Guidelines & Proven Practices for Implementing IBM Planning Analytics V2.0 Hierarchies

Stuart King
Offering Manager, IBM Planning Analytics
Email: stuart.king@ca.ibm.com

Andreas Kugelmeier
Executive Consultant, FOPM
Planning Analytics Architect
IBM Data and AI Expert Labs
Mobile Phone: +1-215-384-7302
Email: kugelmeier@us.ibm.com

Notices & Disclaimers

U.S. Government Users Restricted Rights - Use, duplication or disclosure restricted by GSA ADP Schedule Contract with IBM.

Information in these presentations and papers (including information relating to products that have not yet been announced by IBM) has been reviewed for accuracy as of the date of initial publication and could include unintentional technical or typographical errors. IBM shall have no responsibility to update this information. THIS document is distributed "AS IS" without any warranty, either express or implied. In no event shall IBM be liable for any damage arising from the use of this information, including but not limited to, loss of data, business interruption, loss of profit or loss of opportunity. IBM products and services are warranted according to the terms and conditions of the agreements under which they are provided.

Any statements regarding IBM's future direction, intent or product plans are subject to change or withdrawal without notice.

Performance data contained herein was generally obtained in a controlled, isolated environments. Customer examples are presented as illustrations of how those customers have used IBM products and the results they may have achieved. Actual performance, cost, savings or other results in other operating environments may vary.

References in this document to IBM products, programs, or services does not imply that IBM intends to make such products, programs or services available in all countries in which IBM operates or does business.

Workshops, sessions and associated materials may have been prepared by independent session speakers, and do not necessarily reflect the views of IBM. All materials and discussions are provided for informational purposes only, and are neither intended to, nor shall constitute legal or other guidance or advice to any individual participant or their specific situation.

It is the customer's responsibility to insure its own compliance with legal requirements and to obtain advice of competent legal counsel as to the identification and interpretation of any relevant laws and regulatory requirements that may affect the customer's business and any actions the customer may need to take to comply with such laws. IBM does not provide legal advice or represent or warrant that its services or products will ensure that the customer is in compliance with any law.

Information concerning non-IBM products was obtained from the suppliers of those products, their published announcements or other publicly available sources. IBM has not tested those products in connection with this publication and cannot confirm the accuracy of performance, compatibility or any other claims related to non-IBM products. Questions on the capabilities of non-IBM products should be addressed to the suppliers of those products. IBM does not warrant the quality of any third-party products, or the ability of any such third-party products to interoperate with IBM's products. IBM expressly disclaims all warranties, expressed or implied, including but not limited to, the implied warranties of merchantability and fitness for a particular purpose.

The provision of the information contained herein is not intended to, and does not, grant any right or license under any IBM patents, copyrights, trademarks or other intellectual property right.

IBM, the IBM logo, ibm.com, Aspera®, Bluemix, Blueworks Live, CICS, Clearcase, Cognos®, DOORS®, Emptoris®, Enterprise Document Management System™, FASP®, FileNet®, Global Business Services ®, Global Technology Services ®, IBM ExperienceOne™, IBM SmartCloud®, IBM Social Business®, Information on Demand, ILOG, Maximo®, MQIntegrator®, MQSeries®, Netcool®, OMEGAMON, OpenPower, PureAnalytics™, PureApplication®, pureCluster™, PureCoverage®, PureData®, PureExperience®, PureFlex®, pureQuery®, pureScale®, PureSystems®, QRadar®, Rational®, Rhapsody®, Smarter Commerce®, SoDA, SPSS, Sterling Commerce®, StoredIQ, Tealeaf®, Tivoli®, Trusteer®, Unica®, urban{code}®, Watson, WebSphere®, Worklight®, X-Force® and System z® Z/OS, are trademarks of International Business Machines Corporation, registered in many jurisdictions worldwide. Other product and service names might be trademarks of IBM or other companies. A current list of IBM trademarks is available on the Web at "Copyright and trademark information" at: www.ibm.com/legal/copytrade.shtml.

- IBM's statements regarding its plans, directions, and intent are subject to change or withdrawal without notice at IBM's sole discretion.
- Information regarding potential future products is intended to outline our general product direction and it should not be relied on in making a purchasing decision.
- The information mentioned regarding potential future products is not a commitment, promise, or legal obligation to deliver any material, code or functionality. Information about potential future products may not be incorporated into any contract.
- The development, release, and timing of any future features or functionality described for our products remains at our sole discretion.

Performance is based on measurements and projections using standard IBM benchmarks in a controlled environment. The actual throughput or performance that any user will experience will vary depending upon many factors, including considerations such as the amount of multiprogramming in the user's job stream, the I/O configuration, the storage configuration, and the workload processed. Therefore, no assurance can be given that an individual user will achieve results similar to those stated here.

Document Version History

Date	Version	Author	Description
01/08/2019	1.0	Stuart King	Version 1.0
09/05/2019	1.5	Andreas Kugelmeier	Version 1.5 DRAFT
04/09/2020	1.6	Andreas Kugelmeier	

Contents

1. An Introduction to Planning Analytics V2.0 Hierarchies

IBM Planning Analytics 2.0 introduced multiple hierarchy functionality that allows each dimension to contain multiple hierarchies. The dimension acts as the meta-data container object for the hierarchies, and each hierarchy may contain any number of elements and subsets. The parent-child relationships between the elements in a hierarchy are unique to each hierarchy in a dimension.

One could think of each hierarchy as a hierarchy container, and of the dimension as the object that holds all the containers: The default hierarchy (container) has the same name as the dimension. It is the same object as a dimension without any hierarchy containers. Other hierarchy containers may be named during creation.

An element does not need to exist in all hierarchies in a dimension.

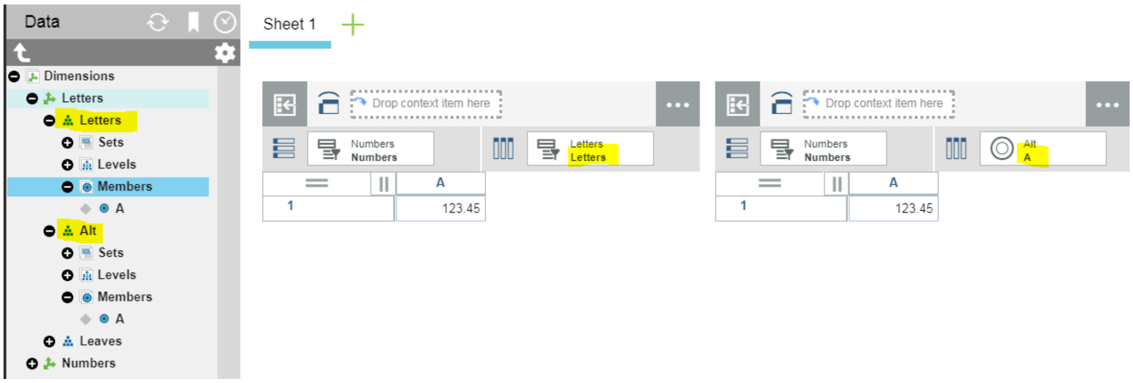
Advantages of alternate hierarchies:

a) Planning Analytics 2.0 Hierarchies function as 'virtual' dimensions: each hierarchy can be nested with other hierarchies from the same dimension and/or from other dimensions.
b) Support of true alternate hierarchies (a parent member can have different descendants in different hierarchies), allowing:
   - Business-unit or responsibility-specific rollups and management of hierarchies (example: a BU gets to maintain 'their' hierarchy themselves)
   - Easy what-if analysis of re-organizations & re-structuring
c) Create virtual attribute dimensions: attribute-based dimension hierarchies can be created by grouping elements based on patterns or common attributes.
d) Reduce the number of dimensions in cubes that contain dimensions based on properties or attributes of other dimensions.
e) Lower resource utilization (RAM) on cubes due to fewer dimensions

1.1 Cubes and Cells

Consider a cube with two dimensions named Letters and Number. The Letters dimension contains a single element 'A' and Numbers dimension contains a single element '1'. This cube will have a single cell that is an intersection between element 'A' and element '1'. If we create an alternate hierarchy for the Letters dimension, and assign element 'A' to that alternate hierarchy the cube still has only a single cell. The value of this cell is the same no matter which hierarchy in the Letters dimension is used to view the cell.
It follows that A leaf level element that exist in multiple hierarchies of a dimension does not create unique cells for each hierarchy.

In the following screenshot the dimension named Letters has a default hierarchy (named Letters) and an alternate hierarchy named Alt. Both hierarchies contain the same leaf level element 'A'. Two views of the same single cell cube are displayed. The difference between these views is the hierarchy in the Letters dimension.



The value of cell ['1','A'] is the same no matter which hierarchies containing element 'A' is used.

1.2 Cubeviews

Only MDX based views can access and process data outside of the default hierarchies, i.e. to process data based on master- and metadata from one or more PA V2.0 hierarchies, an MDX view needs to be created by assigning an MDX expression to the view via TI function ViewMDXSet or creating a view via ViewCreateByMDX.
'Legacy' (non-MDX) Views that are created and consumed in Architect, Perspectives, and Performance Modeler do not support alternate hierarchies.
Regardless of the source view, processing of data (reading or writing) can still be performed via CellGet*, CellIncrementN or CellPut* functions. If any of the targeted elements however is from a hierarchy, this hierarchy has to be specified as part of the CellGet*, CellPut*, or CellIncrement* statement, like:
CellPutN (123, <Cube>, <Element1>, <HierarchyA>:<Element2>, <Hierarchy>:<Element3>, …)
Or
CellPutN (123, 'Sales', 'Smartphones', 'Re-Org': 'Profit Center A', 'Region': 'Company B', '201908');

Just as with TI processes and rules functions, specifying the dimension name as the hierarchy name will process against elements in the default hierarchy (= the 'legacy' dimension container).

1.3 TI functions for element manipulation and processing

Any 'legacy' TM1 rule or TI element manipulation or processing function will only be able to process meta- and master-data of the 'default' hierarchy, i.e. for the regular dimension. To process as per meta- and master-data of a hierarchy, new functions have been made available that each contain the target hierarchy as a function parameter.
To determine the TM1 'level' of an element in a hierarchy for example, the function
ElementLevel (<Dimension>, <Hierarchy>, <Element>) needs to be used.
ELLEV (<Dimension>, <Element>) will only work against elements in the default hierarchy. Note: you can also use the new functions against the default hierarchy. The default hierarchy is the same as the dimension object, and its name is the same as the dimension name. It follows that in order to use ElementLevel (<Dimension>, <Hierarchy>, <Element>) against the default hierarchy, the function is called like this: ElementLevel (<Dimension>, <Dimension>, <Element>).

1.4 Securing Hierarchies

Hierarchies can be secured separately/in addition to 'legacy' dimension security.

The hierarchies of a dimension will show up in }Dimensions as <DimensionName>:<HierarchyName> and can be secured as such.

This feature allows specifying different access rights to different groups by hierarchy, also allowing the definition of group-based hierarchy owners. A group may for example be granted with ADMIN rights to a specific hierarchy, hence allowing the group to fully re-structure this hierarchy without needing to consider other hierarchies and other groups. Specific roles may have admin rights on a 'Re-Org' or 'What-If' hierarchy, allowing them to model/change the hierarchies as needed. Other groups may be blocked from seeing this hierarchy at all, or just be provided READ or WRITE access.

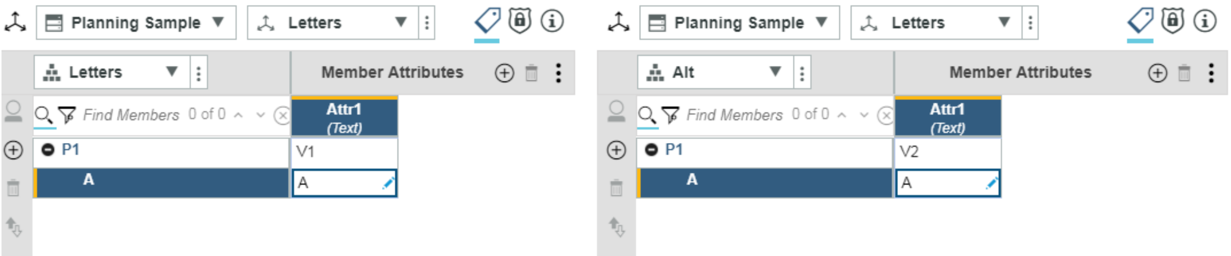
1.5 Element Security and Element Attributes

1.5.1 Leaf Elements

A leaf element with the same name in two or more hierarchies is the same element. These elements share the same attribute values and the same security. Any intersection in a cube that uses this element will have the same value regardless of the hierarchy being used. Not all leaf elements must exist in every hierarchy. Leaf-level ElementSecurity only needs to be processed for one leaf, but can be processed per Hierarchy (it is a good practice to process security fully for each hierarchy).

1.5.2 Consolidations

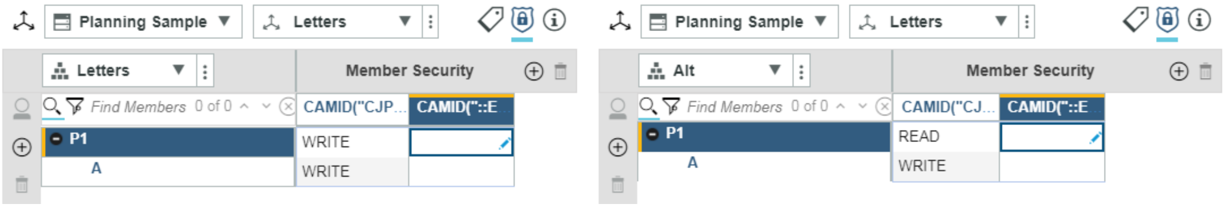
Consolidated elements with the same name in two or more hierarchies are not considered to be the same element. These consolidated elements will have unique attribute values and security for each hierarchy.
Examine the case where a parent element named P1 is added to both hierarchies of the letters dimension. The attribute value for the attribute named Attr1 is always the same for the leaf element 'A', but can differ for the consolidated element 'P1'.



Letters	Attr1 (Text)
P1	V1
A	A

Alt	Attr1 (Text)
P1	V2
A	A

Here we can see a difference in security for the consolidated element P1 in both hierarchies.



Letters	CAMID("CJP...	CAMID("::E
P1	WRITE	
A	WRITE	

Alt	CAMID("CJ...	CAMID("::E
P1	READ	
A	WRITE	

When element security is used, security must be assigned in the alternate hierarchy on consolidated elements in order for non-admin users to access those elements. When element security is used (an }ElementSecurity cube is created to the parent dimension) the default security for a non-admin using on a consolidated element in an alternate hierarchy is = 'NONE'. This is represented as a blank cell when examining the member security in the Workspace hierarchy editor.

1.5.3 Processing Security

'Legacy' TM1 Security processing functions such as ElementSecurityPut cannot be used for assigning security to elements in hierarchies. TM1 element (member) processing and manipulation functions that do not have a hierarchy as one of the parameters will only process to the default hierarchy (the 'regular' dimension). To process for a specific hierarchy, new functions are available such as HierarchyElementSecurityPut and HierarchyElementSecurityGet etc.

It should be noted that it is still a good practice to use CellPutS functions rather than HierarchyElementSecurityPut, because a CellPutS function that writes a security credential to a TM1 security cube is applied immediately.

Note that if one uses the dimension name as the hierarchy parameter in these new functions, they will process/calculate against the default hierarchy, i.e. function just as the legacy functions did. Please refer to the PA documentation and/or the TM1 Reference Guide for more information.

1.6 Subsets

Subsets are unique to the hierarchy the subset is created in. Creating a subset in the default hierarchy does not make the subset available to all hierarchies in the dimension. Architect, Perspectives, and Performance Modeler can only display subsets belonging to the default hierarchy of the dimension. However, these clients fully support creating Ti processes that can be used to create and manipulate subsets in any hierarchy of a dimension.
Using the SubsetCreate Ti function will create the subset in the default hierarchy of the dimension. In order to create a subset in an alternate hierarchy using a Ti script the HierarchySubsetCreate Ti function

must be used. Elements can be added or removed from a subset using the HierarchySubsetElementInsert and HierarchySubsetElementDelete Ti functions.

The list of Ti functions specific to subsets has been updated to accommodate subsets in alternate hierarchies.
https://www.ibm.com/support/knowledgecenter/SSD29G_2.0.0/com.ibm.swg.ba.cognos.tm1_ref.2.0.0.doc/c_subsetmanipulationturbointegratorfunctions_n72463.html

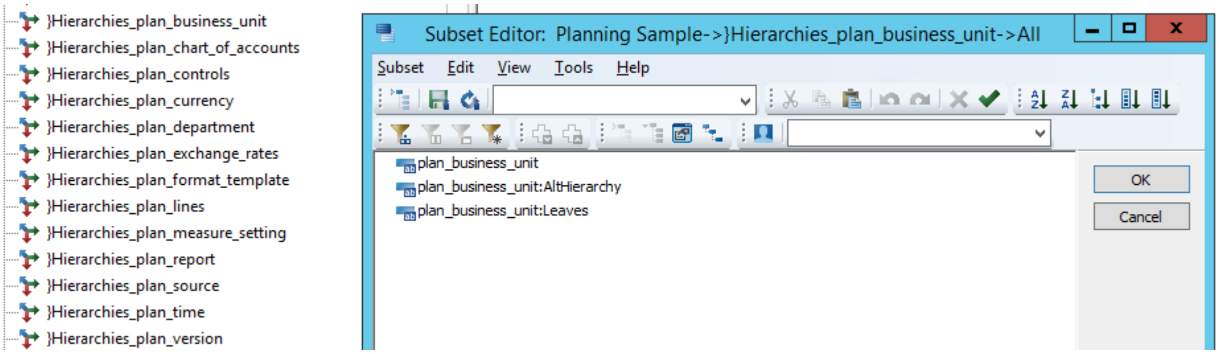
Additional Documentation:
https://www.ibm.com/support/knowledgecenter/en/SSD29G_2.0.0/com.ibm.swg.ba.cognos.tm1_dg_dvlpr.2.0.0.doc/c_multiplehierarchies.html

1.7 Enabling the use of PA V2.0 Hierarchies

By default, multiple hierarchies are not enabled. An administrator must configure the EnableNewHierarchyCreation tm1s.cfg setting (EnableNewHierarchyCreation=T) before you can work with multiple hierarchies and use related TurboIntegrator (TI) and Rules functions. This parameter is dynamic and does not require a restart of the Planning Analytics service after being added to the tm1s.cfg file.

1.8 Architect, Perspectives, and Performance Modeler

TM1 Architect, TM1 Perspectives, Performance Modeler were not updated to support modeling with the use of alternate hierarchies. It is however possible to create and run Ti processes in these clients that create and modify hierarchies. It is also possible to determine which dimension have alternate hierarchies. With control objects enabled examine the dimensions starting with }Hierarchies:


Subset Editor: Planning Sample->}Hierarchies_plan_business_unit->All
Subset
Edit
View
Tools
Help
OK
Cancel

}Hierarchies_plan_business_unit
}Hierarchies_plan_chart_of_accounts
}Hierarchies_plan_controls
}Hierarchies_plan_currency
}Hierarchies_plan_department
}Hierarchies_plan_exchange_rates
}Hierarchies_plan_format_template
}Hierarchies_plan_lines
}Hierarchies_plan_measure_setting
}Hierarchies_plan_report
}Hierarchies_plan_source
}Hierarchies_plan_time
}Hierarchies_plan_version

plan_business_unit
plan_business_unit:AltHierarchy
plan_business_unit:Leaves

Each of these }Hierarchies dimensions contains element representing the hierarchies in the dimension. In the above example the dimension contains elements representing the default hierarchy, and hierarchies name AltHierarchy, and Leaves. If an alternate hierarchy has not been created in a dimension the default hierarchy will still be represented in the corresponding }Hierarchies dimension. The }Hierarchies dimension can be used to configure language localization for hierarchy names in Planning Analytics Workspace and Planning Analytics for Excel.

The Leaves hierarchy seen in the above screenshot is created automatically when the first alternate hierarchy is created.

1.9 TI Processes

1.9.1 TI Functions

To create an alternate hierarchy through a Turbo Intergrator process the HierarchyCreate Ti function can be used.

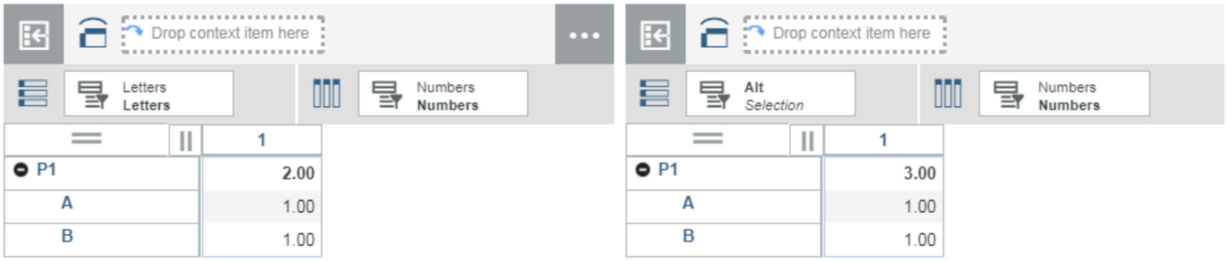
HierarchyCreate('Letters', 'Alt');
for example will create an alternate hierarchy named 'Alt' in the “Letters' dimension. The hierarchy created by this function does not include any elements. Elements can be added to the hierarchy using the HierarchyElementInsert Ti function the same way elements are added to a dimension (really the default hierarchy of a dimension) using the DimensionElementInsert function. The following Ti functions have the same result.
DimensionElementInsert('Letters', '', 'B', 'N');
HierarchyElementInsert('Letters'', 'Letters', '', 'B', 'N');

Note that using the DimensionElementInsert or DimensionElementInsertDirect Ti functions only adds the element to the default hierarchy in the dimension. If a Ti process must insert the element into any alternate hierarchy the HierarchyElementInsert or HierarchyElementInsertDirect Ti function must also be used.

In order to insert element 'B' to the hierarchy named 'Alt' we can use the following Ti function.
HierarchyElementInsert('Letters'', 'Alt', '', 'B', 'N');

If we need to add the element as a child of a parent we can use the following function.
HierarchyElementComponentAdd('Letters', 'Alt', 'P1', 'B', 2);

In the above example we used an element weight of 2 when assigning the element 'B' as a child of 'P1' in the Alt hierarchy. The weight of a child element can be different for the same element in multiple hierarchies. If we examine the same cubeview using both the default and alternate hierarchy we can see the difference in weights being used to determine the consolidated value for cell ['P1', '1'].



	1
P1	2.00
A	1.00
B	1.00

	1
P1	3.00
A	1.00
B	1.00

Another option for generating an alternate hierarchy is the CreateHierarchyByAttribute Ti function.
CreateHierarchyByAttribute('Letters', 'Attr1');

This Ti function uses values of an attribute named 'Attr1' in the Letters dimension to create and insert elements to a hierarchy named Attr1. The hierarchy that is created shares the same name as the attribute. The parent-child relationships of the elements in the created hierarchy are defined based on values of the attribute.

A full list of Hierarchy-specific Ti process functions that can be used to work with alternate hierarchies can be found here:
https://www.ibm.com/support/knowledgecenter/SSD29G_2.0.0/com.ibm.swg.ba.cognos.tm1_ref.2.0.0.doc/c_hierarchymanipulationturbointegratorfunctions.html

'Legacy' TM1 Dimension and Element Information TI Functions (such as DTYPE, DIMNM, ELISPAR etc.) will not apply to elements that are in hierarchies. **New dimension, hierarchy, and element information TI functions are available.** Note that if one uses the dimension name as the hierarchy parameter in these new functions, they will process/calculate against the default hierarchy, i.e. function just as the legacy functions did. Please refer to the PA documentation and/or the TM1 Reference Guide for more information.

1.9.2 TI-process utilities for Dimension and Hierarchy Management and Maintenance

For utilities (pre-built-TI processes) to create and manage hierarchies, please refer to TM1 Dimension Maintenance Tools and Utilities V2 (Documentation), and TM1 Dimension Maintenance Tools and Utilities V2 (TM1 Objects). Note that these tools/utilities are provided for free and 'as-is'.

1.10 Rules

1.10.1 Element Information Rules

Any 'legacy' TM1 element information rules will only be able to process meta- and master-data of the 'default' hierarchy, i.e. for the regular dimension. To process as per meta- and master-data of a hierarchy, new functions have been made available that each contain the target hierarchy as a function parameter. To determine the TM1 'level' of an element in a hierarchy for example, the function
ElementLevel (<Dimension>, <Hierarchy>, <Element>)
needs to be used (instead of ELLEV (<Dimension>, <Element>) which will only work against elements in the default hierarchy).

Note: you can also use the new functions against the default hierarchy. The default hierarchy is the same as the dimension object, and its name is the same as the dimension name. It follows that in order to use ElementLevel (<Dimension>, <Hierarchy>, <Element>) against the default hierarchy, the function is called like this: ElementLevel (<Dimension>, <Dimension>, <Element>).

In some cases, it can be useful to create different attributes per hierarchy. In the following example, a hierarchy-specific level attribute was created for a Company dimension, for hierarchies 'Current', 'Re-Org', '2018', '2017':

```
['TM1 Level'] = S: TRIM ( STR ( ELLEV ( 'Company', !Company ), 2, 0 ));
['TM1 Level - Current'] = S: TRIM ( STR ( ElementLevel ( 'Company', 'Current', !Company ), 2, 0 ));
['TM1 Level - Re-Org'] = S: TRIM ( STR ( ElementLevel ( 'Company', 'Re-Org', !Company ), 2, 0 ));
['TM1 Level - 2018'] = S: TRIM ( STR ( ElementLevel ( 'Company', '2018', !Company ), 2, 0 ));
['TM1 Level - 2017'] = S: TRIM ( STR ( ElementLevel ( 'Company', '2017', !Company ), 2, 0 ));
```

1.10.2 Qualifying Element Names in rules

When qualifying elements in rules (on the left and right hand side of a rules and feeder statement), applicable elements need to be qualified not only by dimension (if their name is ambiguous), but also by hierarchy (if a c-level element is used that may exist in more than one hierarchy). Remember that leaf level values are the same across hierarchies (yet a leaf does not have to exist in all hierarchies), but C-Level elements can have different values across different hierarchies.

Legacy qualification:
['<dimension>':{ '<Element1>', '<Element2>'},…]
Qualification including hierarchy:
['<dimension>': '<HierarchyA>':{ '<Element1>', '<Element2>'},…]

The same applies to DB statements, where the hierarchy is specified just like for CellGet* statements in TI, like:
DB ('Sales', !Version, !Time Period, …, 'Re-Org':!Cost Center, !Account, …);
(for cost centers from the 'Re-Org' hierarchy)

1.11 Attributes and Attribute Processing

'Legacy' Attr functions by default will process to the default hierarchy, but one can also use DimensionName:HierarchyName for the dimension name (instead of only dimension name) to process a hierarchy-specific attribute value.

New functions that allow specifying the hierarchy as an additional parameter were added, such as ElementAttrs (succeeding Attrs), ElementAttrPutS (succeeding AttrPutS), ElementAttrInsert (succeeding AttrInsert). Please refer to the PA documentation and/or the TM1 Reference Guide for more information.

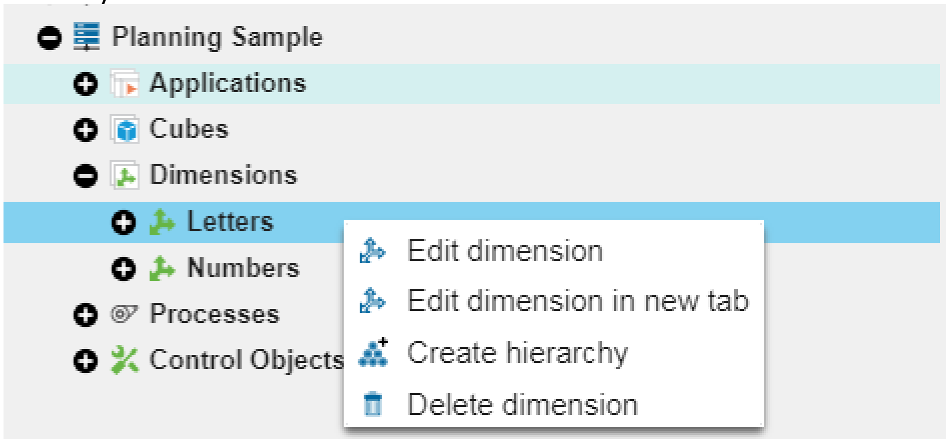
Note that C-level elements that are in multiple hierarchies are handled as different elements. This includes Alias Names. When processing Aliases for c-level elements, care should be taken to process the Alias names by hierarchy, using ElementAttrPutS.

1.12 Hierarchies in the TM1 Data Directory

Hierarchies' master- and meta-data (members and rollups) is not contained in the <dimension>.dim file. The <dimension>.dim file still contains only the members and rollups of the 'legacy'/default hierarchy. Hierarchy master- and meta-data is stored in a folder in the TM1 Data Directory that is named by the dimension name, followed by the suffix '}hiers'. In the <dimension>}hiers directory you then find *.dim files named after the hierarchies (<hierarchyname>.dim).
For example, Cost Center dimension hierarchies would be stored in a folder named 'Cost Center}hiers'. In 'Cost Center}Hiers', you can then find *.dim files named as per the individual hierarchy names. If the Cost Center dimension for example has three hierarchies named 'Re-Org', 'Profitability', and 'Region', the folder 'Cost Center}hiers' would contain three files named Re-Org.dim, Profitability.dim, and Region.dim.

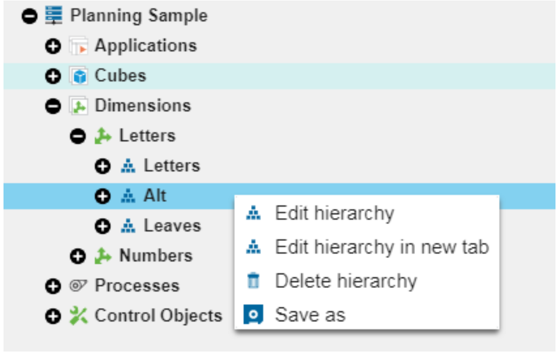
1.13 Planning Analytics Workspace

Planning Analytics Workspace provides a user interface to create and update hierarchies.
To create a new hierarchy a modeler or administrator can right click on a dimension and select Create hierarchy.



If the user chooses Edit dimension they see the same hierarchy editor interface for the default hierarchy. Once the first hierarchy is created the Leaves hierarchy will also be exposed. Unlike Architect, Perspectives, or Performance Modeler, any changes made to the structure of the hierarchy are performed on the Planning Analytics Server when the change is committed in Workspace. Workspace does not provide the ability to preview multiple changes to the hierarchy structure and then save changes to the server.

It is possible to duplicate a hierarchy by right clicking on the hierarchy in the explorer tree and selecting ***Save as***.



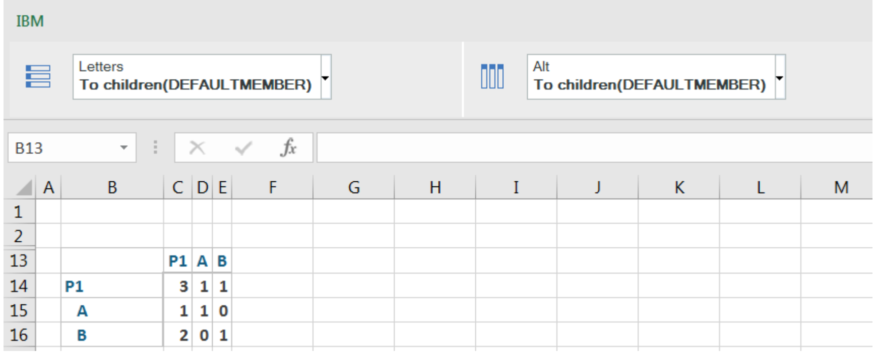
Please see <TI-process utilities for Dimension and Hierarchy Management and Maintenance> for utilities that allow automatic maintenance of hierarchies

1.14 Planning Analytics for Excel

Planning Analytics for Excel can examine and use alternate hierarchies but does not have functionality to create or update the default or alternate hierarchies. It is possible to update subsets in alternate hierarchies in Planning Analytics for Excel. Alternate hierarchies can be used in Explorations and Quick Reports. Dynamic and custom reports do not currently support alternate hierarchies.

It's worth noting that multiple Alternate Hierarchies from the same dimension can be used together in an exploration or quick report. The hierarchies must be used in context of a cube.

Examine the following exploration built with both the default hierarchy and alternate hierarchy from the Letters dimension. A single element from the numbers dimension is placed as a context filter.



Letters
To children(DEFAULTMEMBER)

Alt
To children(DEFAULTMEMBER)

	A	B	C	D	E
1					
2					
13			P1	A	B
14		P1	3	1	1
15		A	1	1	0
16		B	2	0	1

This is a join of the following views. To understand the results being returned from the Planning Analytics server we can examine each intersection of the view.

- The intersection between A and A is valid and shows the value that is common to element A.
- The intersection between B and B is valid and shows the value that is common to element B.
- The intersections between A and B is invalid as it cannot exist in the cube so a value of zero is displayed.
- The intersection between P1 (column) and B (row) shows the weighted (2) rollup value of element B in the Alt hierarchy
- The intersection between P1 (column) and A (row) shows the weighted rollup value of element A in the Alt hierarchy
- The intersection between P1 (column) and P1 (row) shows the weighted rollup value of element A plus element B (weight of 2) in the Alt hierarchy

2. REST API

The Planning Analytics Servers REST API provides capabilities to interact with hierarchies. The following HTTP calls can be used to read, create, and update hierarchies.

List the available hierarchies for a dimension.

GET https://<servername>:<port>/api/v1/Dimensions('Letters')/Hierarchies

Details of the default hierarchy in a dimension:

GET https://<servername>:<port>/api/v1/Dimensions('Letters')/Hierarchies('Letters')

Details of an alternate hierarchy in a dimension

GET https://<servername>:<port>/api/v1/Dimensions('Letters')/Hierarchies('Alt')

List the members of hierarchy. Members are required to read the weight of elements. Elements themselves do not store the weight.

GET https://<servername>:<port>/api/v1/Dimensions('Letters')/Hierarchies('Alt')/Members

List elements of a hierarchy

GET https://<servername>:<port>/api/v1/Dimensions('Letters')/Hierarchies('Alt')?$expand=Elements

List members of a hierarchy with elements

GET https:// ://<servername>:<port>/api/v1/Dimensions('Letters')/Hierarchies('Alt')/Members?$expand=Element

List edges in the hierarchy. Edges can be used to determine the parent-child relationships between elements specific to the hierarchy being used.

GET https://<servername>:<port>/api/v1/Dimensions('Letters')/Hierarchies('Alt')?$expand=Edges

List elements and edges of a hierarchy in a single query

GET https://<servername>:<port>/api/v1/Dimensions('Letters')/Hierarchies('Alt')?$expand=Elements,Edges

Create an alternate hierarchy with no elements

POST https://<servername>:<port>/api/v1/Dimensions('Letters)/Hierarchies

```
{ “Name”:”Alt” }
```

Create an alternate hierarchy with elements

POST https://<servername>:<port>/api/v1/Dimensions('Letters)/Hierarchies

```
{
 “Name”:”Alt”,
”Elements”:[{“Name”,”A”}, {“Name”,”B”}, {“Name”,”P1”}]
}
```

Create an alternate hierarchy with elements and edges (used to define the parent-child relationships of elements)

POST https://<servername>:<port>/api/v1/Dimensions('Letters)/Hierarchies

```
{
 “Name”:”Alt”,
”Elements”:[{“Name”,”A”}, {“Name”,”B”}, {“Name”,”P1”}],
”Edges”:[{“ParentName”:”P1”},{“ComponentName”:”A”}, {“ParentName”:”P1”},{“ComponentName”:”B”}]
}
```

3. Cognos Analytics

Click here for Introductory Guidelines and Proven practices for Planning Analytics with Cognos Analytics

3.1 Support of PA V2.0 Hierarchies

PA V2.0 Hierarchies are supported by Cognos Analytics when using the REST API as a connection method. Click here for information on how to connect to Planning Analytics via Cognos Analytics.

3.2 Recommendations

When implementing the Cognos Analytics 'filler-member' functionality on dimensions with ElementSecurity and that have multiple (alternate) rollups, it is highly recommended to implement the alternate rollups (the different hierarchies) as different PA V2.0 Hierarchies. This is because the filler members are not 'rollup-specific'. If there are two hierarchies in one dimension, and a user does not have access to the root element of either, there will be not two, but only one filler member. As a result, the hierarchies for a restricted user appear 'nested'. If however the different hierarchies are in separate hierarchy containers, the filler members are applied to the one hierarchy (per hierarchy container) only and will appear as expected.

4. Considerations

4.1 Hierarchy and Dimension Updates

Leaf elements are 'shared' between hierarchies of one and the same dimension in that a hierarchy may not have a consolidation element that already exists as a leaf in another hierarchy nor can it have a leaf element that already exists in another hierarchy as a consolidation. It follows that if one encounters errors during hierarchy dimension updates pertaining to elements already being in use, such as

> *Insertion of leaf "<elementname>" into hierarchy "<Dimension>:<Hierarchy>" failed because a consolidated element by that name already exists in the dimension*

one should check the 'other' hierarchies of the dimension for this element.

4.2 Time-Dependent Attributes

If attributes change over time and if the analysis needs to take this into considerations, we still have to 'dimensionalize' the attribute. A PA V2.0 hierarchy - if based on attributes - does only reflect the current state of the element to attribute relationship, whereas if we dimensionalize an attribute, the element to attribute to value assignment can occur in the context of time. Yes, in the context of traditional financial statement analysis, one typically only looks at the last organizational rollup/hierarchy, and thereby the last/current set of attributes of let's say cost centers etc. But if you look at Profitability Analysis for example, it is crucially important to maintain the relationship between elements, attributes, and time. A cost center may move and hence be classified differently which will have impacts on profitability. Also: if you re-state, you need to re-state based on historical structures.

4.3 PA V2.0 Attribute Hierarchies when using Element Security

Important security considerations for PA V2.0 hierarchies based on attributes if using ElementSecurity: Attribute-based hierarchies are rollups that leverage attributes for the consolidation nodes. It follows that when creating an attribute-based PA V2.0 hierarchy, the attributes are created as C-Level elements, with the leaves typically being the corresponding N-Level elements from the dimension. While the hierarchy inherits leaf-level security from the default hierarchy/dimension (i.e. leaf security for a PA V2.0 hierarchy is = security as per ElementSecurity), the access to the consolidation (which in this case here is the attribute value) needs to be granted to applicable users/groups. But: most users may not have access to all the leaves that make up an attribute-consolidation.

Let's take the simple example of a region attribute on Cost Centers. A user may have access to 400 or the 2000 cost centers in region North America, but if one creates an attribute-based hierarchy, and assigns the user access to attribute hierarchy element 'North America' (which is a consolidation of its cost center leaves), this user will be able to read the values for the 'North America' consolidation (ElementSecurity will still prevent the user from seeing the details of each Cost Center under North America – the user will still only be able to see details for the 400 of the 2000 cost centers, but the user will be able to see the total value for 'North America').

Let's use a customer profitability model, and let's say we have an attribute called 'income bracket'. The model is used by financial advisors who do not have access to all customers. But they would want to analyze by income bracket for 'their' customers. But if they look at the bracket '100,000-200,000' for example, they would read the data for ALL customers that belong to that bracket.

While we can run a TI process that will automatically assign READ access to an attribute consolidation if the user has access to all its leaf descendants, a such-secured attribute hierarchy is not very useful, because only folks with access to all leaves could really filter by attributes.

It follows that if there is ElementSecurity on a Dimension, using Attribute-based Hierarchies is typically not applicable.