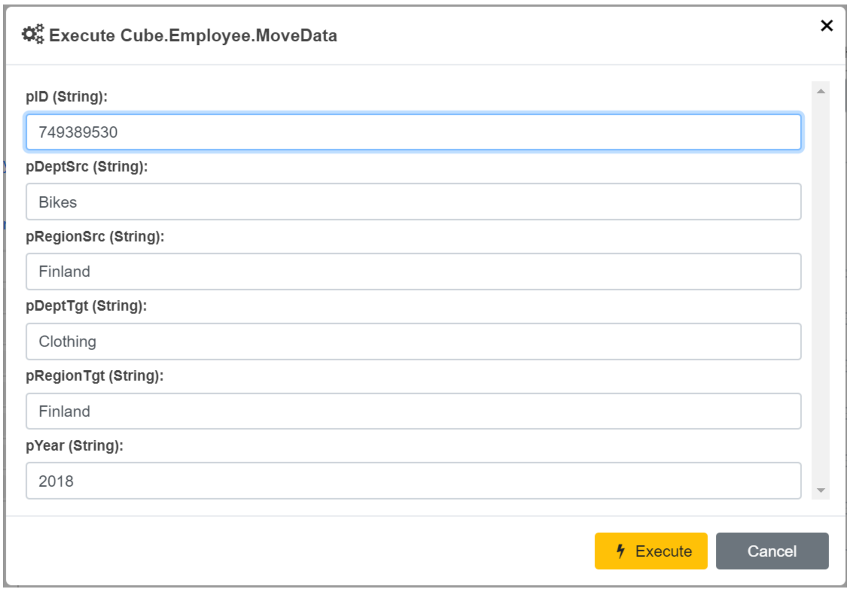
Compare Actual vs Budget Hierarchies

Now that we backed up our **Actual 2018** employees by department via Hierarchy, let's now move some employees from their current department to a new department and then create a new hierarchy **Budget 2018**.

To move employees, we will execute the process **Cube.Employee.MoveData**, opens this process using Arc:



Click the Execute button from the toolbar and move the first employee **749389530** from department **Bikes** to **Clothing** for year 2018:



Execute Cube.Employee.MoveData

pID (String): 749389530

pDeptSrc (String): Bikes

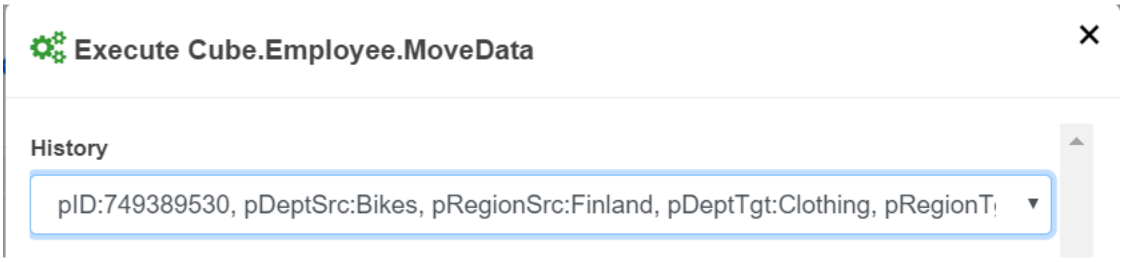
pRegionSrc (String): Finland

pDeptTgt (String): Clothing

pRegionTgt (String): Finland

pYear (String): 2018

Repeat the above steps to move more employees based on the screenshots below, it can be noted that Arc keeps your execution history in the top dropdown as below:

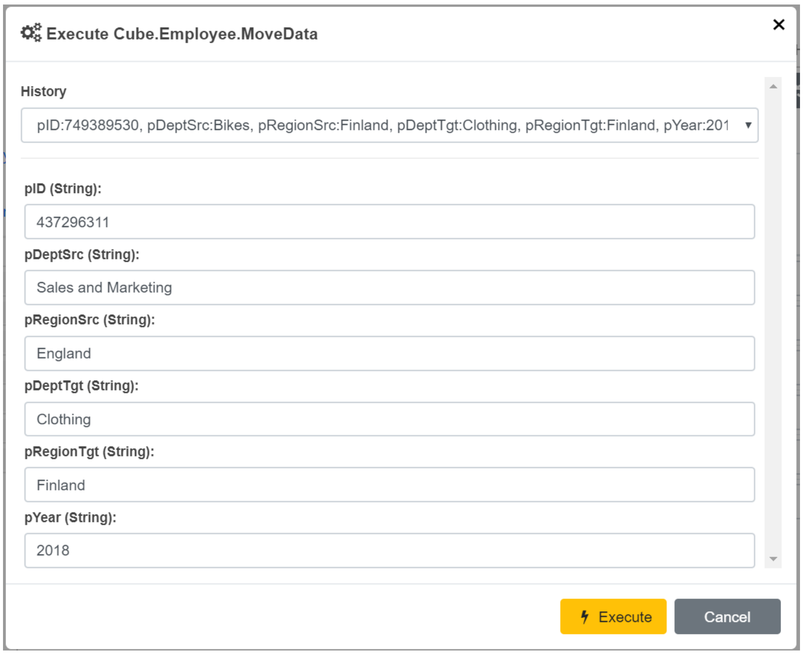


Execute Cube.Employee.MoveData

History

pID:749389530, pDeptSrc:Bikes, pRegionSrc:Finland, pDeptTgt:Clothing, pRegionT

Let's run again this process for the following two employees as below:



History: pID:749389530, pDeptSrc:Bikes, pRegionSrc:Finland, pDeptTgt:Clothing, pRegionTgt:Finland, pYear:201

pID (String): 437296311

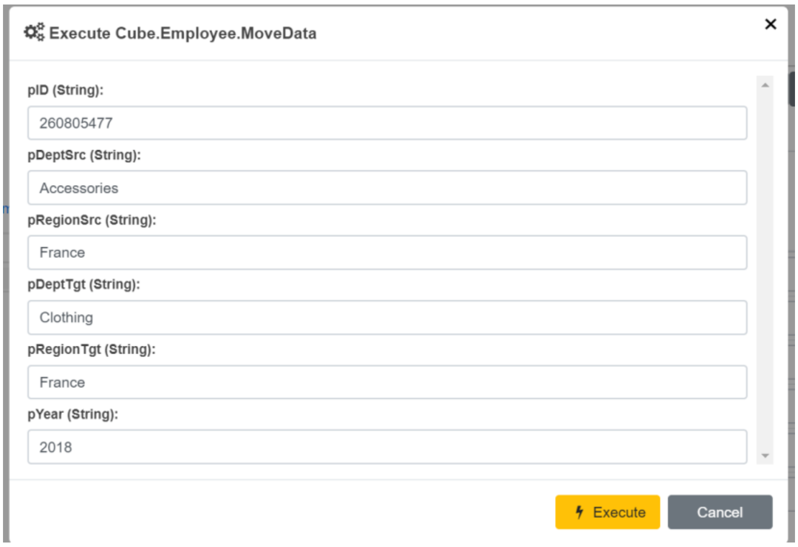
pDeptSrc (String): Sales and Marketing

pRegionSrc (String): England

pDeptTgt (String): Clothing

pRegionTgt (String): Finland

pYear (String): 2018



pID (String): 260805477

pDeptSrc (String): Accessories

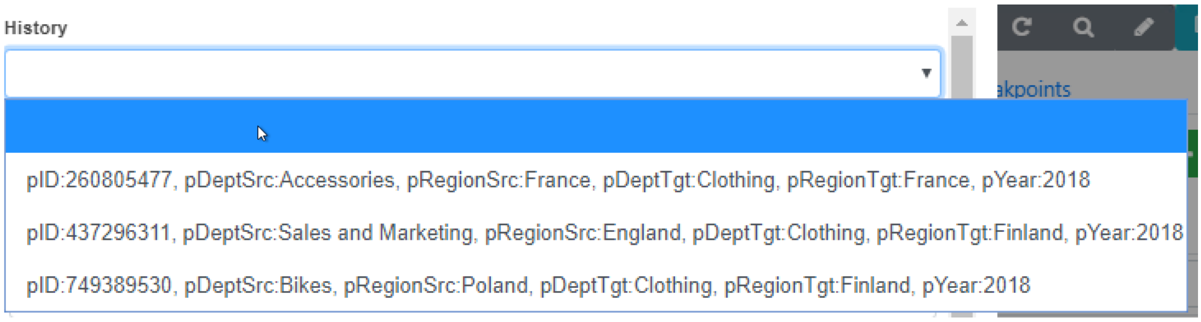
pRegionSrc (String): France

pDeptTgt (String): Clothing

pRegionTgt (String): France

pYear (String): 2018

When executing the process for the second time, Arc will remember the previous parameters values. Prior parameters can be selected from the History dropdown.



pID:260805477, pDeptSrc:Accessories, pRegionSrc:France, pDeptTgt:Clothing, pRegionTgt:France, pYear:2018

pID:437296311, pDeptSrc:Sales and Marketing, pRegionSrc:England, pDeptTgt:Clothing, pRegionTgt:Finland, pYear:2018

pID:749389530, pDeptSrc:Bikes, pRegionSrc:Poland, pDeptTgt:Clothing, pRegionTgt:Finland, pYear:2018

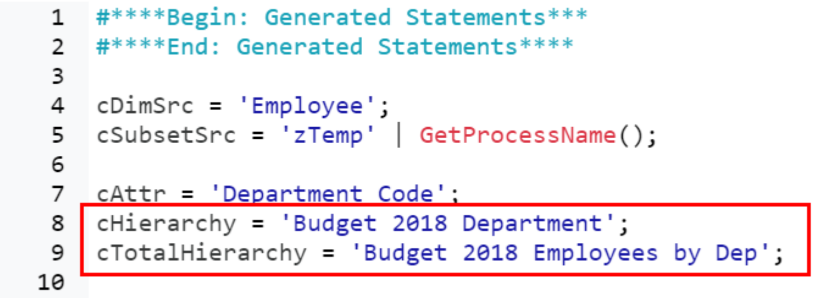
Build Budget 2018 Department hierarchy

Now that we have moved three employees, it is time to back up the new structure. To do this, we are going to use the TI we just built. Rename the hierarchy and the top consolidation.

Open the TI:

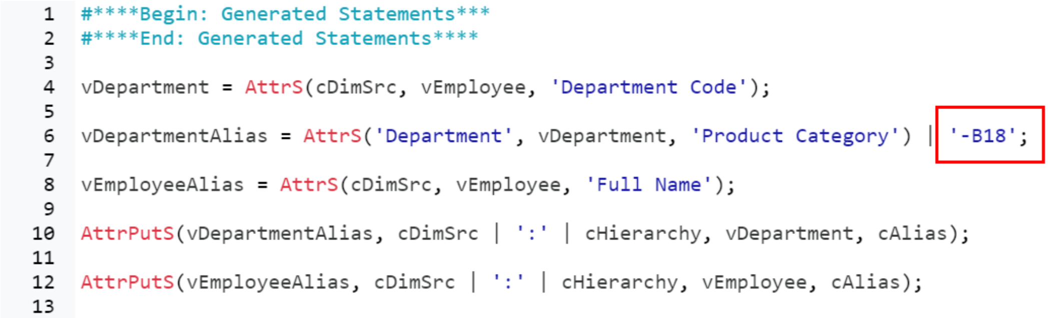
- **Dim.Employee.BuildHierarchy.Department**

In the **Prolog** tab, replace Actual with Budget for the **cHierarchy** and **cTotalHierarchy** constants:



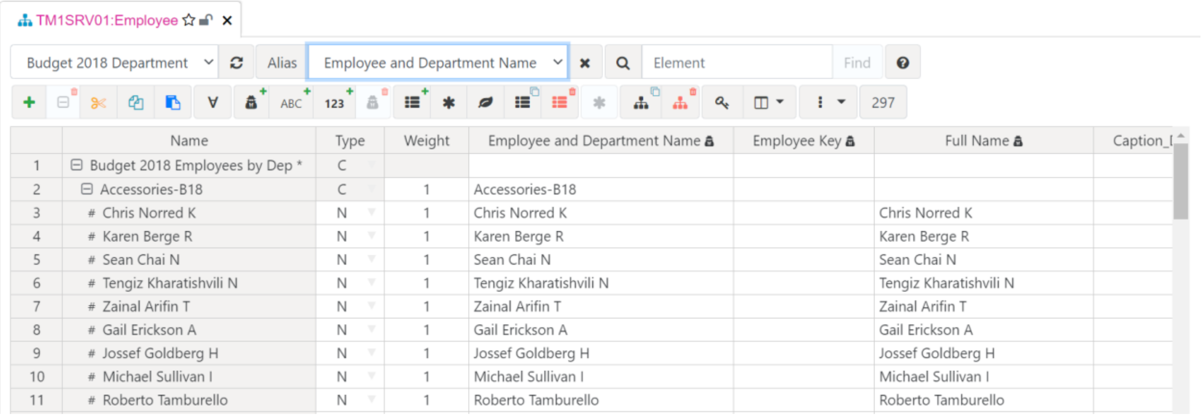
```
#****Begin: Generated Statements***
#****End: Generated Statements****

cDimSrc = 'Employee';
cSubsetSrc = 'zTemp' | GetProcessName();

cAttr = 'Department Code';
cHierarchy = 'Budget 2018 Department';
cTotalHierarchy = 'Budget 2018 Employees by Dep';

```

In the Data tab, replace the suffix from **A18** to **B18** to make sure the alias is unique:



```
#****Begin: Generated Statements***
#****End: Generated Statements****

vDepartment = AttrS(cDimSrc, vEmployee, 'Department Code');

vDepartmentAlias = AttrS('Department', vDepartment, 'Product Category') | '-B18';

vEmployeeAlias = AttrS(cDimSrc, vEmployee, 'Full Name');

AttrPutS(vDepartmentAlias, cDimSrc | ':' | cHierarchy, vDepartment, cAlias);

AttrPutS(vEmployeeAlias, cDimSrc | ':' | cHierarchy, vEmployee, cAlias);

```

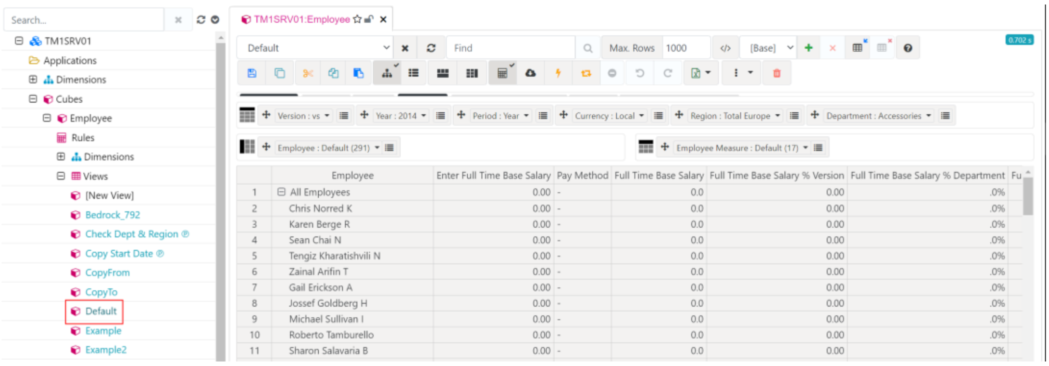
Save and execute. Refresh the Employee dimension and you should see the new hierarchy **Budget 2018 Department**:



	Name	Type	Weight	Employee and Department Name	Employee Key	Full Name	Caption_[
1	⊟ Budget 2018 Employees by Dep *	C					
2	⊟ Accessories-B18	C	1	Accessories-B18			
3	# Chris Norred K	N	1	Chris Norred K		Chris Norred K	
4	# Karen Berge R	N	1	Karen Berge R		Karen Berge R	
5	# Sean Chai N	N	1	Sean Chai N		Sean Chai N	
6	# Tengiz Kharatishvili N	N	1	Tengiz Kharatishvili N		Tengiz Kharatishvili N	
7	# Zainal Arifin T	N	1	Zainal Arifin T		Zainal Arifin T	
8	# Gail Erickson A	N	1	Gail Erickson A		Gail Erickson A	
9	# Jossef Goldberg H	N	1	Jossef Goldberg H		Jossef Goldberg H	
10	# Michael Sullivan I	N	1	Michael Sullivan I		Michael Sullivan I	
11	# Roberto Tamburello	N	1	Roberto Tamburello		Roberto Tamburello	

Compare the two hierarchies

To compare the two hierarchies built previously, we are going to use the Arc cube viewer. Open the **Default** view from the **Employee** cube:

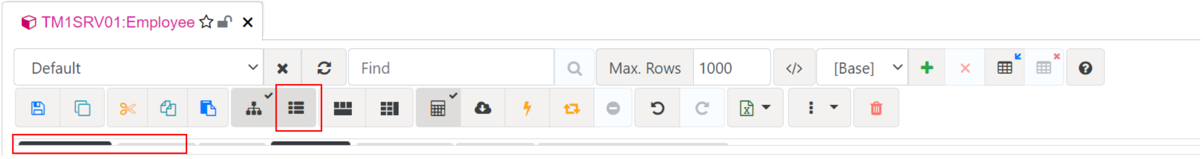


Version : vs | Year : 2014 | Period : Year | Currency : Local | Region : Total Europe | Department : Accessories

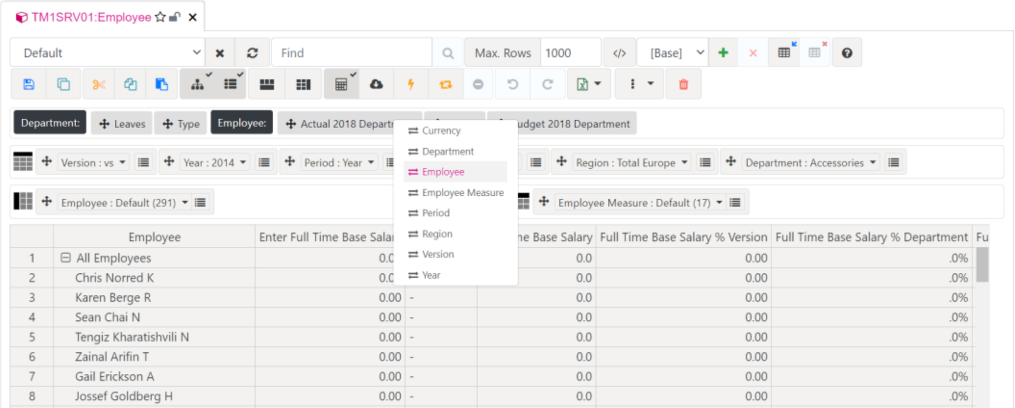
Employee : Default (291) | Employee Measure : Default (17)

	Employee	Enter Full Time Base Salary	Pay Method	Full Time Base Salary	Full Time Base Salary % Version	Full Time Base Salary % Department	Fu
1	⊟ All Employees	0.00	-	0.0	0.00	.0%	
2	Chris Norred K	0.00	-	0.0	0.00	.0%	
3	Karen Berge R	0.00	-	0.0	0.00	.0%	
4	Sean Chai N	0.00	-	0.0	0.00	.0%	
5	Tengiz Kharatishvili N	0.00	-	0.0	0.00	.0%	
6	Zainal Arifin T	0.00	-	0.0	0.00	.0%	
7	Gail Erickson A	0.00	-	0.0	0.00	.0%	
8	Jossef Goldberg H	0.00	-	0.0	0.00	.0%	
9	Michael Sullivan I	0.00	-	0.0	0.00	.0%	
10	Roberto Tamburello	0.00	-	0.0	0.00	.0%	
11	Sharon Salavaria B	0.00	-	0.0	0.00	.0%	

To see the Hierarchies section, just



On rows, replace the Default Employee Hierarchy with **Actual 2018 Department** Hierarchy. To do that just right-click on the **Actual 2018 Department** and select Employee:

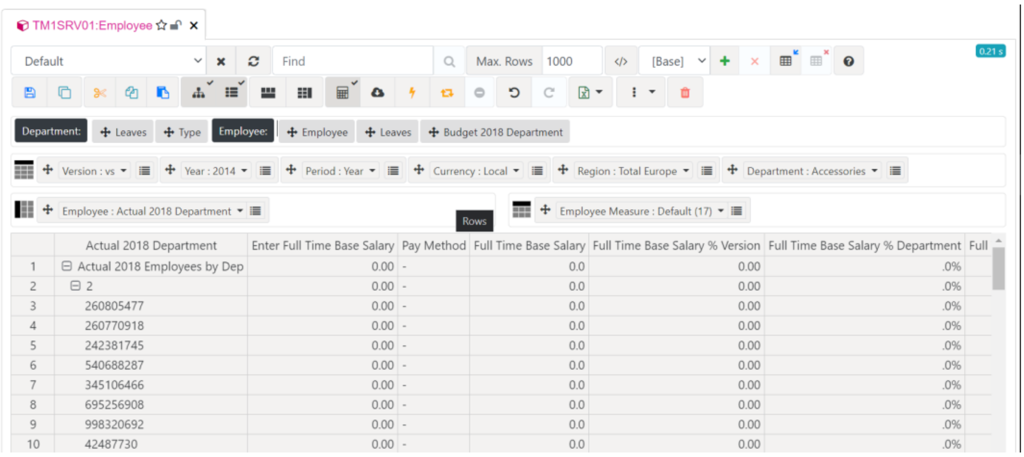


Version : vs | Year : 2014 | Period : Year | Region : Total Europe | Department : Accessories

Employee : Default (291) | Employee Measure : Default (17)

	Employee	Enter Full Time Base Salaı	ne Base Salary	Full Time Base Salary % Version	Full Time Base Salary % Department	Fu
1	⊟ All Employees	0.0	0.0	0.00	.0%	
2	Chris Norred K	0.0	0.0	0.00	.0%	
3	Karen Berge R	0.00 -	0.0	0.00	.0%	
4	Sean Chai N	0.00 -	0.0	0.00	.0%	
5	Tengiz Kharatishvili N	0.00 -	0.0	0.00	.0%	
6	Zainal Arifin T	0.00 -	0.0	0.00	.0%	
7	Gail Erickson A	0.00 -	0.0	0.00	.0%	
8	Jossef Goldberg H	0.00 -	0.0	0.00	.0%	

Arc is going to switch the two hierarchies:

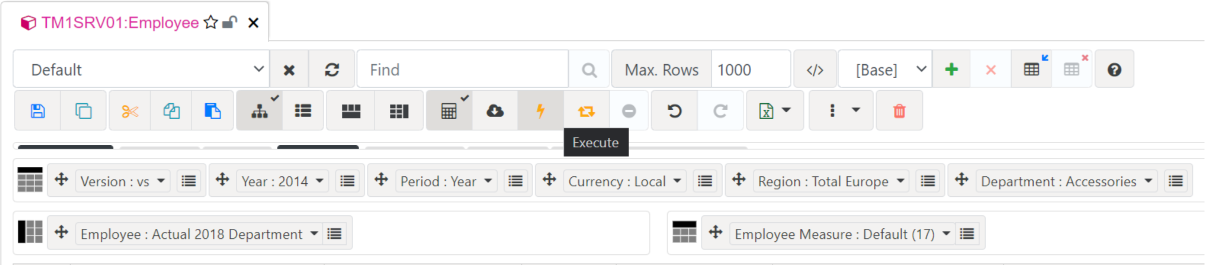


Version : vs | Year : 2014 | Period : Year | Currency : Local | Region : Total Europe | Department : Accessories

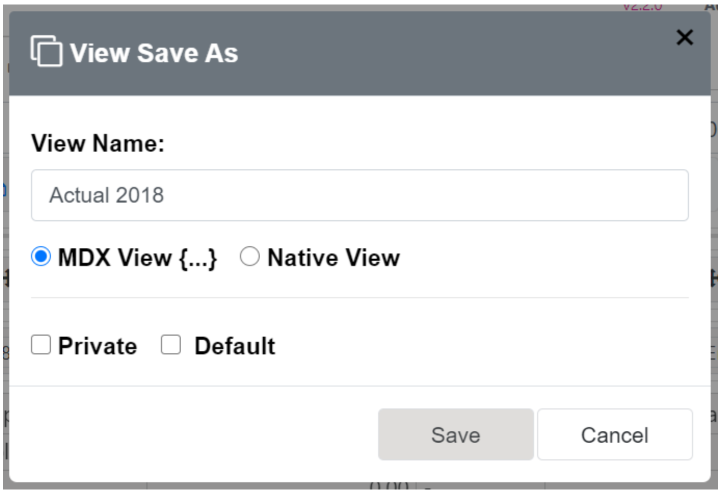
Employee : Actual 2018 Department | Employee Measure : Default (17)

	Actual 2018 Department	Enter Full Time Base Salary	Pay Method	Full Time Base Salary	Full Time Base Salary % Version	Full Time Base Salary % Department	Full
1	⊟ Actual 2018 Employees by Dep	0.00	-	0.0	0.00	.0%	
2	⊟ 2	0.00	-	0.0	0.00	.0%	
3	260805477	0.00	-	0.0	0.00	.0%	
4	260770918	0.00	-	0.0	0.00	.0%	
5	242381745	0.00	-	0.0	0.00	.0%	
6	540688287	0.00	-	0.0	0.00	.0%	
7	345106466	0.00	-	0.0	0.00	.0%	
8	695256908	0.00	-	0.0	0.00	.0%	
9	998320692	0.00	-	0.0	0.00	.0%	
10	42487730	0.00	-	0.0	0.00	.0%	

To refresh the view by clicking the **Execute** button:



Finally click the Save button with **Actual 2018** as the view name:



View Save As

View Name: Actual 2018

● MDX View {...} ○ Native View

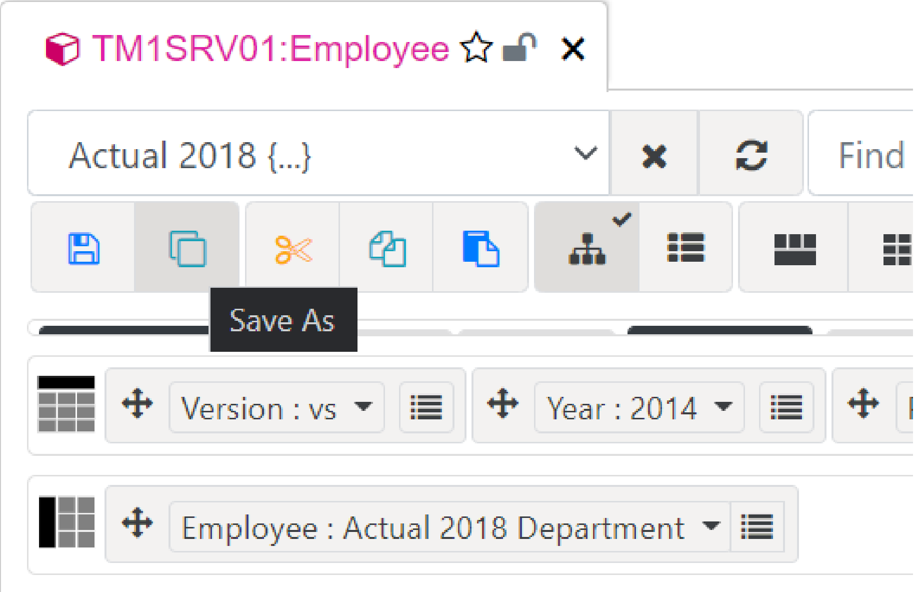
□ Private □ Default

A native view is the name of the standard view you have been used to using in TM1 for many years. These views do not support the new hierarchies and are essentially deprecated. Arc supports reading and creating native views since Arc v1.9.

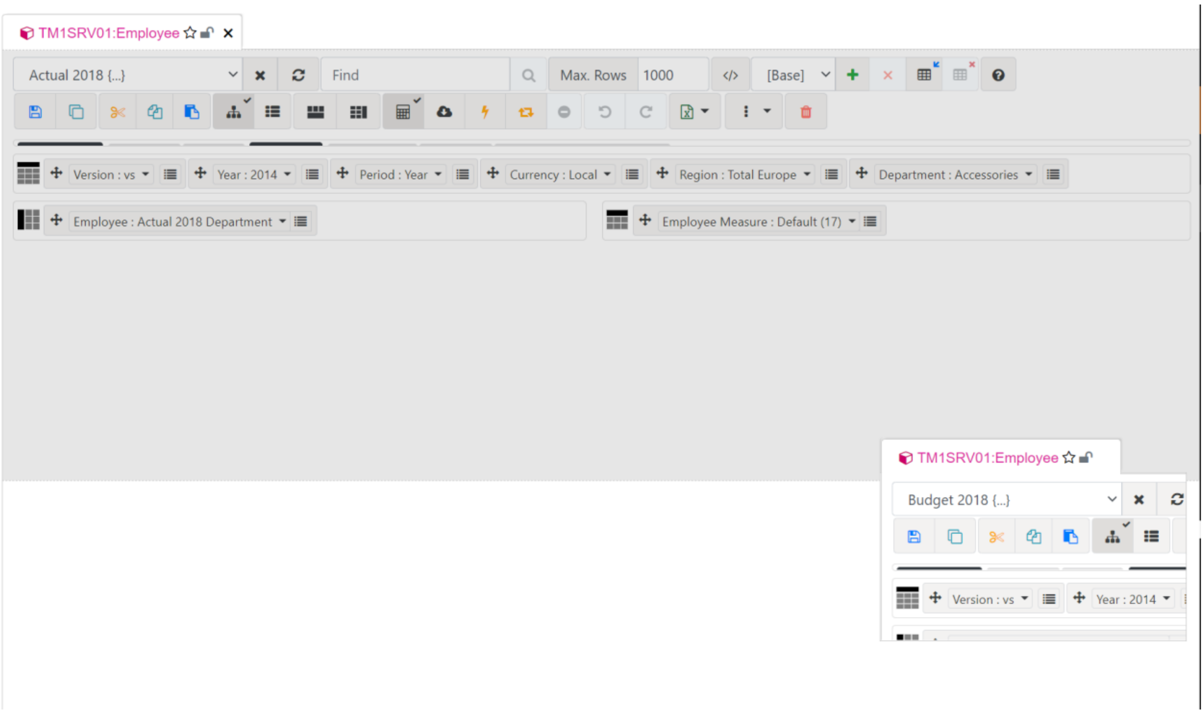
MDX views were added to TM1 to support hierarchies, these views contain a single MDX statement. The Arc cube viewer is based on MDX, each time you drag/drop a hierarchy or select a set of elements the MDX is updated and executed against the TM1 model.

You can save MDX views with Arc, these views can be used in the newer TM1 interfaces such as PAX and PAW.

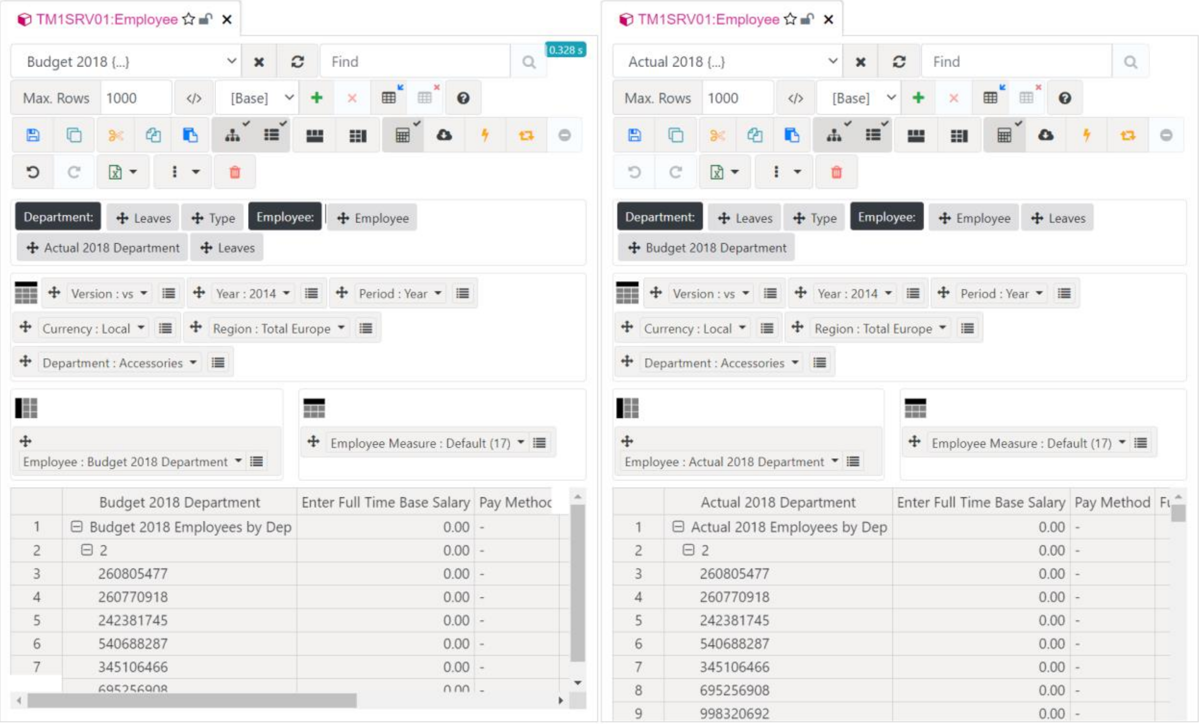
Create a new view called Budget 2018 by showing the **Budget 2018 Department** and then show side by side the two views to compare the two Hierarchies. To create the Budget view, you just need to replace the **Actual 2018** hierarchy with the **Budget 2018** and click the **Save As** button:



Now that the two views have been created, to show them side by side, you just need to drag one tab to the left and you will see a grey area which will show where the tab will be shown as below:



By using Hierarchies, you can compare the same data by two independent structures:



TM1SRV01:Employee

Budget 2018 (...)

Max. Rows 1000 [Base]

Department: Leaves, Type
Employee: Employee
Actual 2018 Department, Leaves

Version : vs
Year : 2014
Period : Year
Currency : Local
Region : Total Europe
Department : Accessories

Employee : Budget 2018 Department
Employee Measure : Default (17)

	Budget 2018 Department	Enter Full Time Base Salary	Pay Methoc
1	⊟ Budget 2018 Employees by Dep	0.00	-
2	⊟ 2	0.00	-
3	260805477	0.00	-
4	260770918	0.00	-
5	242381745	0.00	-
6	540688287	0.00	-
7	345106466	0.00	-
	695256908	0.00	-

TM1SRV01:Employee

Actual 2018 (...)

Max. Rows 1000 [Base]

Department: Leaves, Type
Employee: Employee, Leaves
Budget 2018 Department

Version : vs
Year : 2014
Period : Year
Currency : Local
Region : Total Europe
Department : Accessories

Employee : Actual 2018 Department
Employee Measure : Default (17)

	Actual 2018 Department	Enter Full Time Base Salary	Pay Method	Fu
1	⊟ Actual 2018 Employees by Dep	0.00	-	
2	⊟ 2	0.00	-	
3	260805477	0.00	-	
4	260770918	0.00	-	
5	242381745	0.00	-	
6	540688287	0.00	-	
7	345106466	0.00	-	
8	695256908	0.00	-	
9	998320692	0.00	-	

In Arc you can move tabs by using the tab header, just drag the tab header where you want to show it.

TOPIC 3: DEBUGGING A PROCESS

Introduction to debugging

What is debugging?

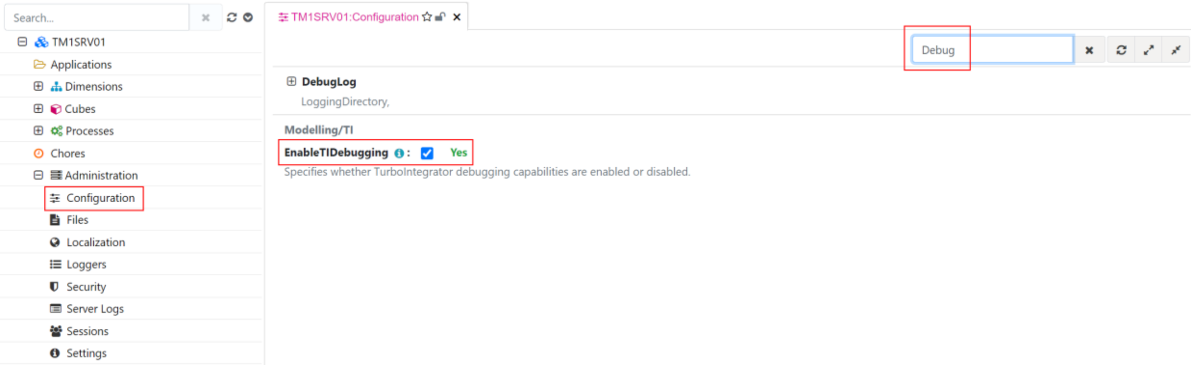
Debugging code allows you to step through line-by-line and see what the values of the variables are as you go. No need to write out to ASCII files with what is going on in TM1 process anymore.

Why debugging?

Debugging is part of the core process of any software developments, now that it is available with TM1/Planning Analytics, it can save you a lot of time. You can now step in to your code and see the values of the variables changing in real time.

Key Debugging terms

- **Enable debugging:** EnableTIDebugging=true (tm1s.cfg). To check a TM1 setting, you can use the Arc Configuration module under Administration and then use the search bar to quickly find the setting:



DebugLog
LoggingDirectory,

Modelling/TI

EnableTIDebugging : ☑ Yes

Specifies whether TurboIntegrator debugging capabilities are enabled or disabled.

- **Debugging**: moving line by line through source code while it is executing.
- **Breakpoint**: a predefined place where the code pauses during execution.
- **Continue**: Run code until the end of the process or a breakpoint (more on these later) is found.
- **Step In**: Moves line to line and will also enter a child process called by the *ExecuteProcess* function.
- **Step Over**: Works the same as Step In by moving line to line except it DOESN'T enter a child process called by *ExecuteProcess*.
- **Step Out**: Continues until the end of the child process and then stops at the next line in the calling process. This should be used with Step In to exit a child process called via *ExecuteProcess*.

Debug your first process

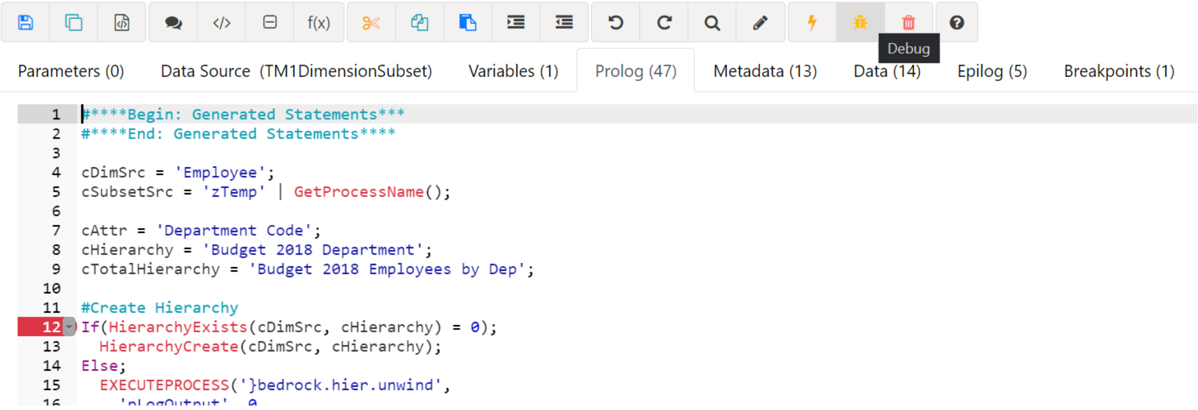
In this exercise we are going to step through the process built previously, Open the TI:

- **Dim.Employee.BuildHierarchy.Department**

If your process is not working, you can continue the training using the process **Dim.Employee.BuildHierarchySolution.**

Add a breakpoint

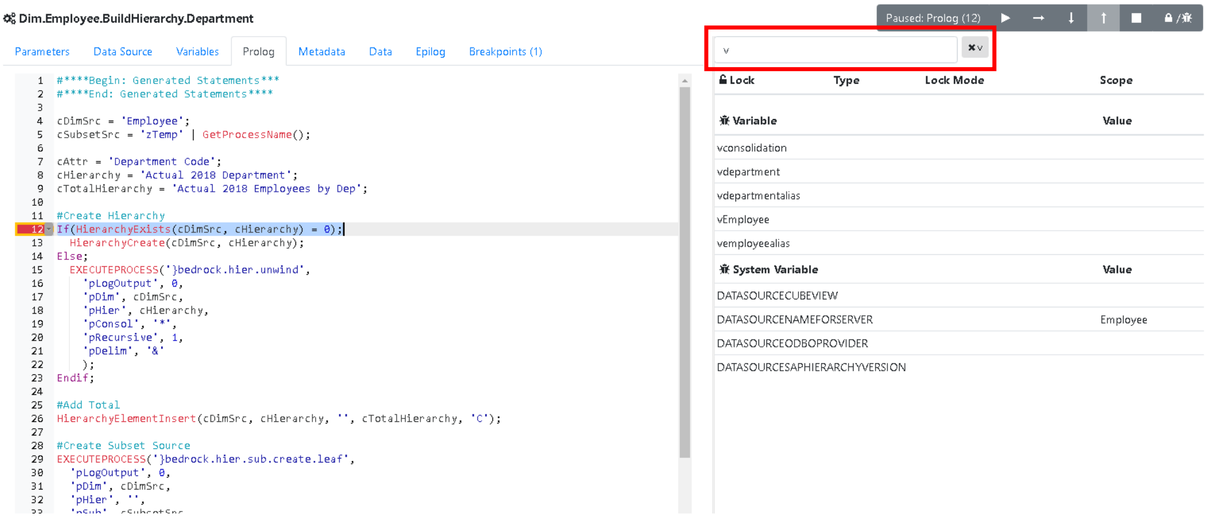
Before running as debug mode, we need to add a breakpoint to make sure the process will stop, just click on row 12 to add a new breakpoint:



```
 1  #****Begin: Generated Statements***
 2  #****End: Generated Statements****
 3
 4  cDimSrc = 'Employee';
 5  cSubsetSrc = 'zTemp' | GetProcessName();
 6
 7  cAttr = 'Department Code';
 8  cHierarchy = 'Budget 2018 Department';
 9  cTotalHierarchy = 'Budget 2018 Employees by Dep';
10
11  #Create Hierarchy
12  If(HierarchyExists(cDimSrc, cHierarchy) = 0);
13    HierarchyCreate(cDimSrc, cHierarchy);
14  Else;
15    EXECUTEPROCESS('}bedrock.hier.unwind',
16      'pLogOutput', 0
```

Run the process with debug mode and click the **continue** button to see the values updating.

Filter the list of variables to show only the variables starting with v:



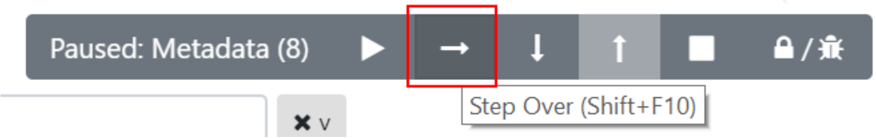
```
 1  #****Begin: Generated Statements***
 2  #****End: Generated Statements****
 3
 4  cDimSrc = 'Employee';
 5  cSubsetSrc = 'zTemp' | GetProcessName();
 6
 7  cAttr = 'Department Code';
 8  cHierarchy = 'Actual 2018 Department';
 9  cTotalHierarchy = 'Actual 2018 Employees by Dep';
10
11  #Create Hierarchy
12  If(HierarchyExists(cDimSrc, cHierarchy) = 0);
13    HierarchyCreate(cDimSrc, cHierarchy);
14  Else;
15    EXECUTEPROCESS('}bedrock.hier.unwind',
16      'pLogOutput', 0,
17      'pDim', cDimSrc,
18      'pHier', cHierarchy,
19      'pConsol', '*',
20      'pRecursive', 1,
21      'pDelim', '&'
22     );
23  Endif;
24
25  #Add Total
26  HierarchyElementInsert(cDimSrc, cHierarchy, '', cTotalHierarchy, 'C');
27
28  #Create Subset Source
29  EXECUTEPROCESS('}bedrock.hier.sub.create.leaf',
30    'pLogOutput', 0,
31    'pDim', cDimSrc,
32    'pHier', '',
33    'pSub', cSubsetSrc
```

Lock	Type	Lock Mode	Scope

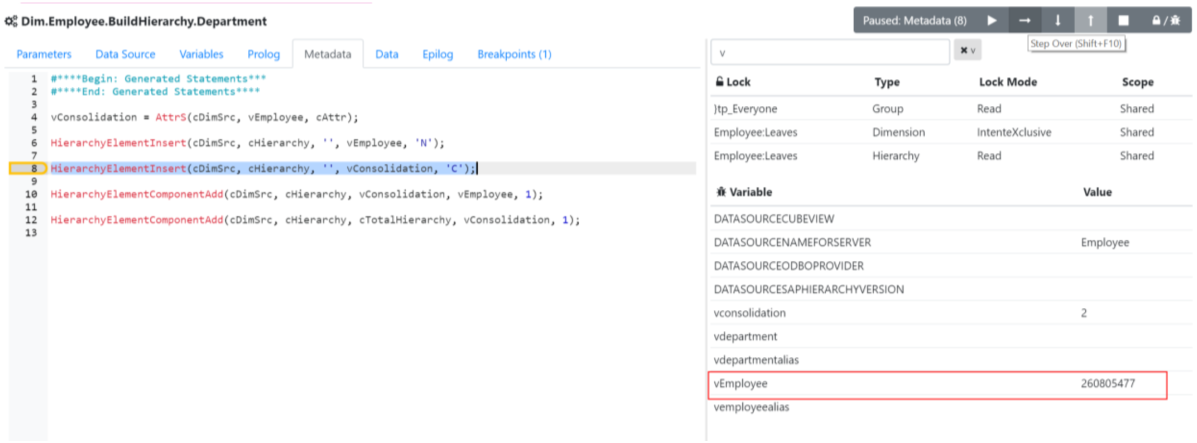
Variable	Value
vconsolidation	
vdepartment	
vdepartmentalias	
vEmployee	
vemployeealias	

System Variable	Value
DATASOURCECUBEVIEW	
DATASOURCENAMEFORSERVER	Employee
DATASOURCEODBOPROVIDER	
DATASOURCESAPHIERARCHYVERSION	

We then use the Step Over button to reach the data folder:



Let’s stop in the Data tab, we can see now that vEmployee has a value:



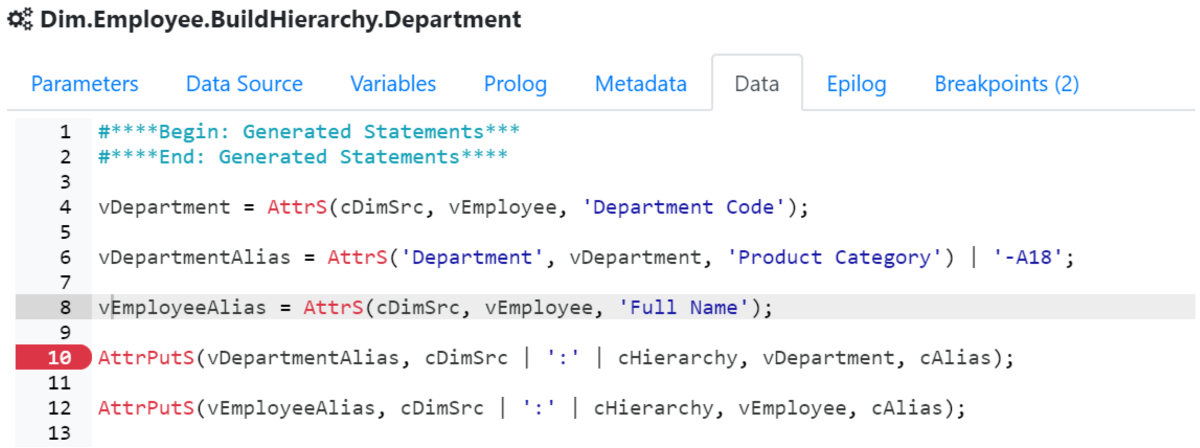
```
#****Begin: Generated Statements***
#****End: Generated Statements****

vConsolidation = AttrS(cDimSrc, vEmployee, cAttr);

HierarchyElementInsert(cDimSrc, cHierarchy, '', vEmployee, 'N');

HierarchyElementInsert(cDimSrc, cHierarchy, '', vConsolidation, 'C');

HierarchyElementComponentAdd(cDimSrc, cHierarchy, vConsolidation, vEmployee, 1);

HierarchyElementComponentAdd(cDimSrc, cHierarchy, cTotalHierarchy, vConsolidation, 1);
```

Lock	Type	Lock Mode	Scope
}tp_Everyone	Group	Read	Shared
Employee:Leaves	Dimension	IntenteXclusive	Shared
Employee:Leaves	Hierarchy	Read	Shared

Variable	Value
DATASOURCECUBEVIEW	
DATASOURCENAMEFORSERVER	Employee
DATASOURCEODBOPROVIDER	
DATASOURCESAPHIERARCHYVERSION	
vconsolidation	2
vdepartment	
vdepartmentalias	
vEmployee	260805477
vemployeealias	

Add a condition to the breakpoint

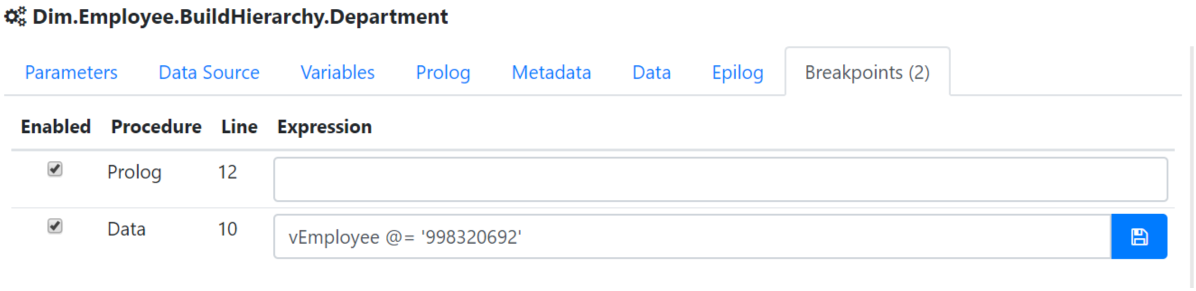
Instead of stopping for every employee, we can add an expression to our breakpoint to stop only for a specific Employee. Let’s add a new breakpoint in the Data tab:



```
#****Begin: Generated Statements***
#****End: Generated Statements****

vDepartment = AttrS(cDimSrc, vEmployee, 'Department Code');

vDepartmentAlias = AttrS('Department', vDepartment, 'Product Category') | '-A18';

vEmployeeAlias = AttrS(cDimSrc, vEmployee, 'Full Name');

AttrPutS(vDepartmentAlias, cDimSrc | ':' | cHierarchy, vDepartment, cAlias);

AttrPutS(vEmployeeAlias, cDimSrc | ':' | cHierarchy, vEmployee, cAlias);
```

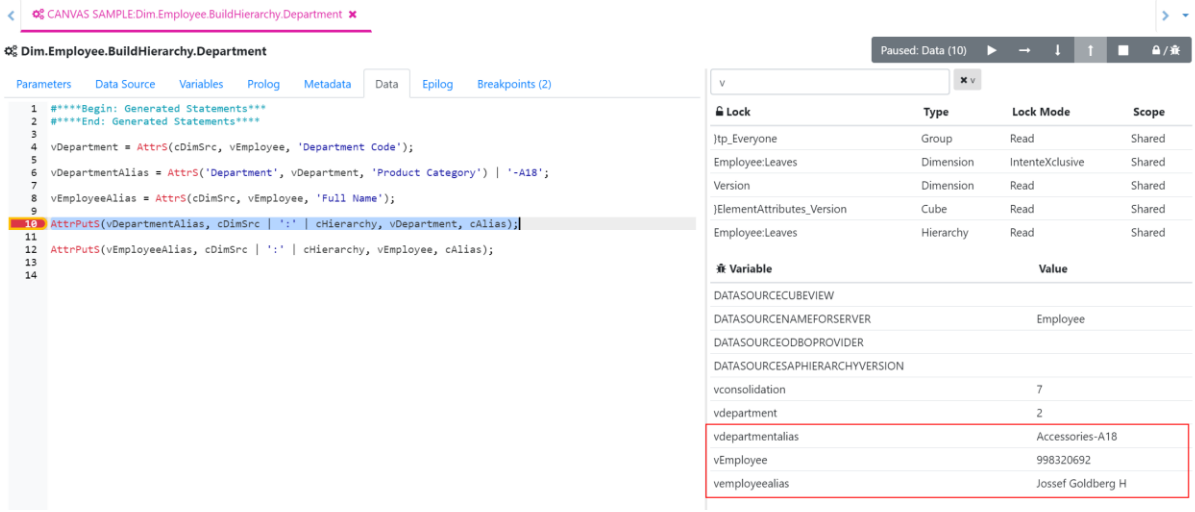
Add this condition in the Breakpoints tab then save:

- vEmployee @= '998320692'



Enabled	Procedure	Line	Expression
☑	Prolog	12	
☑	Data	10	vEmployee @= '998320692'

Run the continue button ▶ and you will see that the process will stop at employee ‘998320692’:



CANVAS SAMPLE:Dim.Employee.BuildHierarchy.Department
Dim.Employee.BuildHierarchy.Department
Parameters
Data Source
Variables
Prolog
Metadata
Data
Epilog
Breakpoints (2)
Paused: Data (10)

```
#****Begin: Generated Statements***
#****End: Generated Statements****

vDepartment = AttrS(cDimSrc, vEmployee, 'Department Code');

vDepartmentAlias = AttrS('Department', vDepartment, 'Product Category') | '-A18';

vEmployeeAlias = AttrS(cDimSrc, vEmployee, 'Full Name');

AttrPutS(vDepartmentAlias, cDimSrc | ':' | cHierarchy, vDepartment, cAlias);

AttrPutS(vEmployeeAlias, cDimSrc | ':' | cHierarchy, vEmployee, cAlias);
```

v

Lock	Type	Lock Mode	Scope
}tp_Everyone	Group	Read	Shared
Employee:Leaves	Dimension	IntenteXclusive	Shared
Version	Dimension	Read	Shared
}ElementAttributes_Version	Cube	Read	Shared
Employee:Leaves	Hierarchy	Read	Shared

Variable	Value
DATASOURCECUBEVIEW	
DATASOURCENAMEFORSERVER	Employee
DATASOURCEODBOPROVIDER	
DATASOURCESAPHIERARCHYVERSION	
vconsolidation	7
vdepartment	2
vdepartmentalias	Accessories-A18
vEmployee	998320692
vemployeealias	Jossef Goldberg H

TOPIC 4: FIND YOUR OBJECTS FASTER

Arc comes with a summary page for each type of TM1 objects, we will see in this section how you can use them to find your objects faster.

Objectives

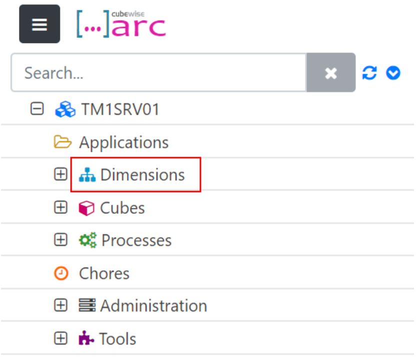
In this section, students will learn:

- How to filter dimensions per text, cubes and parameters
- How to filter cubes per text and dimensions
- Activate/Deactivate Logging
- Execute SaveData and CubeProcessFeeders for selected cube
- How to filter processes by text, parameters and data source type

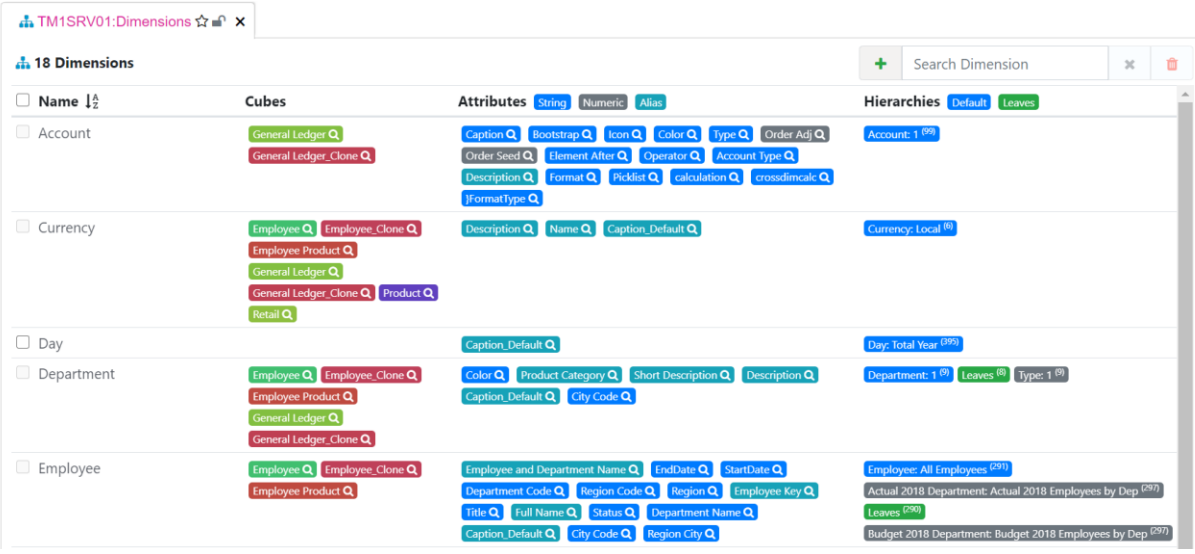
Dimensions summary page

How to access the dimensions summary page

To access the dimensions summary page, from the left menu, just click on the item **Dimensions**:



It will open a new tab containing the list of all dimensions in the instance **TM1SRV01**:

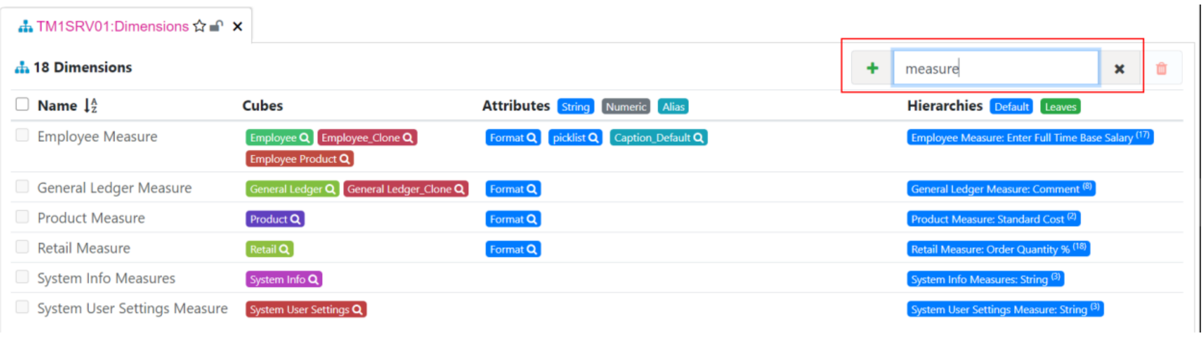


18 Dimensions

Name	Cubes	Attributes String Numeric Alias	Hierarchies Default Leaves
Account	General Ledger General Ledger_Clone	Caption, Bootstrap, Icon, Color, Type, Order Adj Order Seed, Element After, Operator, Account Type Description, Format, Picklist, calculation, crossdimcalc }FormatType	Account: 1 (99)
Currency	Employee, Employee_Clone Employee Product General Ledger General Ledger_Clone, Product Retail	Description, Name, Caption_Default	Currency: Local (6)
Day		Caption_Default	Day: Total Year (395)
Department	Employee, Employee_Clone Employee Product General Ledger General Ledger_Clone	Color, Product Category, Short Description, Description Caption_Default, City Code	Department: 1 (9), Leaves (8), Type: 1 (9)
Employee	Employee, Employee_Clone Employee Product	Employee and Department Name, EndDate, StartDate Department Code, Region Code, Region, Employee Key Title, Full Name, Status, Department Name Caption_Default, City Code, Region City	Employee: All Employees (291) Actual 2018 Department: Actual 2018 Employees by Dep (297) Leaves (290) Budget 2018 Department: Budget 2018 Employees by Dep (297)

Filter dimensions by text

In this tab you can use the search box from the top right to filter all dimensions containing the text **measure**:



TM1SRV01:Dimensions

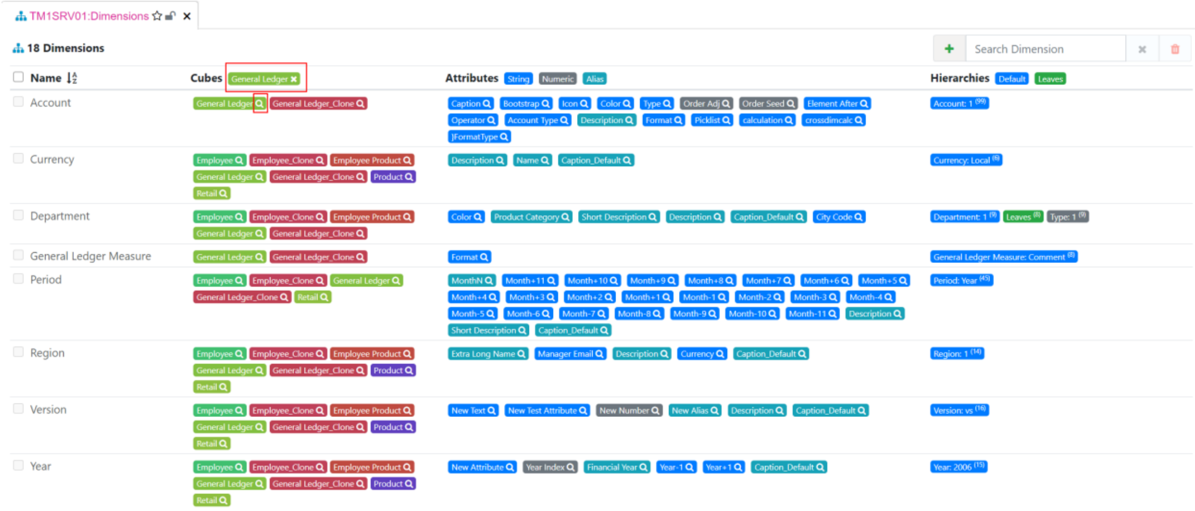
18 Dimensions

measure

□ Name	Cubes	Attributes (String, Numeric, Alias)	Hierarchies (Default, Leaves)
□ Employee Measure	Employee Employee_Clone Employee Product	Format picklist Caption_Default	Employee Measure: Enter Full Time Base Salary (17)
□ General Ledger Measure	General Ledger General Ledger_Clone	Format	General Ledger Measure: Comment (8)
□ Product Measure	Product	Format	Product Measure: Standard Cost (2)
□ Retail Measure	Retail	Format	Retail Measure: Order Quantity % (18)
□ System Info Measures	System Info		System Info Measures: String (3)
□ System User Settings Measure	System User Settings		System User Settings Measure: String (3)

Find dimensions by cube

To see all dimensions used by a cube, click the little search icon 🔍 of a cube you want to filter with. Below we can see the list of all dimensions in the Employee cube:



TM1SRV01:Dimensions

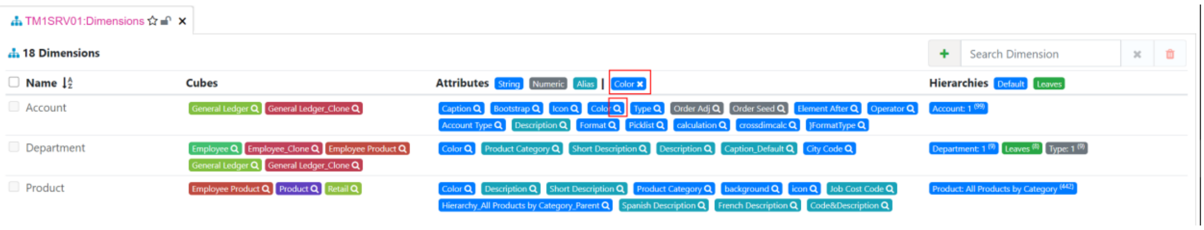
18 Dimensions

Search Dimension

□ Name	Cubes: General Ledger ✖	Attributes (String, Numeric, Alias)	Hierarchies (Default, Leaves)
□ Account	General Ledger General Ledger_Clone	Caption Bootstrap Icon Color Type Order Adj Order Seed Element After Operator Account Type Description Format Picklist calculation crossdimcalc }FormatType	Account: 1 (99)
□ Currency	Employee Employee_Clone Employee Product General Ledger General Ledger_Clone Product Retail	Description Name Caption_Default	Currency: Local (6)
□ Department	Employee Employee_Clone Employee Product General Ledger General Ledger_Clone	Color Product Category Short Description Description Caption_Default City Code	Department: 1 (9) Leaves (8) Type: 1 (9)
□ General Ledger Measure	General Ledger General Ledger_Clone	Format	General Ledger Measure: Comment (8)
□ Period	Employee Employee_Clone General Ledger General Ledger_Clone Retail	MonthN Month+11 Month+10 Month+9 Month+8 Month+7 Month+6 Month+5 Month+4 Month+3 Month+2 Month+1 Month-1 Month-2 Month-3 Month-4 Month-5 Month-6 Month-7 Month-8 Month-9 Month-10 Month-11 Description Short Description Caption_Default	Period: Year (45)
□ Region	Employee Employee_Clone Employee Product General Ledger General Ledger_Clone Product Retail	Extra Long Name Manager Email Description Currency Caption_Default	Region: 1 (14)
□ Version	Employee Employee_Clone Employee Product General Ledger General Ledger_Clone Product Retail	New Text New Test Attribute New Number New Alias Description Caption_Default	Version: vs (16)
□ Year	Employee Employee_Clone Employee Product General Ledger General Ledger_Clone Product Retail	New Attribute Year Index Financial Year Year-1 Year+1 Caption_Default	Year: 2006 (15)

Filter dimensions by parameter

The same feature can be used with parameters, click on one parameter search icon to filter the list of dimensions. For example, below we filter all dimensions with **Color** as one of their parameters:



TM1SRV01:Dimensions

18 Dimensions

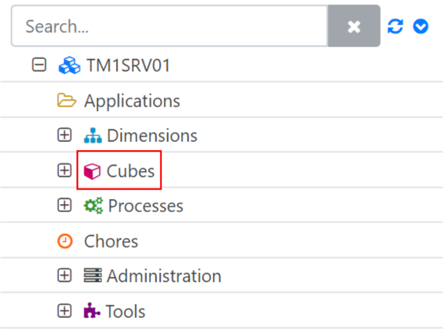
Search Dimension

□ Name	Cubes	Attributes (String, Numeric, Alias) | Color ✖	Hierarchies (Default, Leaves)
□ Account	General Ledger General Ledger_Clone	Caption Bootstrap Icon Color Type Order Adj Order Seed Element After Operator Account Type Description Format Picklist calculation crossdimcalc }FormatType	Account: 1 (99)
□ Department	Employee Employee_Clone Employee Product General Ledger General Ledger_Clone	Color Product Category Short Description Description Caption_Default City Code	Department: 1 (9) Leaves (8) Type: 1 (9)
□ Product	Employee Product Product Retail	Color Description Short Description Product Category background icon Job Cost Code Hierarchy_All Products by Category_Parent Spanish Description French Description Code&Description	Product: All Products by Category (442)

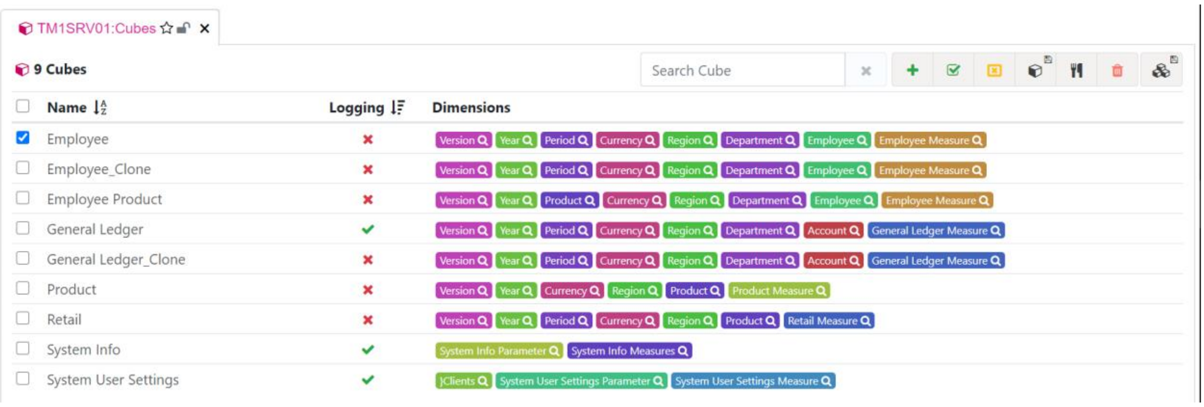
Cubes summary page

Accessing the Cube summary page

To access the cubes summary page, from the left menu, just click on the item **Cubes**:



This page contains lots of features, to active all buttons, you will need to select few cubes, for example you can tick the **Employee** and **General Ledger** cube.



☐	Name	Logging	Dimensions
☑	Employee	✗	Version, Year, Period, Currency, Region, Department, Employee, Employee Measure
☐	Employee_Clone	✗	Version, Year, Period, Currency, Region, Department, Employee, Employee Measure
☐	Employee Product	✗	Version, Year, Product, Currency, Region, Department, Employee, Employee Measure
☐	General Ledger	✓	Version, Year, Period, Currency, Region, Department, Account, General Ledger Measure
☐	General Ledger_Clone	✗	Version, Year, Period, Currency, Region, Department, Account, General Ledger Measure
☐	Product	✗	Version, Year, Currency, Region, Product, Product Measure
☐	Retail	✗	Version, Year, Period, Currency, Region, Product, Retail Measure
☐	System Info	✓	System Info Parameter, System Info Measures
☐	System User Settings	✓	}Clients, System User Settings Parameter, System User Settings Measure

There are lots of things you can do in this page, here the list:

- Create a new cube
- Activate logging*
- Deactivate Logging*
- Save Data for selected cubes
- Process Feeders for selected cubes
- Delete selected cubes
- Execute **SaveDataAll**

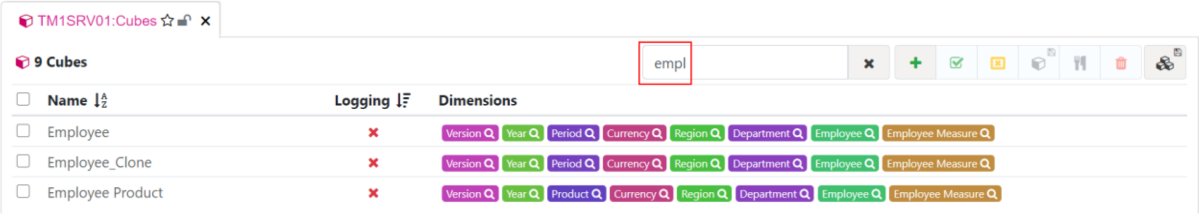
*When logging is turned on for a cube, TM1 will track all data changes. When one cell is updated, TM1 will store the previous value and the new value. This is very useful when users input values but when a cube is updated by a process, you don't wantS TM1 to store all values updated by the process, that is why it is recommended to deactive the logging when the process run and turned it back on when it finishes.

Select Employee and General Ledger and do the following:

1. Deactivate logging for these two cubes
2. Execute Save Data
3. Execute CubeProcessFeeders

Filter cubes by text

In this page you can also use the **Search** box to filter the list by text as below:



TM1SRV01:Cubes ☆ ✕

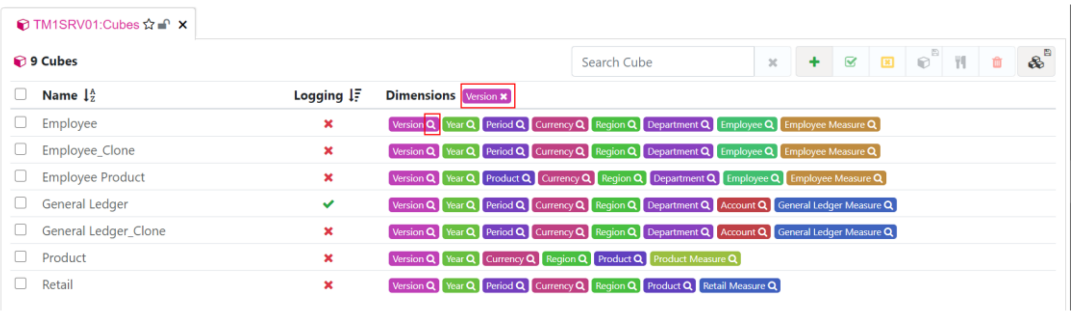
9 Cubes

empl ✖

☐	Name	Logging	Dimensions
☐	Employee	✖	Version Year Period Currency Region Department Employee Employee Measure
☐	Employee_Clone	✖	Version Year Period Currency Region Department Employee Employee Measure
☐	Employee Product	✖	Version Year Product Currency Region Department Employee Employee Measure

Filter cubes by dimension

You can click on the search icon for a specific dimension to filter the list of cubes. In the example below, the search icon on the **Version** dimension was clicked to show all cubes with a **Version** dimension:



TM1SRV01:Cubes ☆ ✕

9 Cubes

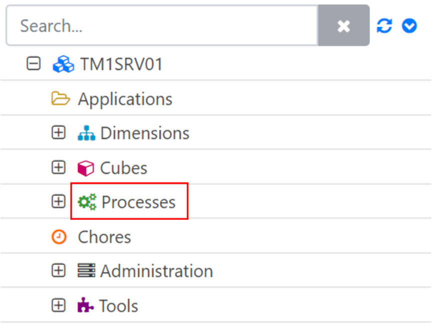
Search Cube ✖

☐	Name	Logging	Dimensions Version ✖
☐	Employee	✖	Version Year Period Currency Region Department Employee Employee Measure
☐	Employee_Clone	✖	Version Year Period Currency Region Department Employee Employee Measure
☐	Employee Product	✖	Version Year Product Currency Region Department Employee Employee Measure
☐	General Ledger	✓	Version Year Period Currency Region Department Account General Ledger Measure
☐	General Ledger_Clone	✖	Version Year Period Currency Region Department Account General Ledger Measure
☐	Product	✖	Version Year Currency Region Product Product Measure
☐	Retail	✖	Version Year Period Currency Region Product Retail Measure

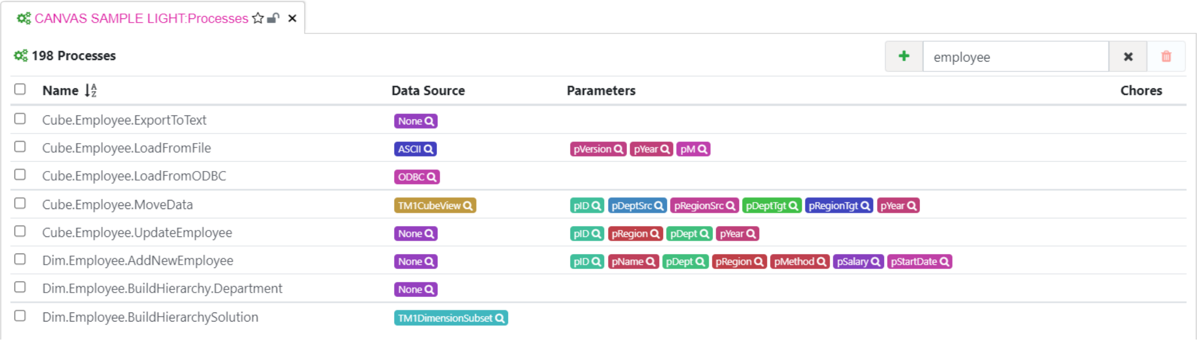
Processes summary page

How to access the processes page

To access the processes summary page, just click the **Processes** item from the menu:



It will show the list of all processes, you can then use the search box to filter the list of processes by text:



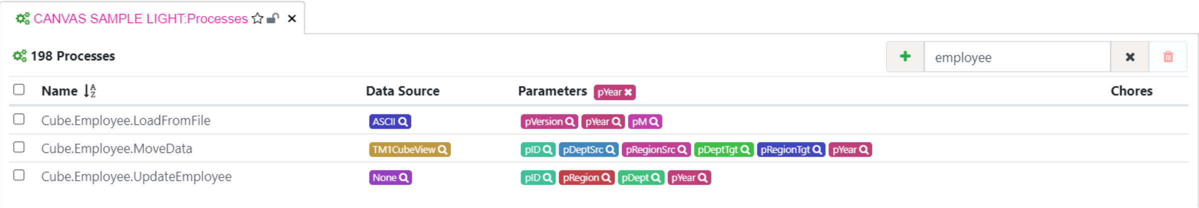
CANVAS SAMPLE LIGHT:Processes

198 Processes

employee

☐ Name	Data Source	Parameters	Chores
☐ Cube.Employee.ExportToText	None		
☐ Cube.Employee.LoadFromFile	ASCII	pVersion pYear pM	
☐ Cube.Employee.LoadFromODBC	ODBC		
☐ Cube.Employee.MoveData	TM1CubeView	pID pDeptSrc pRegionSrc pDeptTgt pRegionTgt pYear	
☐ Cube.Employee.UpdateEmployee	None	pID pRegion pDept pYear	
☐ Dim.Employee.AddNewEmployee	None	pID pName pDept pRegion pMethod pSalary pStartDate	
☐ Dim.Employee.BuildHierarchy.Department	None		
☐ Dim.Employee.BuildHierarchySolution	TM1DimensionSubset		

You can filter by parameters, just click the search icon for a parameter as below:



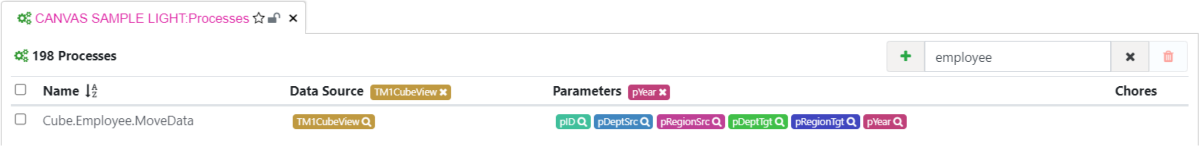
CANVAS SAMPLE LIGHT:Processes

198 Processes

employee

☐ Name	Data Source	Parameters pYear ✖	Chores
☐ Cube.Employee.LoadFromFile	ASCII	pVersion pYear pM	
☐ Cube.Employee.MoveData	TM1CubeView	pID pDeptSrc pRegionSrc pDeptTgt pRegionTgt pYear	
☐ Cube.Employee.UpdateEmployee	None	pID pRegion pDept pYear	

Finally, you can also filter by Data Source type by clicking again on the search icon:



CANVAS SAMPLE LIGHT:Processes

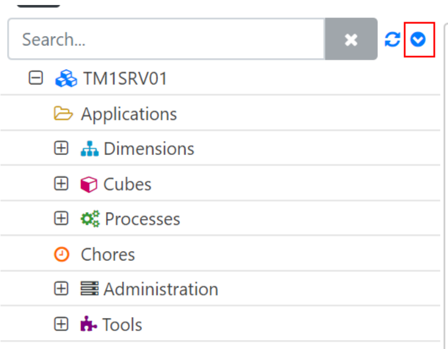
198 Processes

employee

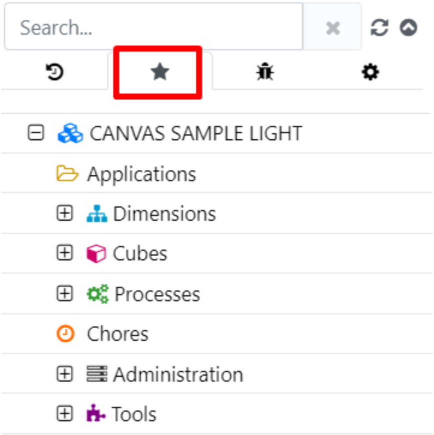
☐ Name	Data Source TM1CubeView ✖	Parameters pYear ✖	Chores
☐ Cube.Employee.MoveData	TM1CubeView	pID pDeptSrc pRegionSrc pDeptTgt pRegionTgt pYear	

Add to favourites

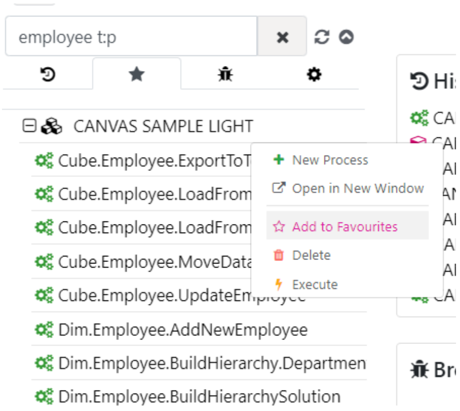
Arc enables you to group what you are working on as favourites. Adding your current development priorities to the Favourites tab helps you stay focused and quickly resume your workflow after an interruption, to access the **Favourites** tab, you will need first to open the **Options** as below:



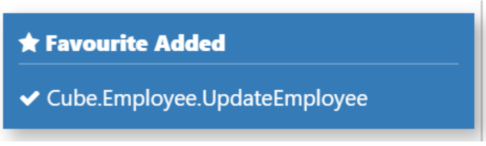
And then click the star icon



Currently our favourites are empty, let's add one process. We want to add all processes containing the string **Employee** as favourites. To do that let's first search the menu with **employee t:p**, right-click on one process and the context menu will appear and click the **Add to Favourites** for each process:



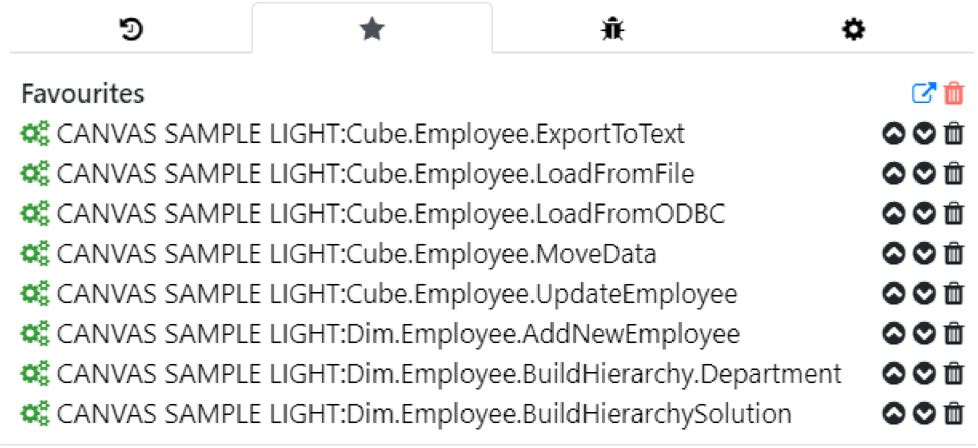
One an object has been added, you will see a notification on the bottom left of your screen:



Favourite Added

✓ Cube.Employee.UpdateEmployee

Once all Employee processes have been added, you can easily remove them by clicking the trash icon 🗑 :



Favourites

CANVAS SAMPLE LIGHT:Cube.Employee.ExportToText
CANVAS SAMPLE LIGHT:Cube.Employee.LoadFromFile
CANVAS SAMPLE LIGHT:Cube.Employee.LoadFromODBC
CANVAS SAMPLE LIGHT:Cube.Employee.MoveData
CANVAS SAMPLE LIGHT:Cube.Employee.UpdateEmployee
CANVAS SAMPLE LIGHT:Dim.Employee.AddNewEmployee
CANVAS SAMPLE LIGHT:Dim.Employee.BuildHierarchy.Department
CANVAS SAMPLE LIGHT:Dim.Employee.BuildHierarchySolution

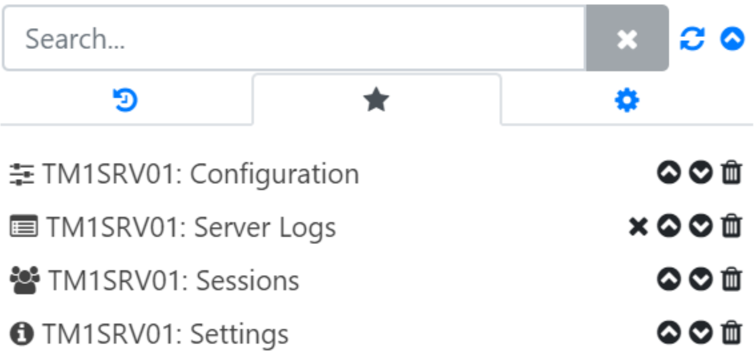
TOPIC 5: GET A BETTER INSIGHT OF YOUR SERVER

Objectives

In this section, students will learn:

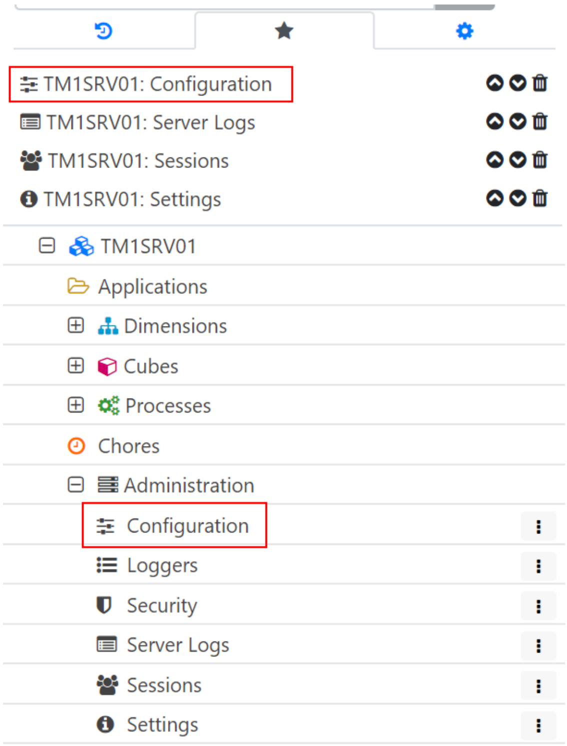
- How to find the TM1 version
- How to update TM1 configuration parameters
- How to filter tm1server.log
- How to cancel a long running thread

Let’s add to favourites the following administration tools, so it will be quicker to access them:

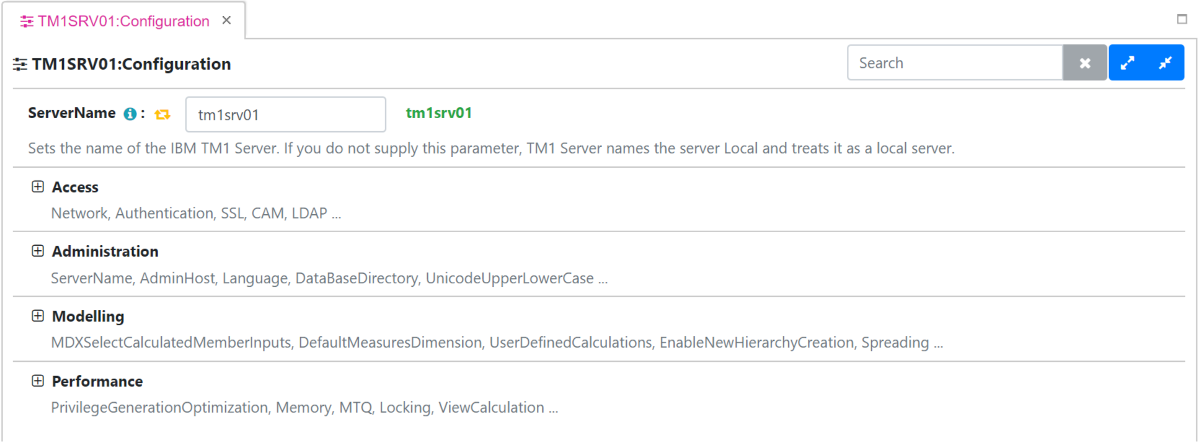


Configuration - Update all TM1 server parameters:

Let’s open first the **Configuration** tool, just click the **Configuration** item from the left menu or the one from the favourites:



Once opened, you will be able to see all TM1 server parameters:



TM1SRV01:Configuration ×

TM1SRV01:Configuration

Search

ServerName : tm1srv01 tm1srv01

Sets the name of the IBM TM1 Server. If you do not supply this parameter, TM1 Server names the server Local and treats it as a local server.

⊞ Access

Network, Authentication, SSL, CAM, LDAP ...

⊞ Administration

ServerName, AdminHost, Language, DataBaseDirectory, UnicodeUpperLowerCase ...

⊞ Modelling

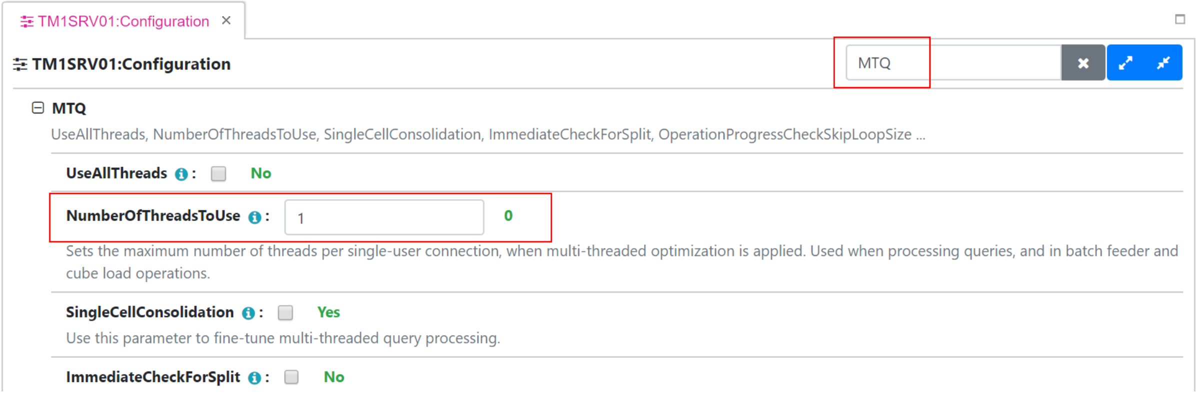
MDXSelectCalculatedMemberInputs, DefaultMeasuresDimension, UserDefinedCalculations, EnableNewHierarchyCreation, Spreading ...

⊞ Performance

PrivilegeGenerationOptimization, Memory, MTQ, Locking, ViewCalculation ...

As you can see the TM1 server has lots of parameters, you can then use the Search box to filter the list.

Our objective is to update the value of the **MTQ** (MultiThreadQuery) parameter, let’s type **MTQ** in the top right search box and expand the **MTQ**, you should see all parameters linked with MTQ as below:



TM1SRV01:Configuration ×

TM1SRV01:Configuration

MTQ

⊟ MTQ

UseAllThreads, NumberOfThreadsToUse, SingleCellConsolidation, ImmediateCheckForSplit, OperationProgressCheckSkipLoopSize ...

UseAllThreads : □ No

NumberOfThreadsToUse : 1 0

Sets the maximum number of threads per single-user connection, when multi-threaded optimization is applied. Used when processing queries, and in batch feeder and cube load operations.

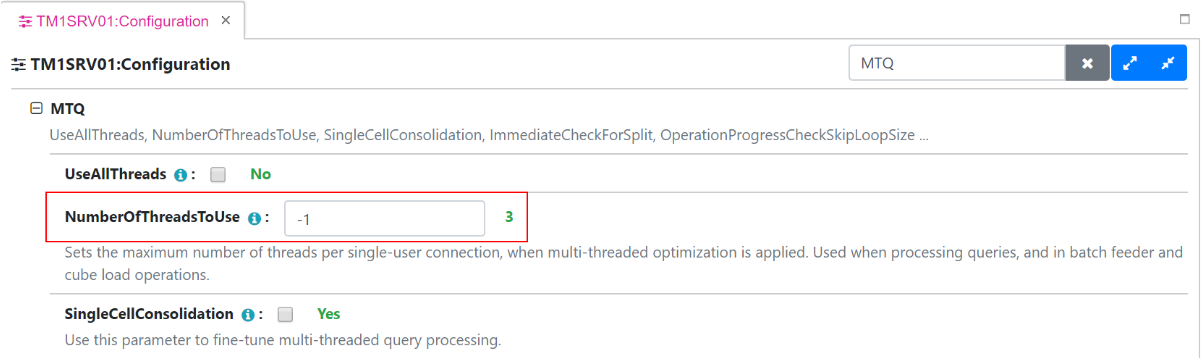
SingleCellConsolidation : □ Yes

Use this parameter to fine-tune multi-threaded query processing.

ImmediateCheckForSplit : □ No

The **NumberOfThreadsToUse** is set to 1 and the value used by TM1 is 0 because TM1 subtract one to this value so the final value is 0. The recommended setting is to set this value to **-1** (By setting this value to **-1**, TM1 will take the number of cores and subtract 1).

Replace 1 with -1 and you will see that the new value is now **3**, because in this example the server has 4 cores:



TM1SRV01:Configuration ×

TM1SRV01:Configuration

MTQ

⊟ MTQ

UseAllThreads, NumberOfThreadsToUse, SingleCellConsolidation, ImmediateCheckForSplit, OperationProgressCheckSkipLoopSize ...

UseAllThreads : □ No

NumberOfThreadsToUse : -1 3

Sets the maximum number of threads per single-user connection, when multi-threaded optimization is applied. Used when processing queries, and in batch feeder and cube load operations.

SingleCellConsolidation : □ Yes

Use this parameter to fine-tune multi-threaded query processing.

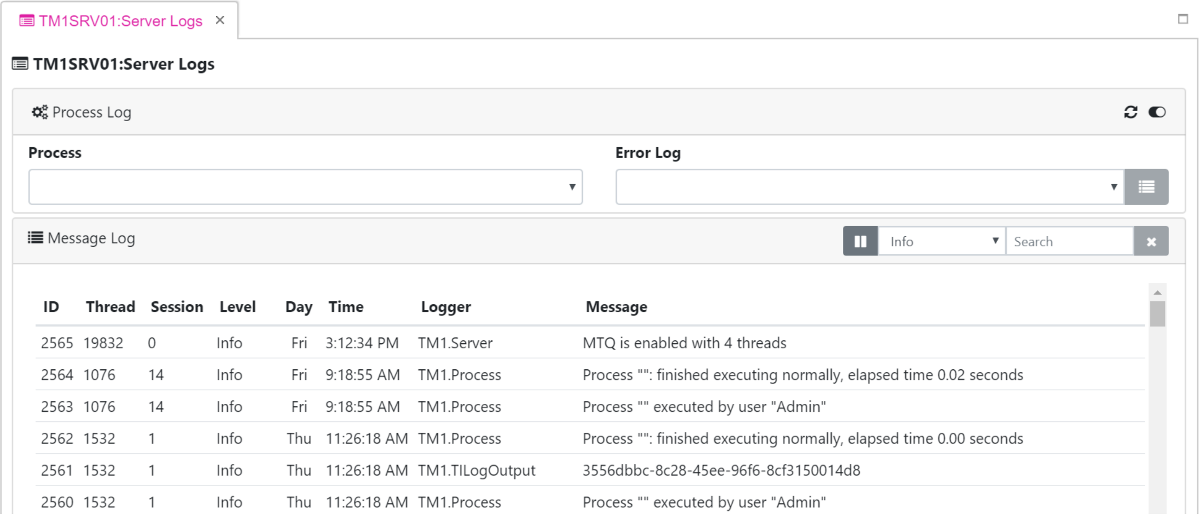
If the server had 8 cores, the value would have been 7.

Server logs – find faster errors in the tm1server.log

Arc includes a tool called **Server Logs** which will enable you to filter the logs.

Let's open it, by clicking the **Server Logs**, either from the favourites or from the **Administration** list. In the first panel, you will find a dropdown with all the processes which ran unsuccessfully. In **Error Log** dropdown, you will find the list of error file for the selected process.

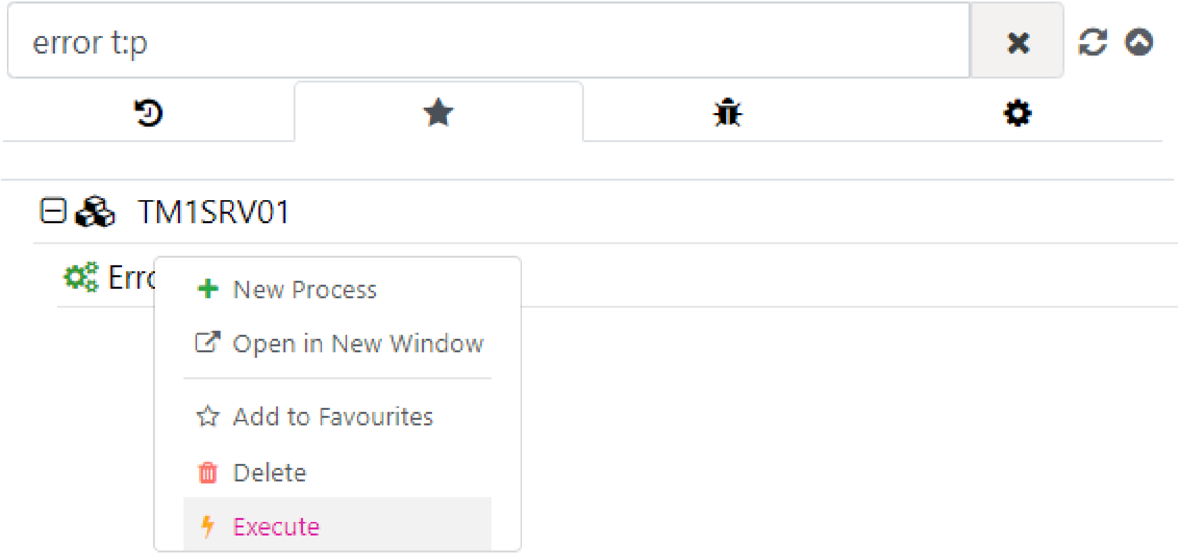
In the second panel you will find the messages from the **tm1server.log** file:



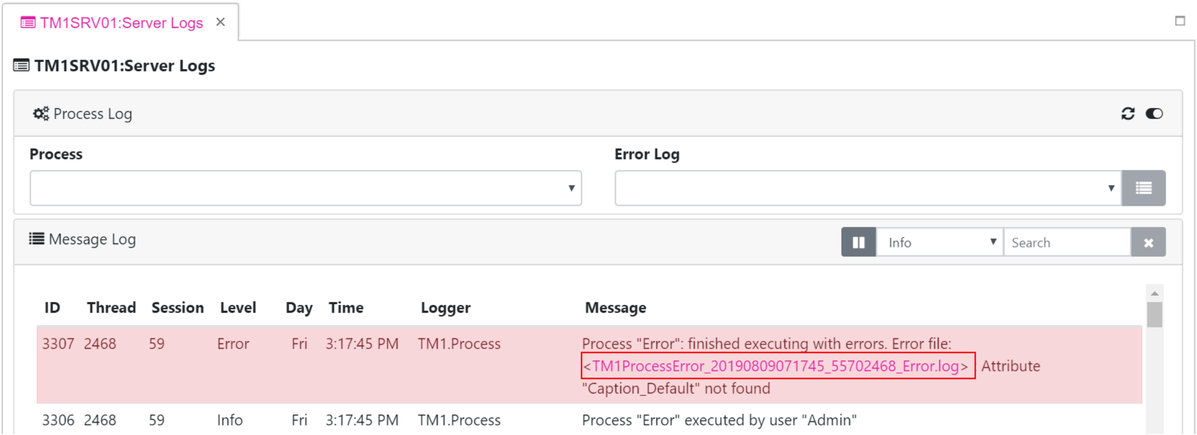
ID	Thread	Session	Level	Day	Time	Logger	Message
2565	19832	0	Info	Fri	3:12:34 PM	TM1.Server	MTQ is enabled with 4 threads
2564	1076	14	Info	Fri	9:18:55 AM	TM1.Process	Process "": finished executing normally, elapsed time 0.02 seconds
2563	1076	14	Info	Fri	9:18:55 AM	TM1.Process	Process "" executed by user "Admin"
2562	1532	1	Info	Thu	11:26:18 AM	TM1.Process	Process "": finished executing normally, elapsed time 0.00 seconds
2561	1532	1	Info	Thu	11:26:18 AM	TM1.TILogOutput	3556dbbc-8c28-45ee-96f6-8cf3150014d8
2560	1532	1	Info	Thu	11:26:18 AM	TM1.Process	Process "" executed by user "Admin"

Currently in the message log, there are only Info messages. Let's run a process which generates errors in order to see different type of messages.

To see some errors, let's execute the process called **Error**, to find it you can type in the search box **error t:p** then right click on the process name and click Execute from the context menu:

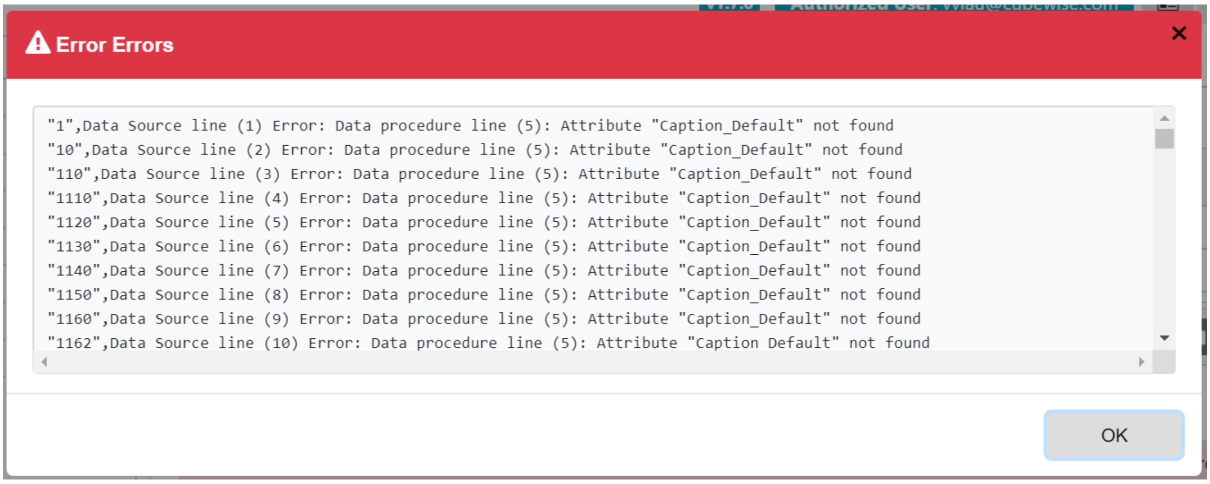


In the **Server** Logs, you should be able to see an **Error** level message. For each error, Arc will give you a link to the log file, just click on the link as below:



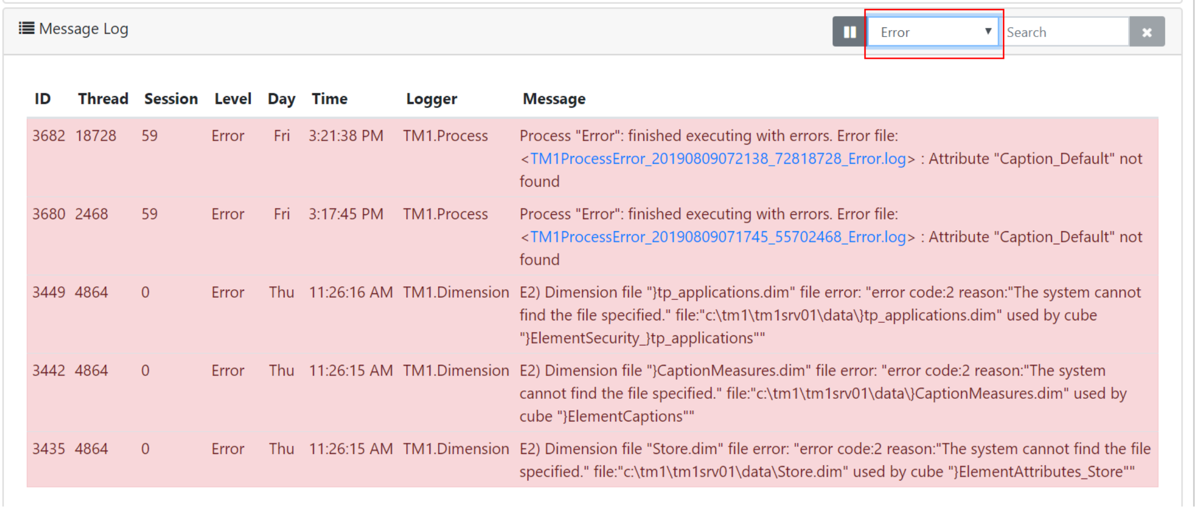
ID	Thread	Session	Level	Day	Time	Logger	Message
3307	2468	59	Error	Fri	3:17:45 PM	TM1.Process	Process "Error": finished executing with errors. Error file: <TM1ProcessError_20190809071745_55702468_Error.log> Attribute "Caption_Default" not found
3306	2468	59	Info	Fri	3:17:45 PM	TM1.Process	Process "Error" executed by user "Admin"

After clicking on the link, a new window will pop-up with the content of the Error message



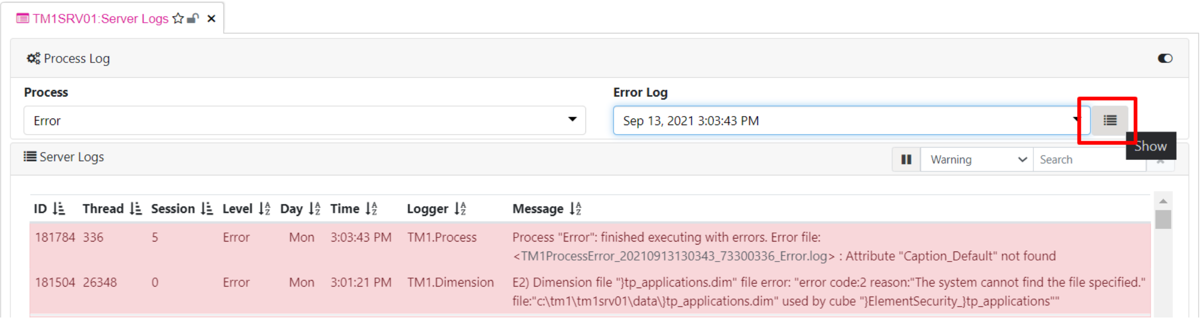
```
"1",Data Source line (1) Error: Data procedure line (5): Attribute "Caption_Default" not found
"10",Data Source line (2) Error: Data procedure line (5): Attribute "Caption_Default" not found
"110",Data Source line (3) Error: Data procedure line (5): Attribute "Caption_Default" not found
"1110",Data Source line (4) Error: Data procedure line (5): Attribute "Caption_Default" not found
"1120",Data Source line (5) Error: Data procedure line (5): Attribute "Caption_Default" not found
"1130",Data Source line (6) Error: Data procedure line (5): Attribute "Caption_Default" not found
"1140",Data Source line (7) Error: Data procedure line (5): Attribute "Caption_Default" not found
"1150",Data Source line (8) Error: Data procedure line (5): Attribute "Caption_Default" not found
"1160",Data Source line (9) Error: Data procedure line (5): Attribute "Caption_Default" not found
"1162",Data Source line (10) Error: Data procedure line (5): Attribute "Caption Default" not found
```

Let's run a second time the **Error** process to get more error messages in our Server Log. When there are lots of messages, it can be handy to filter the list by error type. For example to see only the **Level Error** messages, you can use the dropdown close to the search box and select **Error,** it is going to show you only the message with Level = Error:



ID	Thread	Session	Level	Day	Time	Logger	Message
3682	18728	59	Error	Fri	3:21:38 PM	TM1.Process	Process "Error": finished executing with errors. Error file: <TM1ProcessError_20190809072138_72818728_Error.log> : Attribute "Caption_Default" not found
3680	2468	59	Error	Fri	3:17:45 PM	TM1.Process	Process "Error": finished executing with errors. Error file: <TM1ProcessError_20190809071745_55702468_Error.log> : Attribute "Caption_Default" not found
3449	4864	0	Error	Thu	11:26:16 AM	TM1.Dimension	E2) Dimension file "}tp_applications.dim" file error: "error code:2 reason:"The system cannot find the file specified." file:"c:\tm1\tm1srv01\data\}tp_applications.dim" used by cube "}ElementSecurity_}tp_applications""
3442	4864	0	Error	Thu	11:26:15 AM	TM1.Dimension	E2) Dimension file "}CaptionMeasures.dim" file error: "error code:2 reason:"The system cannot find the file specified." file:"c:\tm1\tm1srv01\data\}CaptionMeasures.dim" used by cube "}ElementCaptions""
3435	4864	0	Error	Thu	11:26:15 AM	TM1.Dimension	E2) Dimension file "Store.dim" file error: "error code:2 reason:"The system cannot find the file specified." file:"c:\tm1\tm1srv01\data\Store.dim" used by cube "}ElementAttributes_Store""

If you want to see all errors for a specific process, you can select the process from the process dropdown as below and then select the log message you want to see:

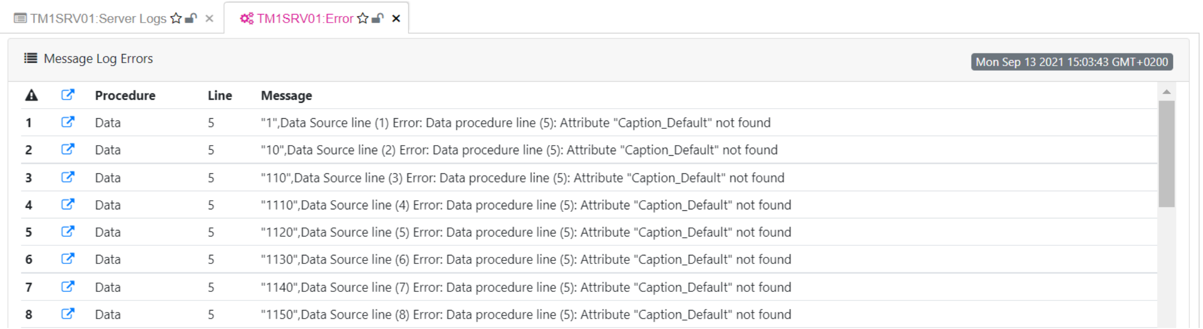


Process: Error

Error Log: Sep 13, 2021 3:03:43 PM

ID	Thread	Session	Level	Day	Time	Logger	Message
181784	336	5	Error	Mon	3:03:43 PM	TM1.Process	Process "Error": finished executing with errors. Error file: <TM1ProcessError_20210913130343_73300336_Error.log> : Attribute "Caption_Default" not found
181504	26348	0	Error	Mon	3:01:21 PM	TM1.Dimension	E2) Dimension file "}tp_applications.dim" file error: "error code:2 reason:"The system cannot find the file specified." file:"c:\tm1\tm1srv01\data\}tp_applications.dim" used by cube "}ElementSecurity_}tp_applications""

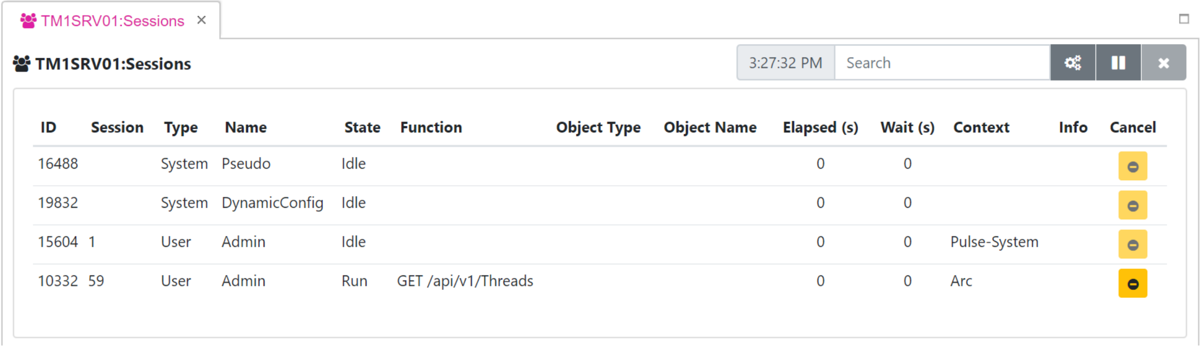
If you can’t see the latest messages, just refresh the browser. Once the Error Log selected, the content of the message will open in a new window:



⚠		Procedure	Line	Message
1	↗	Data	5	"1",Data Source line (1) Error: Data procedure line (5): Attribute "Caption_Default" not found
2	↗	Data	5	"10",Data Source line (2) Error: Data procedure line (5): Attribute "Caption_Default" not found
3	↗	Data	5	"110",Data Source line (3) Error: Data procedure line (5): Attribute "Caption_Default" not found
4	↗	Data	5	"1110",Data Source line (4) Error: Data procedure line (5): Attribute "Caption_Default" not found
5	↗	Data	5	"1120",Data Source line (5) Error: Data procedure line (5): Attribute "Caption_Default" not found
6	↗	Data	5	"1130",Data Source line (6) Error: Data procedure line (5): Attribute "Caption_Default" not found
7	↗	Data	5	"1140",Data Source line (7) Error: Data procedure line (5): Attribute "Caption_Default" not found
8	↗	Data	5	"1150",Data Source line (8) Error: Data procedure line (5): Attribute "Caption_Default" not found

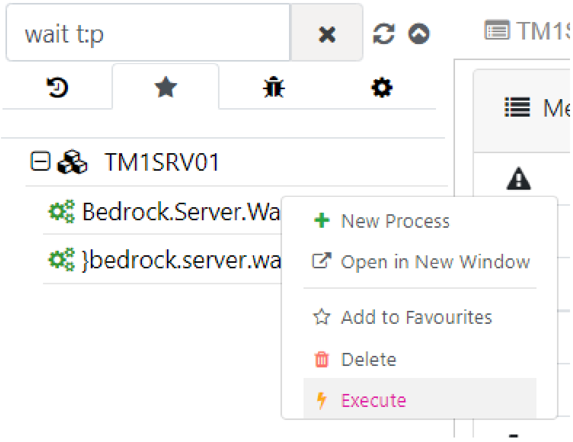
Sessions

Let’s have a look now at who is logged in to your TM1 instance, Arc comes with a **Sessions** tool to see everyone who is currently logged in:

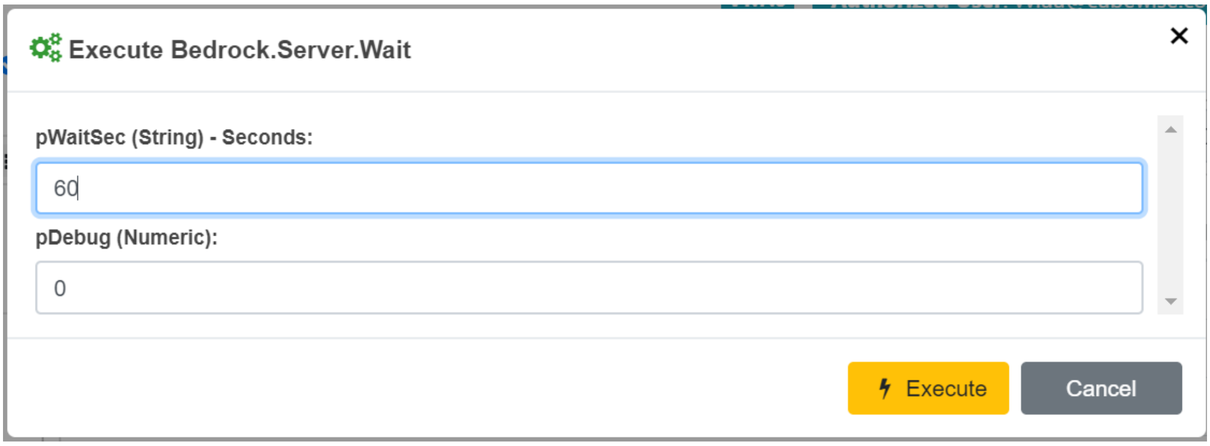


ID	Session	Type	Name	State	Function	Object Type	Object Name	Elapsed (s)	Wait (s)	Context	Info	Cancel
16488		System	Pseudo	Idle				0	0			⊖
19832		System	DynamicConfig	Idle				0	0			⊖
15604	1	User	Admin	Idle				0	0	Pulse-System		⊖
10332	59	User	Admin	Run	GET /api/v1/Threads			0	0	Arc		⊖

Let’s execute **Bedrock.Server.Wait**:



Set the **pWaitSec** value to 60 seconds and click execute:

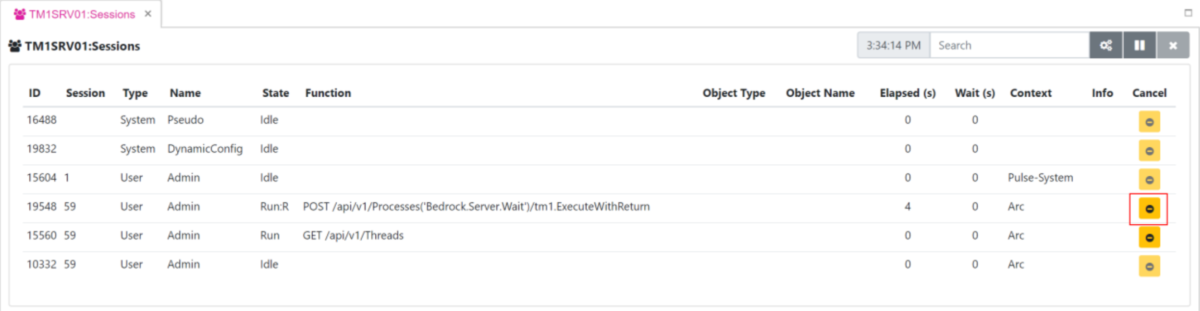


Execute Bedrock.Server.Wait

pWaitSec (String) - Seconds: 60

pDebug (Numeric): 0

From the **Sessions**, you will be able to see the process running, you can also click the Cancel button to stop the process.

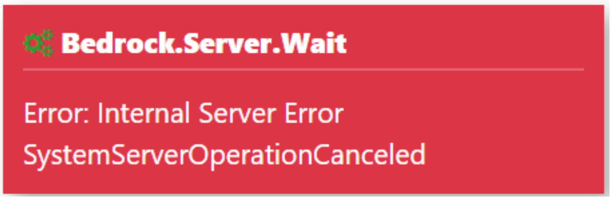


TM1SRV01:Sessions

3:34:14 PM

ID	Session	Type	Name	State	Function	Object Type	Object Name	Elapsed (s)	Wait (s)	Context	Info	Cancel
16488		System	Pseudo	Idle				0	0			⊖
19832		System	DynamicConfig	Idle				0	0			⊖
15604	1	User	Admin	Idle				0	0	Pulse-System		⊖
19548	59	User	Admin	Run:R	POST /api/v1/Processes('Bedrock.Server.Wait')/tm1.ExecuteWithReturn			4	0	Arc		⊖
15560	59	User	Admin	Run	GET /api/v1/Threads			0	0	Arc		⊖
10332	59	User	Admin	Idle				0	0	Arc		⊖

Once the process has been cancelled, a red alert will pop-up from the bottom left of your screen:

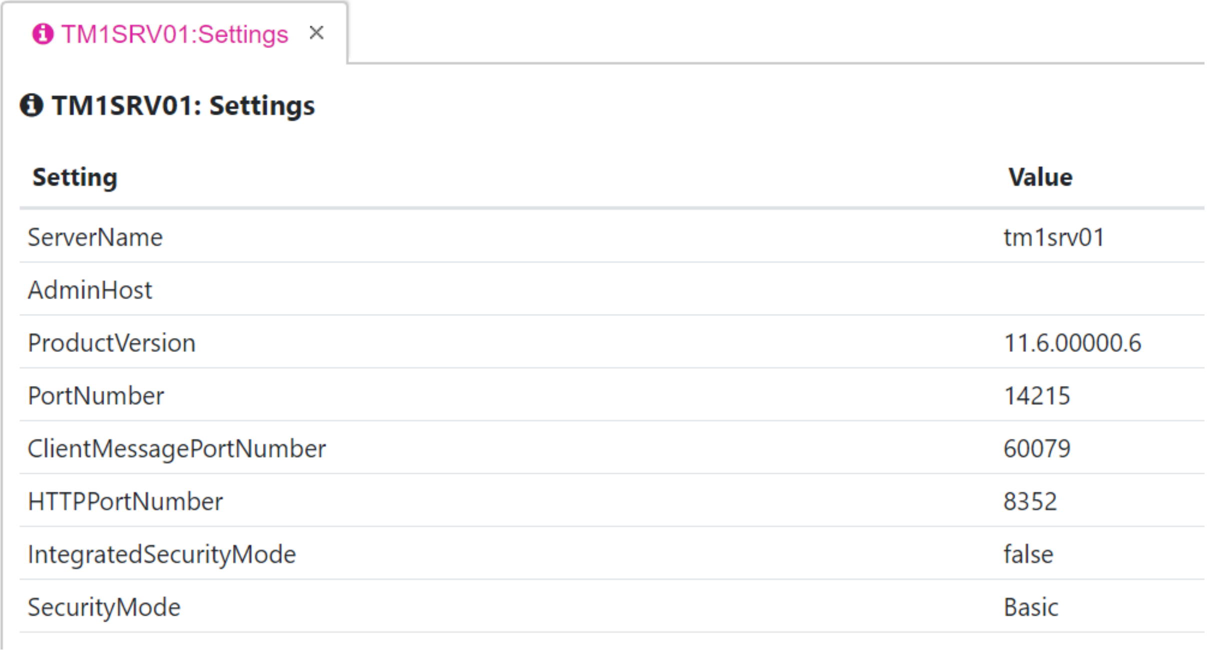


Bedrock.Server.Wait

Error: Internal Server Error SystemServerOperationCanceled

Settings

The **Settings** tool can be used to have a quick overview of the most important TM1 server settings. For example, you can use it to find the TM1 version of the TM1 server. In the list it is the **ProductVersion:**



TM1SRV01: Settings

Setting	Value
ServerName	tm1srv01
AdminHost	
ProductVersion	11.6.00000.6
PortNumber	14215
ClientMessagePortNumber	60079
HTTPPortNumber	8352
IntegratedSecurityMode	false
SecurityMode	Basic

TOPIC 6: LEVERAGE ARC TOOLS TO SPEED UP DEVELOPMENTS

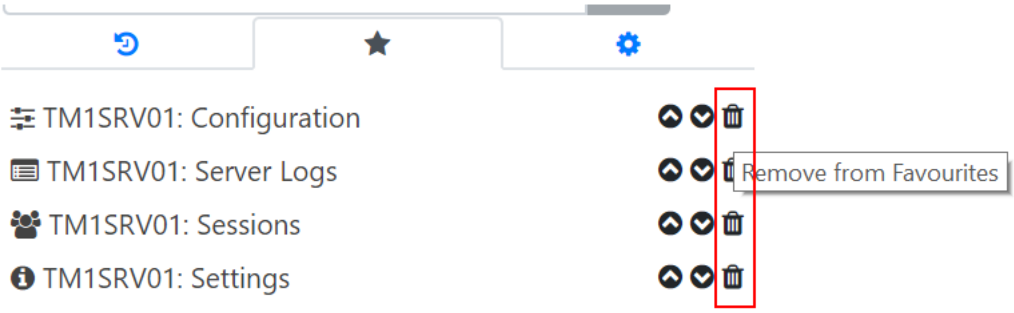
Objectives

In this section, students will learn:

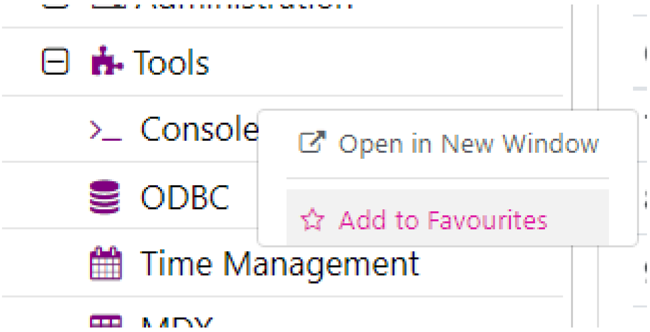
- Using Arc Console to execute TI functions
- Finding which subsets is used by which views
- Testing your MDX queries
- Running your first REST API queries

In this section we are going to focus on the Arc tools, let's clean the favourites tabs and add the tools from the **Tools** menu.

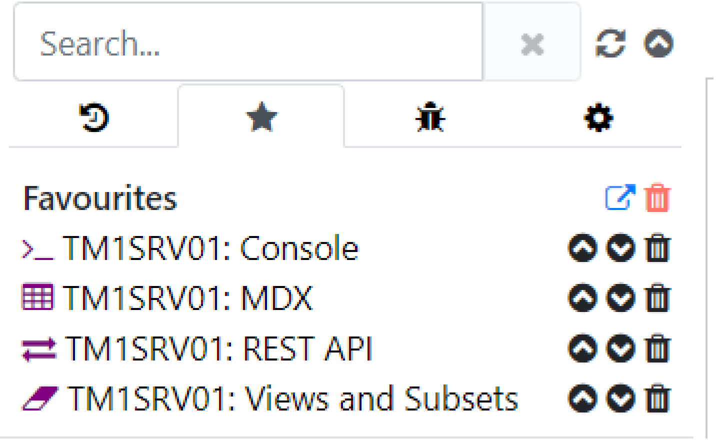
To remove items from the favourites list, just click the trash icon:



Now let's add all the tools:

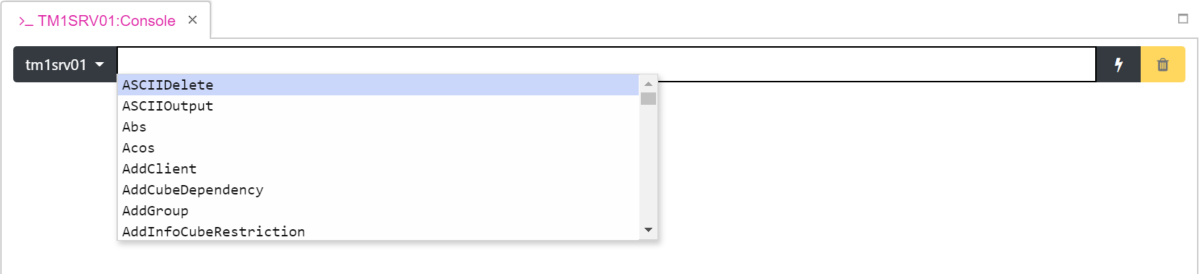


Your favourites list should look like this now:

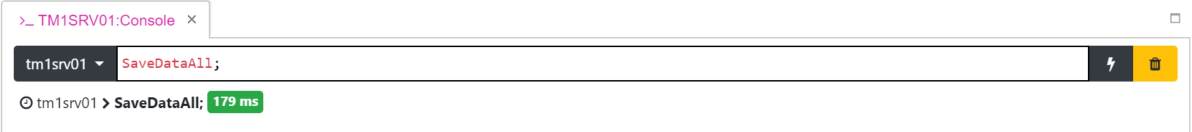


Speed-up development with Arc Console

Let's open the **Console**, from the **Console** you can execute any TM1 process functions, hit **CTRL+SPACE** to show the list of available TI functions:



Let's first execute a SaveDataAll; and press enter to execute the function, Arc will show you how long it took to execute this function:

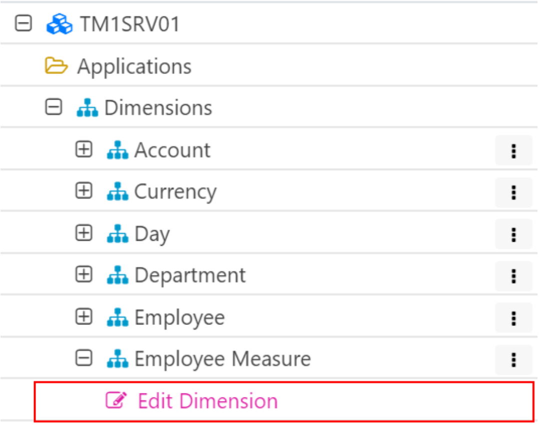


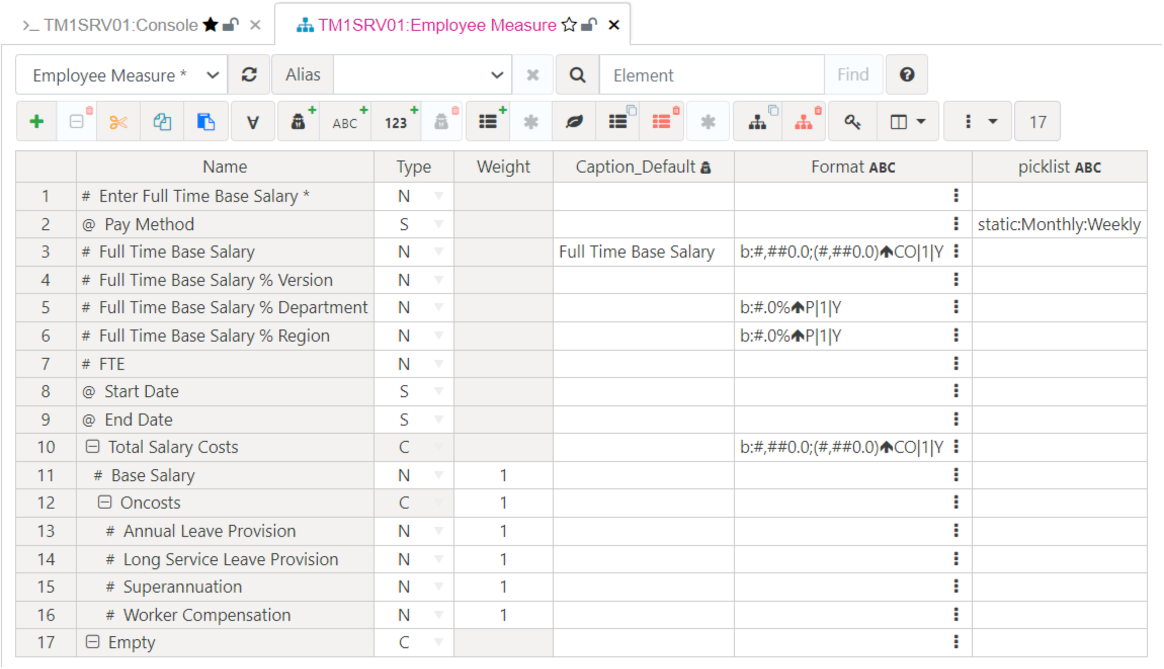
```
SaveDataAll;
```

```
tm1srv01 > SaveDataAll; 179 ms
```

Next we want to add a new element **Leaves** to the **Employee Measure** dimension without opening the dimension editor

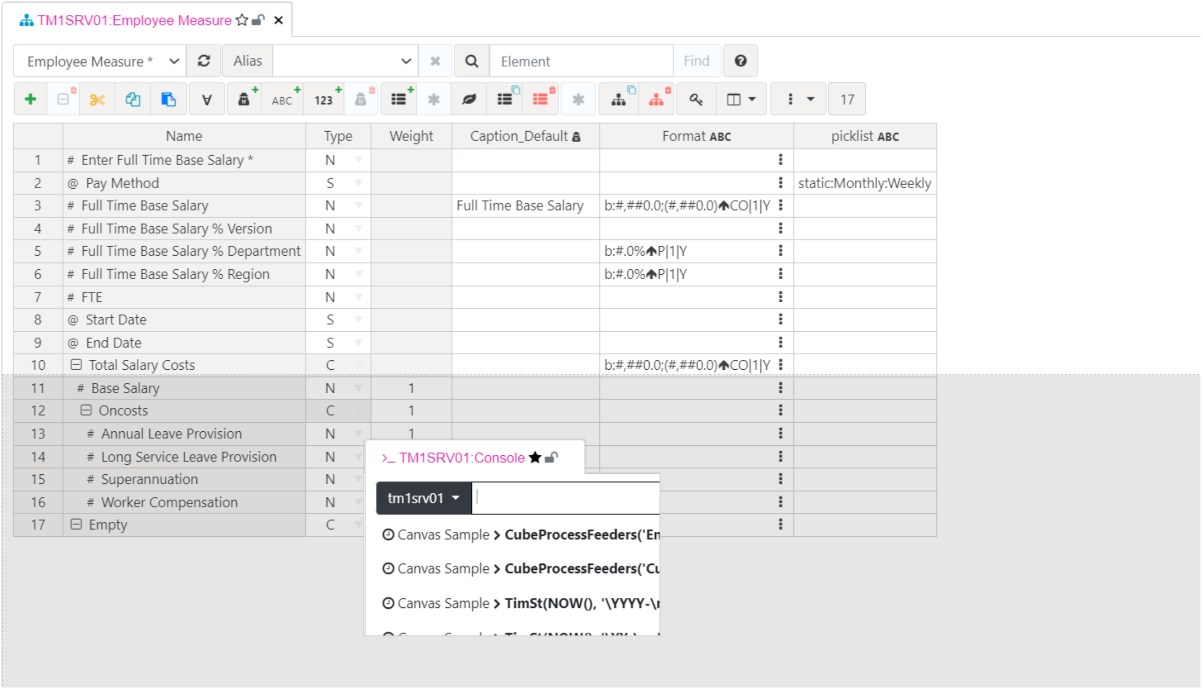
First let's open the Employee Measure dimension editor:





	Name	Type	Weight	Caption_Default	Format ABC	picklist ABC
1	# Enter Full Time Base Salary *	N				
2	@ Pay Method	S				static:Monthly:Weekly
3	# Full Time Base Salary	N		Full Time Base Salary	b:#,##0.0;(#,##0.0)▲CO|1|Y	
4	# Full Time Base Salary % Version	N				
5	# Full Time Base Salary % Department	N			b:#.0%▲P|1|Y	
6	# Full Time Base Salary % Region	N			b:#.0%▲P|1|Y	
7	# FTE	N				
8	@ Start Date	S				
9	@ End Date	S				
10	⊟ Total Salary Costs	C			b:#,##0.0;(#,##0.0)▲CO|1|Y	
11	# Base Salary	N	1			
12	⊟ Oncosts	C	1			
13	# Annual Leave Provision	N	1			
14	# Long Service Leave Provision	N	1			
15	# Superannuation	N	1			
16	# Worker Compensation	N	1			
17	⊟ Empty	C				

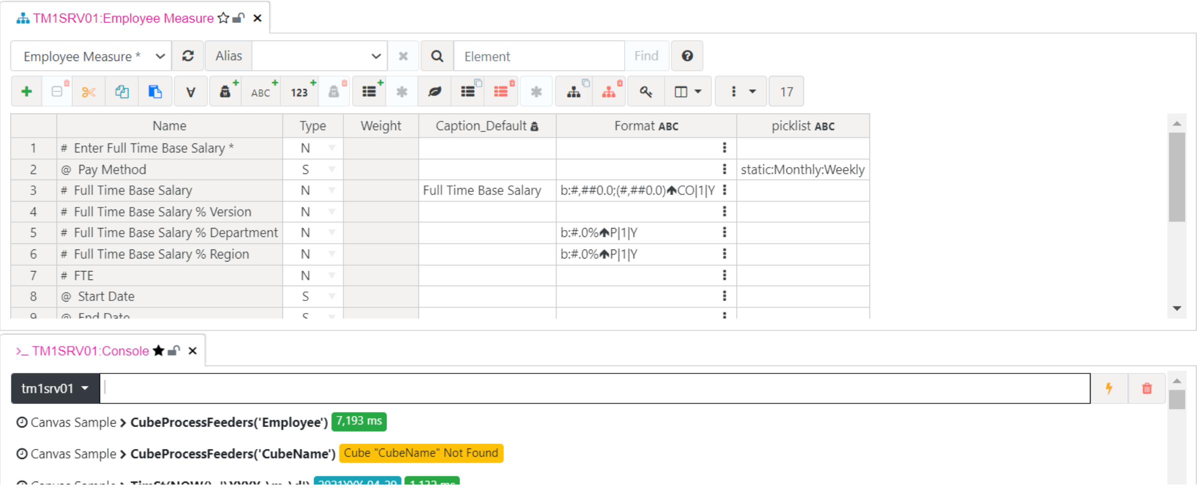
Let’s move the Arc Console at the bottom, just drag and drop the Console tab to the bottom, once you see the grey area as below just release it:



	Name	Type	Weight	Caption_Default	Format ABC	picklist ABC
1	# Enter Full Time Base Salary *	N				
2	@ Pay Method	S				static:Monthly:Weekly
3	# Full Time Base Salary	N		Full Time Base Salary	b:#,##0.0;(#,##0.0)▲CO|1|Y	
4	# Full Time Base Salary % Version	N				
5	# Full Time Base Salary % Department	N			b:#.0%▲P|1|Y	
6	# Full Time Base Salary % Region	N			b:#.0%▲P|1|Y	
7	# FTE	N				
8	@ Start Date	S				
9	@ End Date	S				
10	⊟ Total Salary Costs	C			b:#,##0.0;(#,##0.0)▲CO|1|Y	
11	# Base Salary	N	1			
12	⊟ Oncosts	C	1			
13	# Annual Leave Provision	N	1			
14	# Long Service Leave Provision	N				
15	# Superannuation	N				
16	# Worker Compensation	N				
17	⊟ Empty	C				

```
Canvas Sample > CubeProcessFeeders('En
Canvas Sample > CubeProcessFeeders('Cu
Canvas Sample > TimSt(NOW(), '\YYYY-\
```

You should see this your two tabs as below:



TM1SRV01:Employee Measure

Employee Measure * | Alias | Element | Find | 17

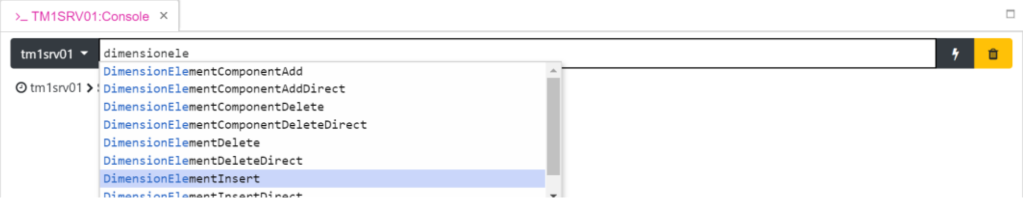
	Name	Type	Weight	Caption_Default	Format ABC	picklist ABC
1	# Enter Full Time Base Salary *	N				
2	@ Pay Method	S				static:Monthly:Weekly
3	# Full Time Base Salary	N		Full Time Base Salary	b:#,##0.0;(#,##0.0)CO|1|Y	
4	# Full Time Base Salary % Version	N				
5	# Full Time Base Salary % Department	N			b:#.0%P|1|Y	
6	# Full Time Base Salary % Region	N			b:#.0%P|1|Y	
7	# FTE	N				
8	@ Start Date	S				
9	@ End Date	S				

TM1SRV01:Console

tm1srv01

```
Canvas Sample > CubeProcessFeeders('Employee') 7,193 ms
Canvas Sample > CubeProcessFeeders('CubeName') Cube "CubeName" Not Found
```

Start typing **DimensionEle** and hit **CTRL+SPACE** to open the snippets and then select **DimensionElementInsert** function:



TM1SRV01:Console

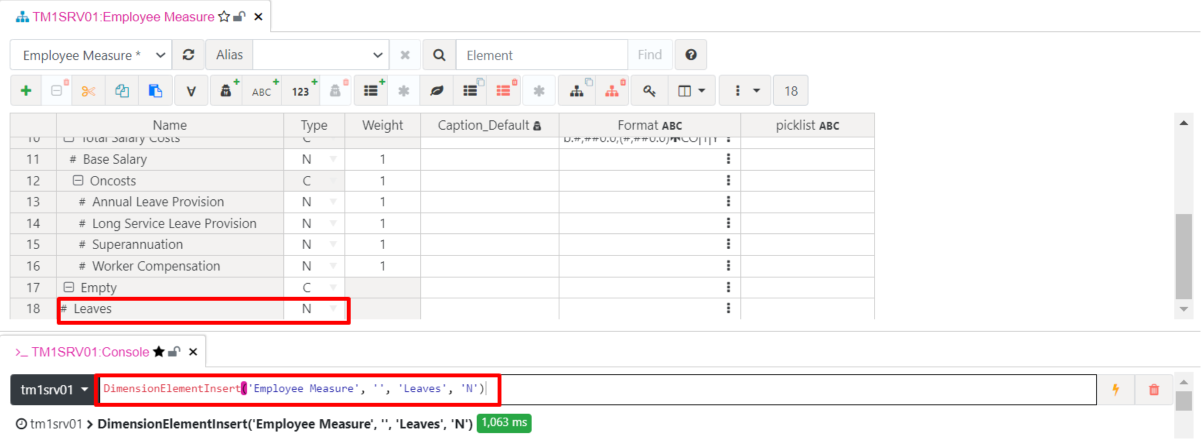
tm1srv01

```
dimensionele
DimensionElementComponentAdd
DimensionElementComponentAddDirect
DimensionElementComponentDelete
DimensionElementComponentDeleteDirect
DimensionElementDelete
DimensionElementDeleteDirect
DimensionElementInsert
DimensionElementInsertDirect
```

Fill the parameters as below:

DimensionElementInsert('Employee Measure', '', 'Leaves', 'N')

And then press **Enter** to execute the function, after clicking on the **refresh** button of the dimension editor, you should be able to see the new element **Leaves**, you can repeat this operation to add multiple elements without having to do a lot of clicks:



TM1SRV01:Employee Measure

Employee Measure * | Alias | Element | Find | 18

	Name	Type	Weight	Caption_Default	Format ABC	picklist ABC
10	⊟ Total Salary Costs	C			b:#,##0.0;(#,##0.0)CO|1|Y	
11	# Base Salary	N	1			
12	⊟ Oncosts	C	1			
13	# Annual Leave Provision	N	1			
14	# Long Service Leave Provision	N	1			
15	# Superannuation	N	1			
16	# Worker Compensation	N	1			
17	⊟ Empty	C				
18	# Leaves	N				

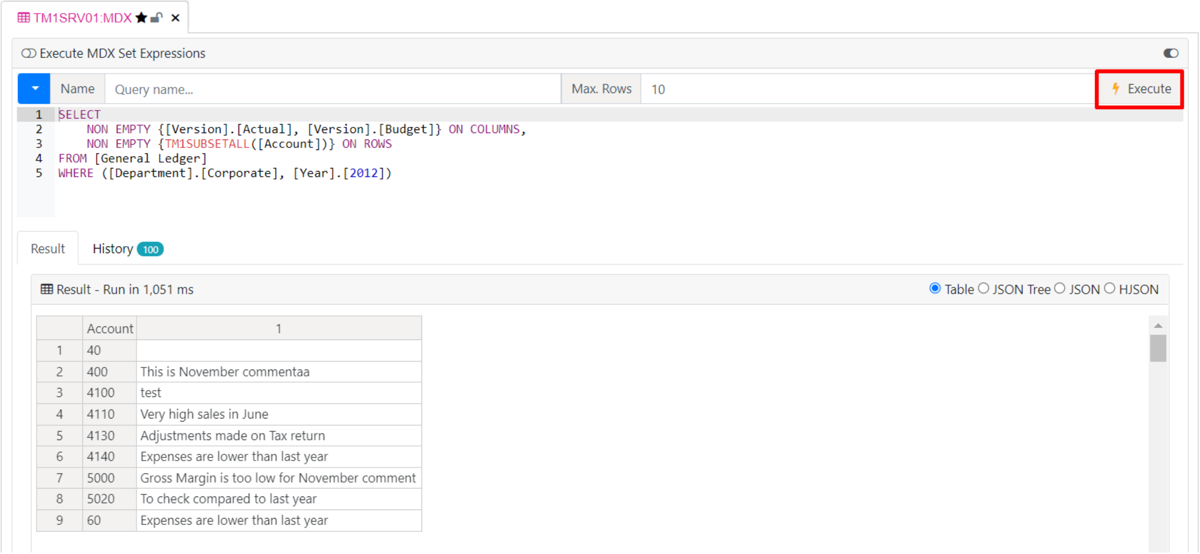
TM1SRV01:Console

tm1srv01

```
DimensionElementInsert('Employee Measure', '', 'Leaves', 'N')
tm1srv01 > DimensionElementInsert('Employee Measure', '', 'Leaves', 'N') 1,063 ms
```

MDX

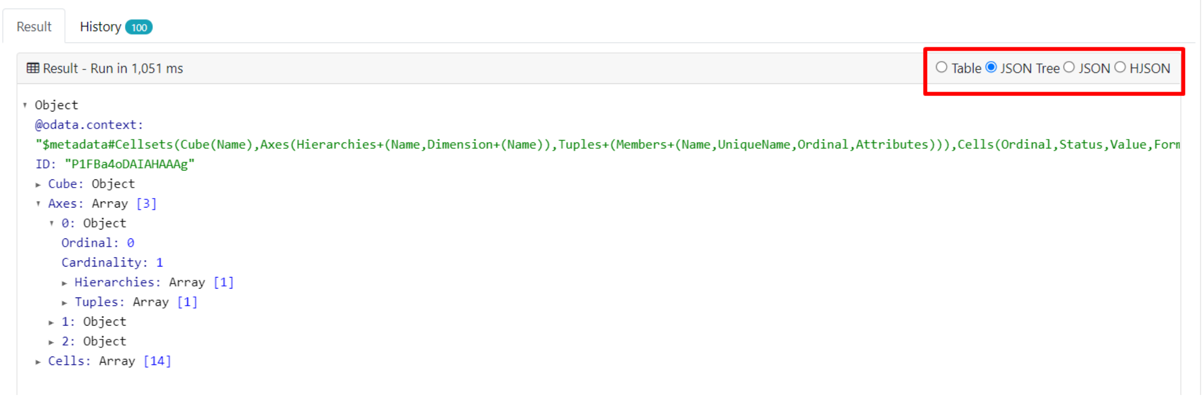
Arc comes with a very handy MDX tool to help you to write MDX queries, to open the tool, just click the **MDX** item from the menu. Once open, the tool comes with one sample, just click the **Execute** function to see the result:



```
SELECT
    NON EMPTY {[Version].[Actual], [Version].[Budget]} ON COLUMNS,
    NON EMPTY {TM1SUBSETALL([Account])} ON ROWS
FROM [General Ledger]
WHERE ([Department].[Corporate], [Year].[2012])
```

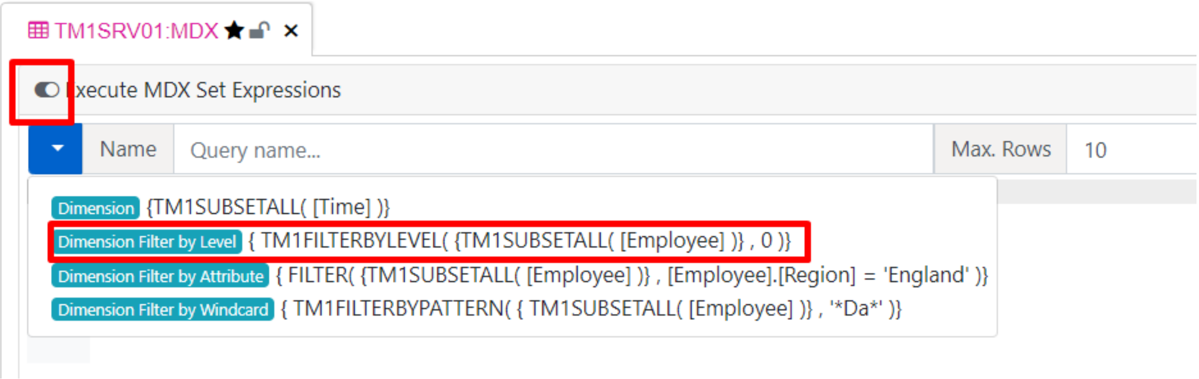
	Account	1
1	40	
2	400	This is November commentaa
3	4100	test
4	4110	Very high sales in June
5	4130	Adjustments made on Tax return
6	4140	Expenses are lower than last year
7	5000	Gross Margin is too low for November comment
8	5020	To check compared to last year
9	60	Expenses are lower than last year

If you are interested about how the data comes back from TM1, you can tick the **JSON** tooltip from the **Result** header and you will see the result as a JSON format:



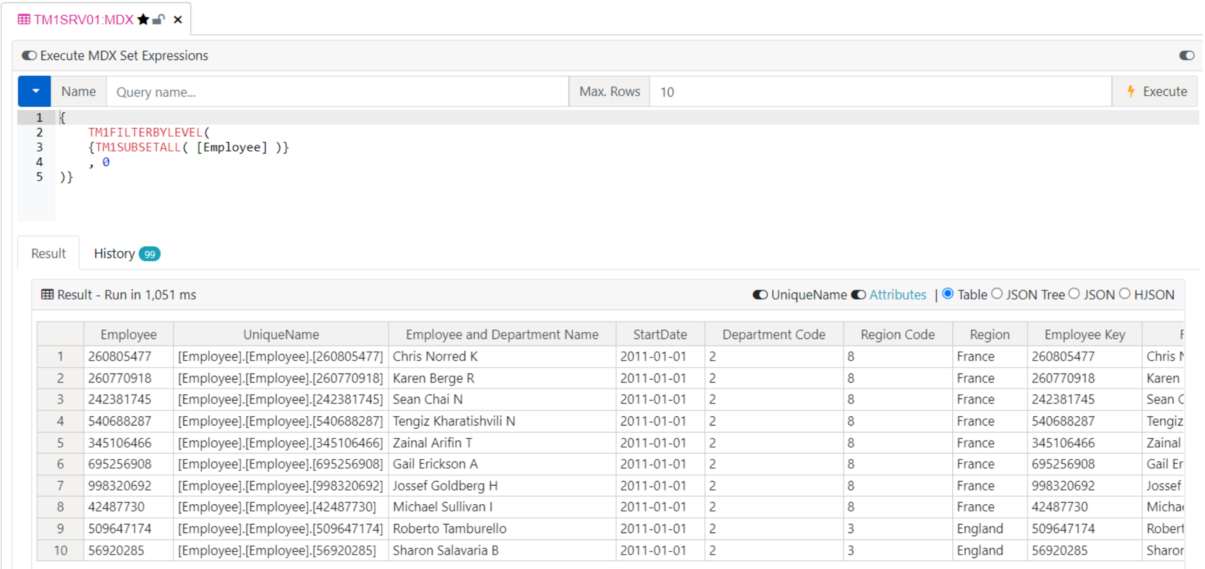
```
▾ Object
  @odata.context:
  "$metadata#Cellsets(Cube(Name),Axes(Hierarchies+(Name,Dimension+(Name)),Tuples+(Members+(Name,UniqueName,Ordinal,Attributes))),Cells(Ordinal,Status,Value,Form
  ID: "P1FBa4oDAIAHAAg"
  ▸ Cube: Object
  ▾ Axes: Array [3]
    ▾ 0: Object
      Ordinal: 0
      Cardinality: 1
      ▸ Hierarchies: Array [1]
      ▸ Tuples: Array [1]
    ▸ 1: Object
    ▸ 2: Object
  ▸ Cells: Array [14]
```

Now let's use the MDX Editor to query a dimension, to do that we need first to tick **Execute MDX Set Expressions** as below and you can select the second MDX example:



- Dimension {TM1SUBSETALL([Time])}
- Dimension Filter by Level { TM1FILTERBYLEVEL({TM1SUBSETALL([Employee])} , 0)}
- Dimension Filter by Attribute { FILTER({TM1SUBSETALL([Employee])} , [Employee].[Region] = 'England')}
- Dimension Filter by Windcard { TM1FILTERBYPATTERN({ TM1SUBSETALL([Employee])} , '*Da*')}

It is going to prepopulate the MDX and finally click the **Execute** button to see the result, you can then tick the **Attributes** if you want to see the attributes values:



Max. Rows 10

```
{
    TM1FILTERBYLEVEL(
    {TM1SUBSETALL( [Employee] )}
    , 0
)}
```

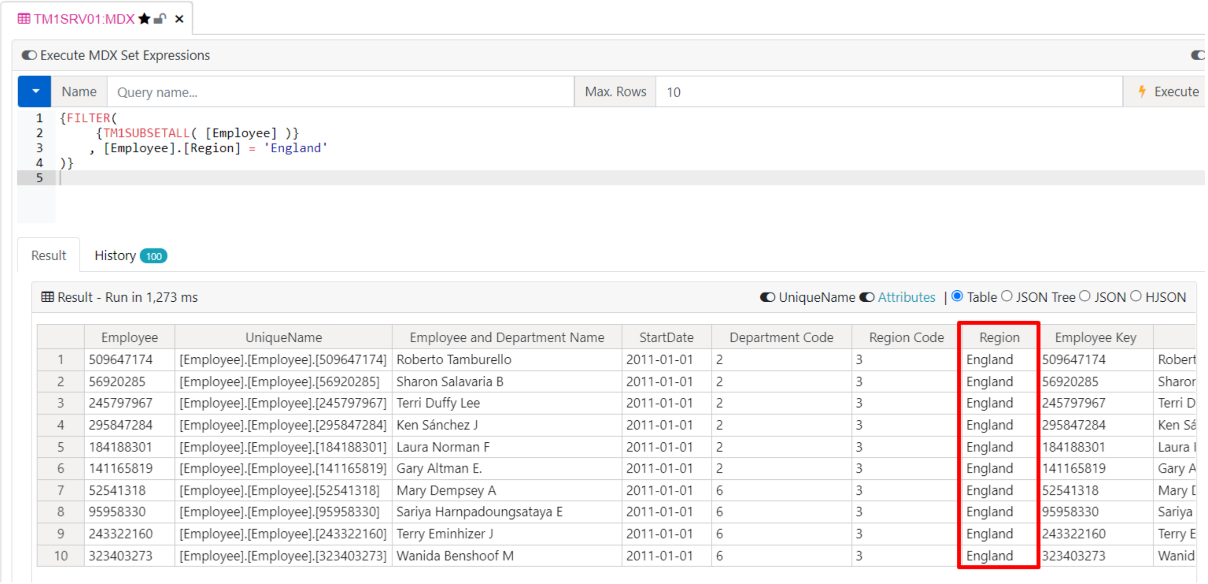
Result - Run in 1,051 ms

◐ UniqueName ◐ Attributes | ● Table ○ JSON Tree ○ JSON ○ HJSON

	Employee	UniqueName	Employee and Department Name	StartDate	Department Code	Region Code	Region	Employee Key	F
1	260805477	[Employee].[Employee].[260805477]	Chris Norred K	2011-01-01	2	8	France	260805477	Chris N
2	260770918	[Employee].[Employee].[260770918]	Karen Berge R	2011-01-01	2	8	France	260770918	Karen
3	242381745	[Employee].[Employee].[242381745]	Sean Chai N	2011-01-01	2	8	France	242381745	Sean C
4	540688287	[Employee].[Employee].[540688287]	Tengiz Kharatishvili N	2011-01-01	2	8	France	540688287	Tengiz
5	345106466	[Employee].[Employee].[345106466]	Zainal Arifin T	2011-01-01	2	8	France	345106466	Zainal
6	695256908	[Employee].[Employee].[695256908]	Gail Erickson A	2011-01-01	2	8	France	695256908	Gail Er
7	998320692	[Employee].[Employee].[998320692]	Jossef Goldberg H	2011-01-01	2	8	France	998320692	Jossef
8	42487730	[Employee].[Employee].[42487730]	Michael Sullivan I	2011-01-01	2	8	France	42487730	Micha
9	509647174	[Employee].[Employee].[509647174]	Roberto Tamburello	2011-01-01	2	3	England	509647174	Robert
10	56920285	[Employee].[Employee].[56920285]	Sharon Salavaria B	2011-01-01	2	3	England	56920285	Sharor

To filter the list of Employees based on the Region attribute value England we can use the following query:

```
{FILTER(
        {TM1SUBSETALL( [Employee] )}
        , [Employee].[Region] = 'England'
)}
```



Max. Rows 10

```
{FILTER(
     {TM1SUBSETALL( [Employee] )}
    , [Employee].[Region] = 'England'
)}

```

Result - Run in 1,273 ms

◐ UniqueName ◐ Attributes | ● Table ○ JSON Tree ○ JSON ○ HJSON

	Employee	UniqueName	Employee and Department Name	StartDate	Department Code	Region Code	Region	Employee Key	
1	509647174	[Employee].[Employee].[509647174]	Roberto Tamburello	2011-01-01	2	3	England	509647174	Robert
2	56920285	[Employee].[Employee].[56920285]	Sharon Salavaria B	2011-01-01	2	3	England	56920285	Sharor
3	245797967	[Employee].[Employee].[245797967]	Terri Duffy Lee	2011-01-01	2	3	England	245797967	Terri D
4	295847284	[Employee].[Employee].[295847284]	Ken Sánchez J	2011-01-01	2	3	England	295847284	Ken Sá
5	184188301	[Employee].[Employee].[184188301]	Laura Norman F	2011-01-01	2	3	England	184188301	Laura I
6	141165819	[Employee].[Employee].[141165819]	Gary Altman E.	2011-01-01	2	3	England	141165819	Gary A
7	52541318	[Employee].[Employee].[52541318]	Mary Dempsey A	2011-01-01	6	3	England	52541318	Mary D
8	95958330	[Employee].[Employee].[95958330]	Sariya Harnpadoungsataya E	2011-01-01	6	3	England	95958330	Sariya
9	243322160	[Employee].[Employee].[243322160]	Terry Eminhizer J	2011-01-01	6	3	England	243322160	Terry E
10	323403273	[Employee].[Employee].[323403273]	Wanida Benshoof M	2011-01-01	6	3	England	323403273	Wanid

TOPIC 7: EXPLORING THE REST API

Objectives

In this section, students will learn:

- Understanding what is the TM1 REST API
- Using the Arc REST API tool
- Writing your first REST API queries

What is the TM1 REST API

The TM1 REST API is a way of accessing data and everything else in TM1. Rather than being a proprietary API like old TM1 interfaces it is based on web standards making it accessible to a wide range of developers.

REST stands for **REpresentational State Transfer** and is an architectural style (not a standard) that has become the common way of designing API for applications on the internet. The TM1 REST API does however support the OData standard (v4) which provides a common way for accessing data through queries and also updating data. OData is used by multiple vendors and is supported by companies like Microsoft, SAP and of course IBM.

If you want to become a true TM1 REST API guru you should read up the documentation on the odata.org website. TM1 supports v4 of the standard, you may find examples out on the web that are an earlier version of the standard that won't work with TM1.

An important distinction to make about REST and OData is that neither of these are programming languages. The REST API is language agnostic, you can use the language of your choice: Java, JavaScript, Python, C#, Visual Basic, etc. All of these languages have HTTP libraries that will allow you to access the REST API.

What is great about the REST API?

It is fast! The REST API is built in to the TM1 server, there is no external web servers or parts you need to install. Because of this the REST API has direct access to all of that data stored in memory in your TM1 server.

It is fast over slow networks! The REST API is optimized for access over the internet, it takes advantage of all of the goodness that is built into the HTTP protocol such as compression, streaming, etc.

Did I mention it is fast? In our applications, we are noticing no discernible difference in speed when accessing applications from a local server to one that is hosted on the other side of the world.

You can use any programming language. Whatever your favourite programming language is the chances are you can use it with the REST API. As long as there is a HTTP library for your language, you can use it. Examples of these are:

- Java: Apache HTTP Client, okHttp, etc.
- C# and VB.NET: Built into the .NET Framework: System.Net.WebClient and System.Net.HttpWebRequest

- JavaScript: jquery, Angular $http.
- Python: httpie.
- Others: Search github.

It is very easy to get most information out of TM1, there is even a built-in document that describes the API called $metadata.

What ARE the hard bits ABOUT the REST API?

If we are going to make an objective assessment of the REST API we also need to understand that it isn't all smooth sailing with the REST API. In fact, because of the points below Cubewise decided to build a tool in Arc to help TM1 developers write REST API queries without all of the scary bits.

- It is low level. The REST API is built to access TM1 objects directly, this is easy for many queries but is more difficult for updating data, accessing application entries and a number of other things.
- Difficult for the traditional Accountant who recently became a TM1 Developer. As stated before, the REST API can be complicated for people without an IT background.
- Little cross-over from existing TM1 concepts and terminology. If you are a long time TM1 person you will be very familiar with functions such as DBRW and SUBNM as well as Active Forms and Slices. These concepts don't exist in the REST API but do in Canvas.

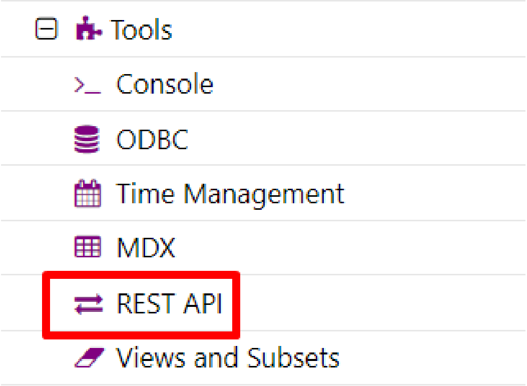
What can you use the REST API for?

Anything, the REST API is NOT a user interface however, it is a way to access objects/data in TM1. It could be used for pretty much anything:

- As an ETL tool to move data into and out of TM1.
- As a data source for a user interface.
- To sync data between TM1 servers.
- To backup data to another source or file.
- Create, update and delete TM1 objects.

How to access the Arc REST API tool

To open the **REST API** tool, you will find it under **Tools**, just click on **REST API** item to open it:

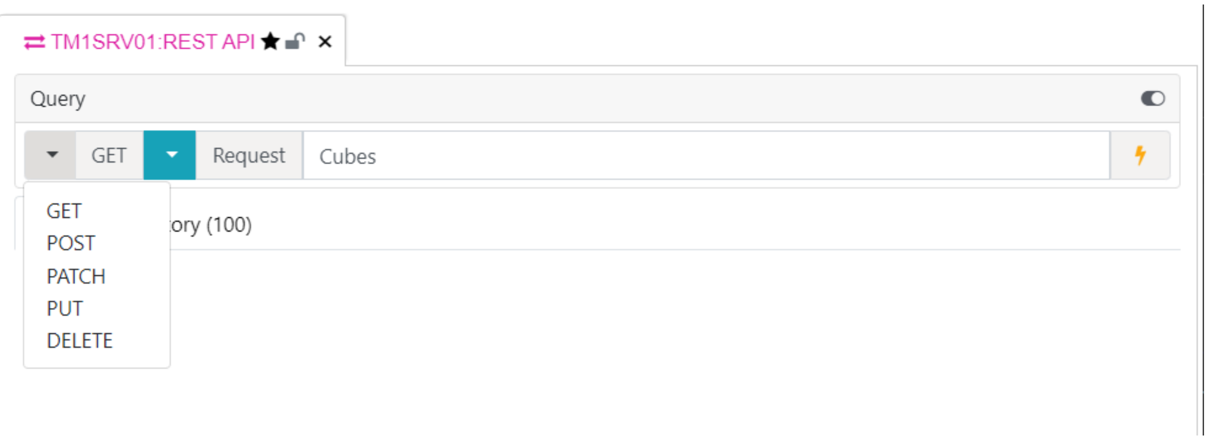


My first REST API queries

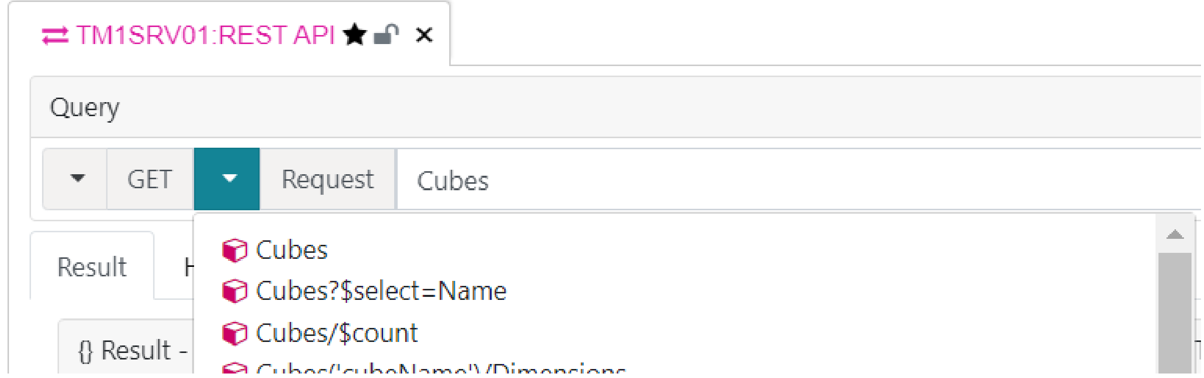
The REST API tools come with few samples, first you will need to choose the method, there are 4 methods:

- **GET**: To READ something
- **POST**: To CREATE something
- **PATCH**: To UPDATE something
- **DELETE**: To DELETE something

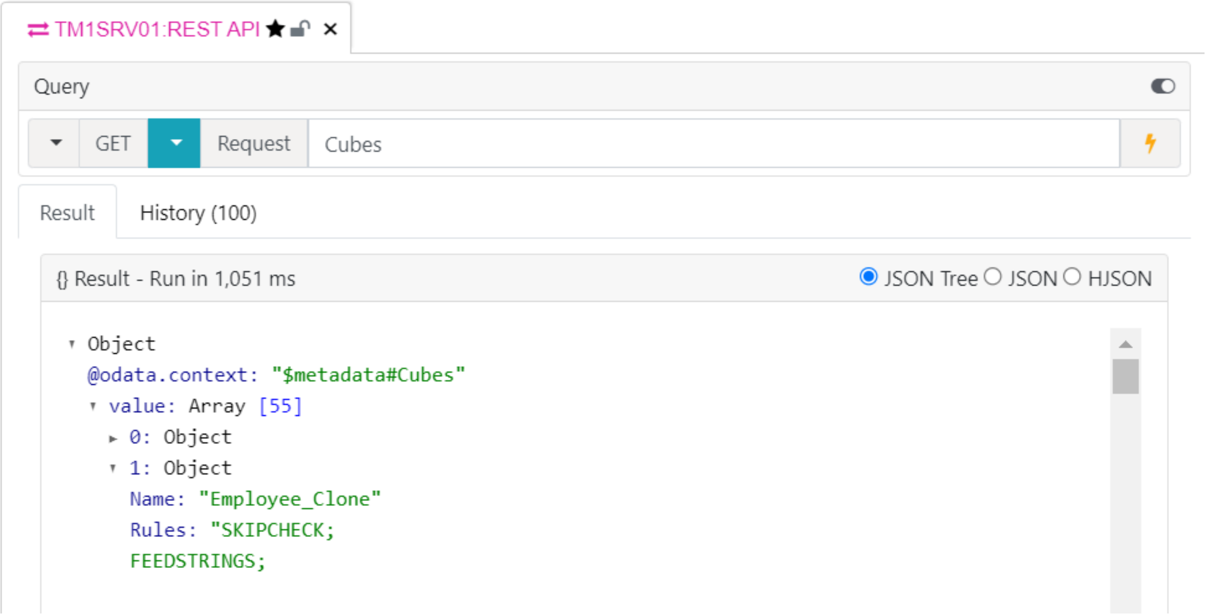
Let's start with the GET requests:



Then we can use the blue dropdown to select an example, let's start with **Cubes**:



Then click the **Execute** button to see the **Result**:



Cubes

{} Result - Run in 1,051 ms

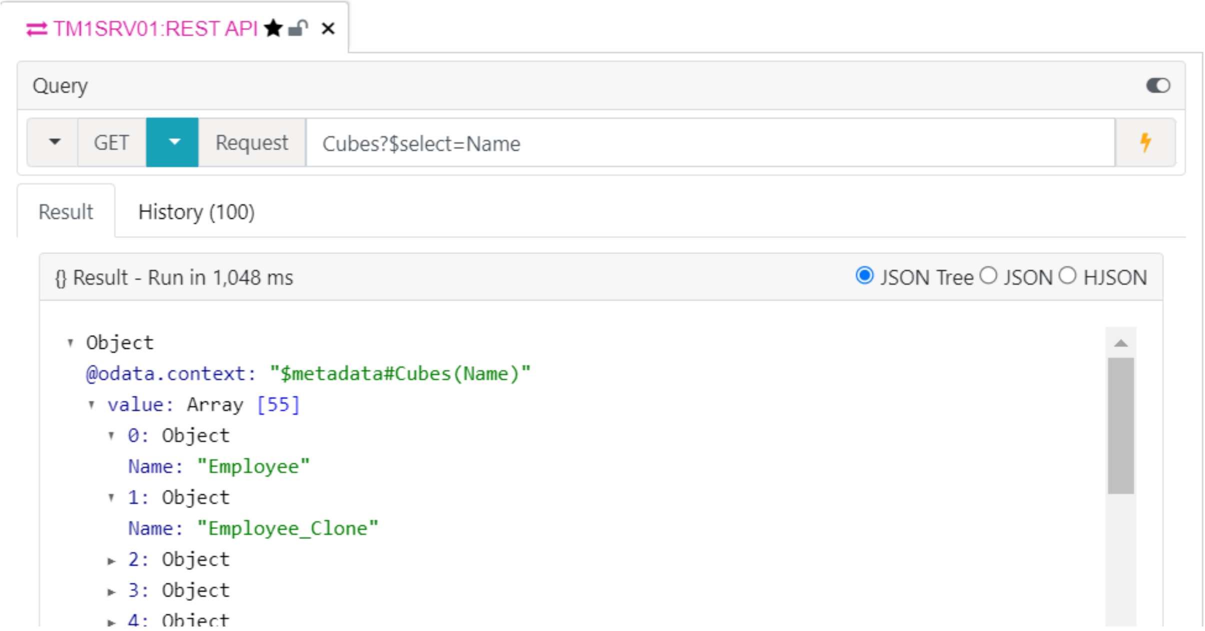
```
▾ Object
  @odata.context: "$metadata#Cubes"
  ▾ value: Array [55]
    ▸ 0: Object
    ▾ 1: Object
      Name: "Employee_Clone"
      Rules: "SKIPCHECK;
      FEEDSTRINGS;
```

The Result is the form of a JSON object, Arc uses a JSON viewer enabling you to expand/collapse all properties.

Cubes will return the list of cubes with their properties. As this list includes the content of the rule files, if your applications contain lots of cubes with large rule files, it might take some time to retrieve the list of cubes.

Getting all cubes name

A faster way to retrieve the list of cubes, is **Cubes?$select=Name**, this will retrieve the list of cubes with only the name as properties:



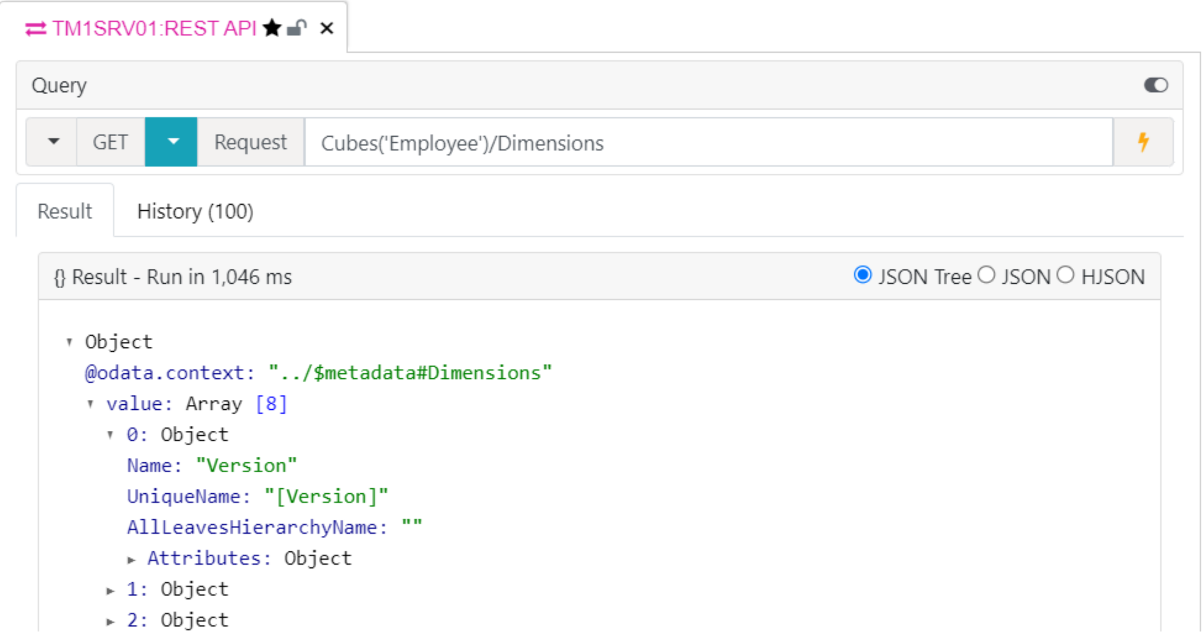
Cubes?$select=Name

{} Result - Run in 1,048 ms

```
▾ Object
  @odata.context: "$metadata#Cubes(Name)"
  ▾ value: Array [55]
    ▾ 0: Object
      Name: "Employee"
    ▾ 1: Object
      Name: "Employee_Clone"
    ▸ 2: Object
    ▸ 3: Object
    ▸ 4: Object
```

Getting all dimensions of a cube

To retrieve all dimensions of a cube, you will have to specify the cube name in the query such as **Cubes('Employee')/Dimensions** to retrieve the dimensions of the Employee cube:



Cubes('Employee')/Dimensions

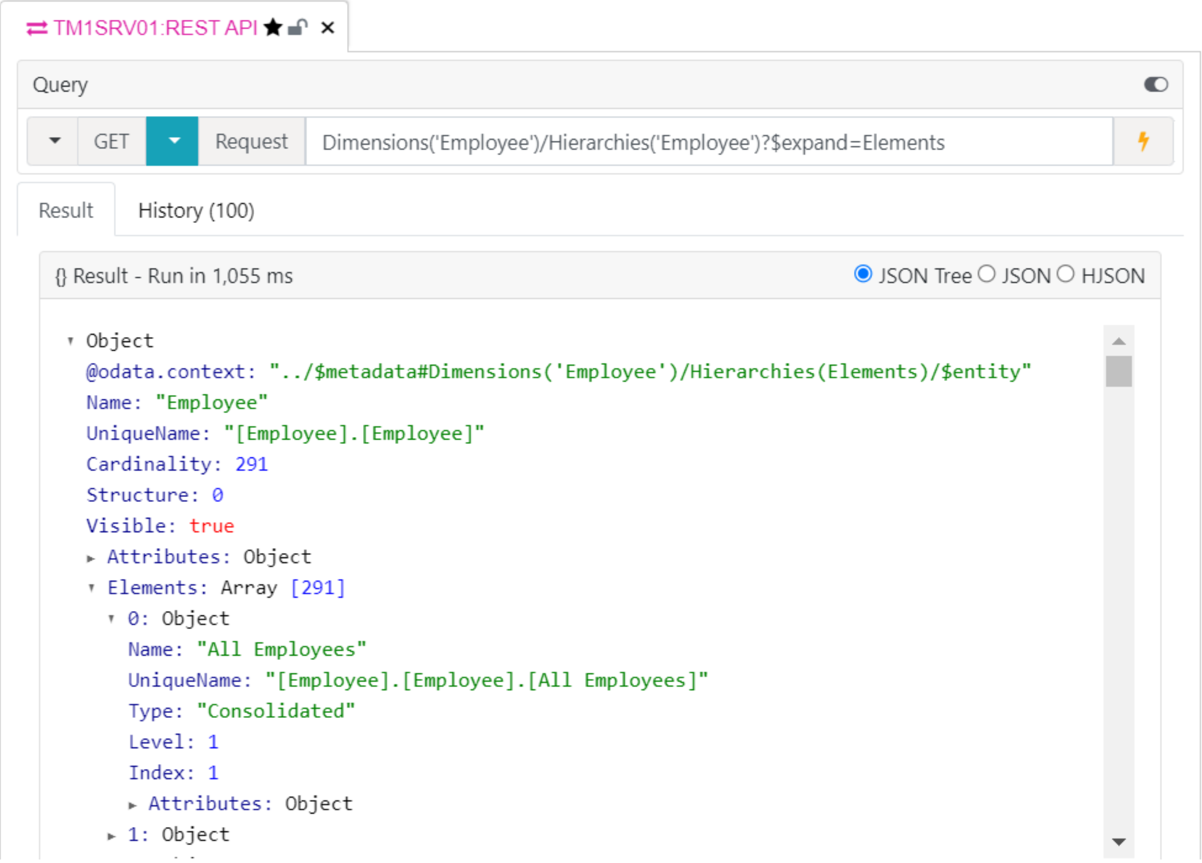
{} Result - Run in 1,046 ms

```
▾ Object
  @odata.context: "../$metadata#Dimensions"
  ▾ value: Array [8]
    ▾ 0: Object
      Name: "Version"
      UniqueName: "[Version]"
      AllLeavesHierarchyName: ""
      ▸ Attributes: Object
    ▸ 1: Object
    ▸ 2: Object
```

Getting the list of elements

To retrieve all elements of a specific dimension you will need to specify the Hierarchy as well, if this dimension does not have any hierarchies, the default Hierarchy name is the dimension name, for example to get the list of all elements of the Employee dimension, we will need to use

- **Dimensions('Employee')/Hierarchies('Employee')?$expand=Elements**



Dimensions('Employee')/Hierarchies('Employee')?$expand=Elements

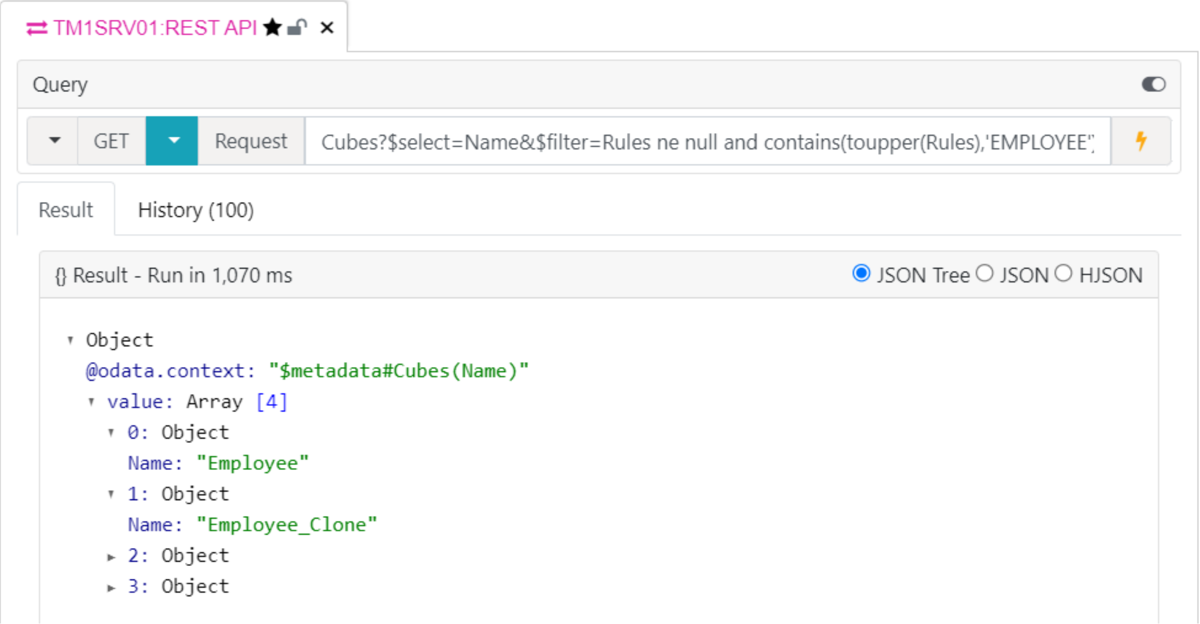
{} Result - Run in 1,055 ms

```
▾ Object
  @odata.context: "../$metadata#Dimensions('Employee')/Hierarchies(Elements)/$entity"
  Name: "Employee"
  UniqueName: "[Employee].[Employee]"
  Cardinality: 291
  Structure: 0
  Visible: true
  ▸ Attributes: Object
  ▾ Elements: Array [291]
    ▾ 0: Object
      Name: "All Employees"
      UniqueName: "[Employee].[Employee].[All Employees]"
      Type: "Consolidated"
      Level: 1
      Index: 1
      ▸ Attributes: Object
    ▸ 1: Object
```

Looking for a text in all rule files

You can use the REST API to do pretty much anything you want, for example you can write a query to look for a text inside all rule files. For example, the following query will search for the text **EMPLOYEE** in all rule files:

Cubes?$select=Name&$filter=Rules ne null and contains(toupper(Rules),'EMPLOYEE')



Cubes?$select=Name&$filter=Rules ne null and contains(toupper(Rules),'EMPLOYEE',

{} Result - Run in 1,070 ms

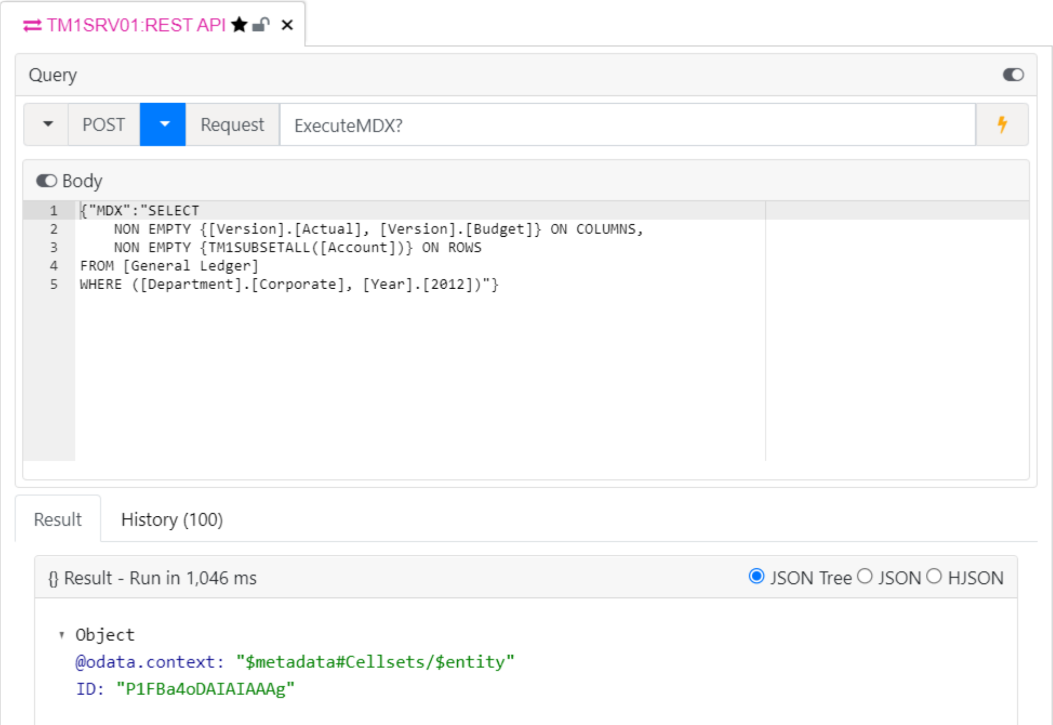
```
▾ Object
  @odata.context: "$metadata#Cubes(Name)"
  ▾ value: Array [4]
    ▾ 0: Object
      Name: "Employee"
    ▾ 1: Object
      Name: "Employee_Clone"
    ▸ 2: Object
    ▸ 3: Object
```

The text EMPLOYEE has been found in 3 rule files.

Getting cube dataa

To execute MDX query, you will need to change the method to a **POST.** Once you select the **POST** method, the Body area will appear. All **POST** queries require a Body. To query data from an MDX query, you will need to add the MDX query as part of the body.

The Request to execute MDX query is **ExecuteMDX?**



ExecuteMDX?

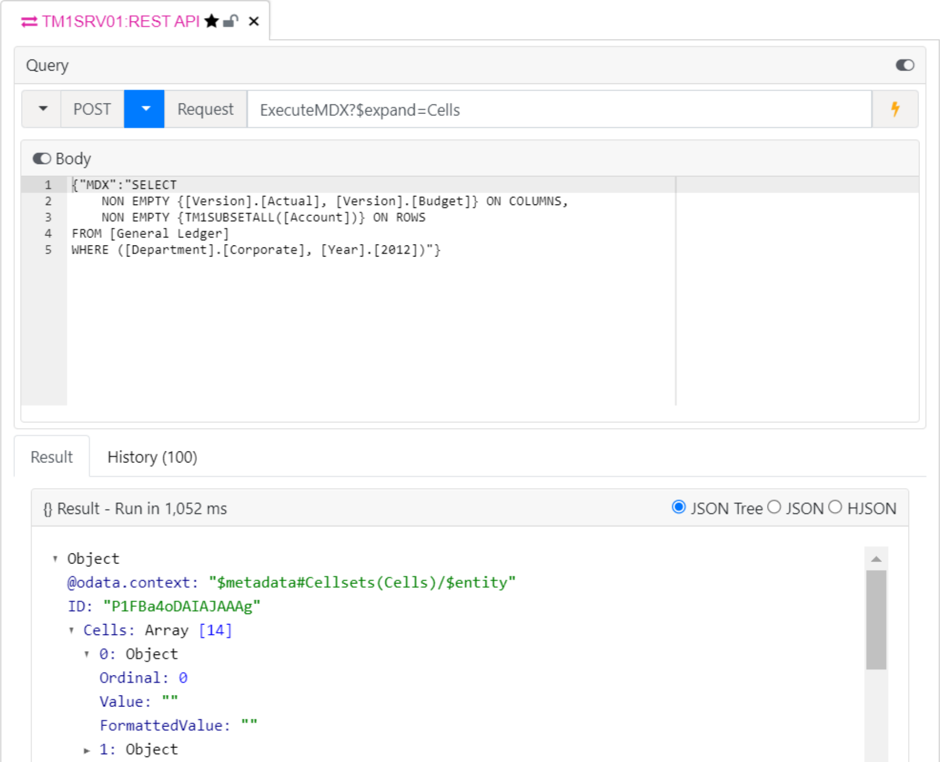
```
{"MDX":"SELECT
    NON EMPTY {[Version].[Actual], [Version].[Budget]} ON COLUMNS,
    NON EMPTY {TM1SUBSETALL([Account])} ON ROWS
FROM [General Ledger]
WHERE ([Department].[Corporate], [Year].[2012])"}
```

{} Result - Run in 1,046 ms

```
▾ Object
  @odata.context: "$metadata#Cellsets/$entity"
  ID: "P1FBa4oDAIAIAAAg"
```

ExecuteMDX? will check if the query is working but will not retrieve any values. To retrieve values, you will need to add **$expand=Cells** to the query as below:

ExecuteMDX?$expand=Cells



ExecuteMDX?$expand=Cells

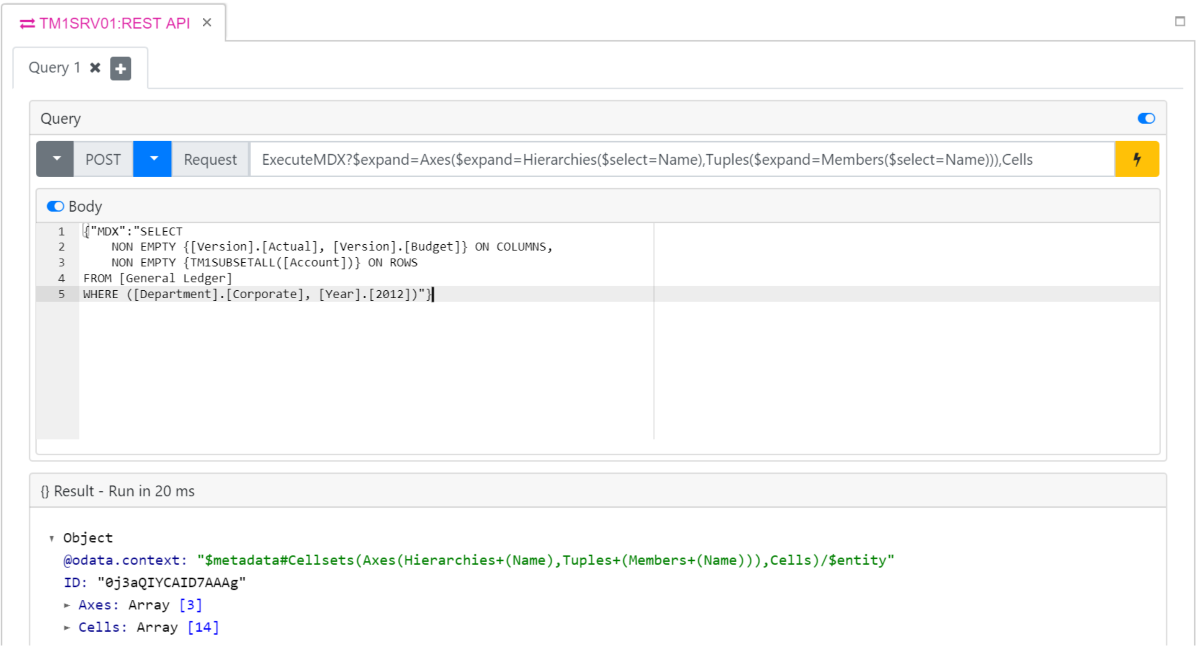
```
{"MDX":"SELECT
    NON EMPTY {[Version].[Actual], [Version].[Budget]} ON COLUMNS,
    NON EMPTY {TM1SUBSETALL([Account])} ON ROWS
FROM [General Ledger]
WHERE ([Department].[Corporate], [Year].[2012])"}
```

{} Result - Run in 1,052 ms

```
▾ Object
  @odata.context: "$metadata#Cellsets(Cells)/$entity"
  ID: "P1FBa4oDAIAJAAAg"
  ▾ Cells: Array [14]
    ▾ 0: Object
        Ordinal: 0
        Value: ""
        FormattedValue: ""
    ▸ 1: Object
```

This will retrieve all values from the MDX query but not the intersections. To retrieve the elements for each value you will need to add the **Axes** information from each Hierarchy as below:

ExecuteMDX?$expand=**Axes($expand=Hierarchies($select=Name),Tuples($expand=Members($select=Name))),**Cells



ExecuteMDX?$expand=Axes($expand=Hierarchies($select=Name),Tuples($expand=Members($select=Name))),Cells

```
{"MDX":"SELECT
    NON EMPTY {[Version].[Actual], [Version].[Budget]} ON COLUMNS,
    NON EMPTY {TM1SUBSETALL([Account])} ON ROWS
FROM [General Ledger]
WHERE ([Department].[Corporate], [Year].[2012])"}
```

{} Result - Run in 20 ms

```
▾ Object
  @odata.context: "$metadata#Cellsets(Axes(Hierarchies+(Name),Tuples+(Members+(Name))),Cells)/$entity"
  ID: "0j3aQIYCAID7AAAg"
  ▸ Axes: Array [3]
  ▸ Cells: Array [14]
```

TOPIC 8: MORE EXERCISES

Build Actual 2018 Region

a. Build a new hierarchy in the Employee dimension called "**Actual 2018 Region**" based on the **Region** attribute. Use this time the **ElementAttrInsert** function to create the alias.
b. Move employees using the Canvas page.
c. Build **Budget 2019 Region**.
d. Compare **Actual 2018 Region** with **Budget 2019 Region**

Debugging - Using Step in

To debug a process there are two other buttons that we did not cover in this hands-on, **Step-In** and **Step-out**. Debug the **Cub.GeneralLedger.Demo** process and use the **Step-In** button to debug the sub process **Cube.Wholesale.LoadFromFile**. Once inside the sub-process, you can then use the **Step-out** button to go back to the main process.

TOPIC 9: MORE TIPS

To learn more tips about Arc, you can go through the following articles:

- Deep dive into the Arc TM1 Security Editor
- Deep dive into the Arc TM1 Process Editor
- Deep dive into the Arc TM1 Rule Editor
- Using the Arc Cube Viewer
- Using the Arc Subset Editor

More articles:

- Arc blog: https://code.cubewise.com/arc-blog
- Arc help articles: https://code.cubewise.com/arc-help