Applies to:
SAP BI 7.0 Consultants. For more information, visit the EDW homepage.

Summary
This Article gives detail level of understanding on Hierarchy from modeling and Technical Aspects.

Author: Vipin A
Company: Cognizant Technology Pvt. Ltd
Created on: 29th August 2011

Author Bio
Vipin A is a Senior BI Consultant in Cognizant Technology Pvt. Ltd. He has 6 years of experience in BW and ABAP
Table of Contents

Hierarchy Definition ........................................................................................................................................... 3
Hierarchy Structures ........................................................................................................................................ 4
Hierarchy Maintenance .................................................................................................................................. 5
Hierarchy Activation ....................................................................................................................................... 6
Query Execution Details for Hierarchy ....................................................................................................... 6
Hierarchy Setting for Reporting Purpose ..................................................................................................... 8
NODE Attribute .............................................................................................................................................. 9
Hierarchy Version ........................................................................................................................................ 9
Intervals .......................................................................................................................................................... 9
Important Tables .......................................................................................................................................... 10
Important Temporary Hierarchy Tables for IDOC ..................................................................................... 10
Important Packages ..................................................................................................................................... 11
Related Content .......................................................................................................................................... 15
Disclaimer and Liability Notice .................................................................................................................. 16
**Hierarchy Definition**

A hierarchy is a method of displaying a characteristic structured and grouped according to individual evaluation criteria.

Example:

```
Sales Hierarchy

Region 1
| District 1
| Area 1  
| Area 2  
| Area 3  
| Area 3a 

Region 2
| District 2
| Area 4  
| Area 5  

Region 3
| District 3
| Area 6  
| Area 7  
| Area 8  
```


Hierarchy Structures

A) Components of Hierarchy:

<table>
<thead>
<tr>
<th>Root</th>
<th>• These are Top Of Hierarchy – They don’t have superior Nodes.</th>
</tr>
</thead>
<tbody>
<tr>
<td>Nodes</td>
<td>• The uppermost node is the root.</td>
</tr>
<tr>
<td></td>
<td>• Precisely one other node is directly superior to each node</td>
</tr>
<tr>
<td></td>
<td>(except for the uppermost node).</td>
</tr>
<tr>
<td></td>
<td>• Nodes that can be posted to refer to the hierarchy basic</td>
</tr>
<tr>
<td></td>
<td>characteristic.</td>
</tr>
<tr>
<td></td>
<td>• Nodes that cannot be posted either text nodes or external</td>
</tr>
<tr>
<td></td>
<td>characteristic nodes.</td>
</tr>
<tr>
<td>Leaves</td>
<td>• Consist of characteristic values for the basic characteristic -</td>
</tr>
<tr>
<td></td>
<td>have entries in the fact table.</td>
</tr>
<tr>
<td>Hierarchy Levels</td>
<td>• All nodes on the same level of the hierarchy (nodes that are</td>
</tr>
<tr>
<td></td>
<td>the same distance away from the root) form a hierarchy level.</td>
</tr>
<tr>
<td></td>
<td>• The root equals level 1.</td>
</tr>
<tr>
<td>Intervals</td>
<td>• Set of leaves that are described by its upper and lower</td>
</tr>
<tr>
<td></td>
<td>boundaries.</td>
</tr>
<tr>
<td></td>
<td>• Can be created if a node has more than one leaf.</td>
</tr>
</tbody>
</table>

B) Structure of Node:

<table>
<thead>
<tr>
<th>Nodes that can be Posted</th>
<th>Nodes that can not be Posted</th>
</tr>
</thead>
<tbody>
<tr>
<td>• All nodes that refer to the characteristic (hierarchy basic characteristic) for which the hierarchy has been created, are nodes that can be posted to.</td>
<td>• Nodes that cannot be posted do not refer to the basic characteristic. They are either text nodes that you can include in the hierarchy or external characteristic nodes.</td>
</tr>
<tr>
<td>• Transaction data can only be dragged from the data target for nodes that can be posted to.</td>
<td>• Text nodes function purely as headers, and they are there only to improve the structure of your hierarchy. They are special characteristic nodes for characteristic 0HIER_NODE.</td>
</tr>
<tr>
<td>• Leaves are nodes that can be posted and that do not have any lower-level nodes.</td>
<td>• In addition, you can attach external characteristic nodes to the hierarchy basic characteristic. You can identify any nodes, including a set of leaves with the characteristic value of any characteristic (InfoObject).</td>
</tr>
<tr>
<td>• The leaves of a hierarchy are characteristic values or characteristic value intervals (if intervals are allowed in the hierarchy for the characteristic).</td>
<td></td>
</tr>
</tbody>
</table>
Hierarchy Maintenance

RSD1 -> <Characteristics Name> -> Hierarchy

Hierarchy Details

Hierarchy Properties

- Hierarchies, version-dependent
- Hierarchy not time-dependent
- Entire hierarchy is time-dependent
- Time-Dependent Hierarchy Structure
- Use Temporal Hierarchy Join
- Intervals Permitted in Hierarchy
- Reverse +/- Sign for Nodes

Tables for Hierarchies

- Hierarchy table: /BIO/HPLANT
- Hierarchy SID tab: /BIO/KPLANT
- SID HierarchyStruct: /BIO/IPLANT
- HierInterval Table: 

Hierarchy Details

- Sales Hierarchy
  - Root Node
  - Nodes
  - Leaves

Technical Details

- Text Node
- Characteristic Node
- ‘Sales Representative’ Node
- Interval
Hierarchy Activation

- The hierarchy is either loaded into BW, i.e. from a R/3 or a flat file, or created in RSD1. The hierarchy has status inactive and is filed in table RSHIEDIR as M (modified) version (field OBJVERS).
- Activation creates in the corresponding hierarchy tables an A version from the M version.
- All nodes are checked for permitted characters.
- For those nodes that do not yet have an SID, an SID is created and stored in the corresponding SID table /BI*/S<IOBJNM>.
- Inclusion table /BI*/I<IOBJNM>, SID node table /BI*/K<IOBJNM>, and the RSRHIEDIR_O LAP table are updated, or new entries are created.
- The basis for creating the inclusion table is FM RSSH_HIERARCHY_READ, which delivers the hierarchy nodes for the current key date (the key date is only relevant for time-dependent hierarchies), where existing link nodes are expanded. This FM can be called up for each hierarchy in test mode.
- With hierarchies whose structures are time-dependent, the activation occurs for the SY-DATE key date. By calling with a key date in the query, an active version is created for this key date.

Situation during Activation:

All or some values of the Infoobject are displayed under the node “not assigned” in the query where the hierarchy is used

Solutions:
- There is no error – it could be correct. If the transaction data does not fit to the data in the hierarchy (for example, Plant 1000 is not included in the hierarchy), then this value is displayed under “not assigned”.
- The hierarchy data is not mapped to the correct Infoobject (for example, the 0Plant is loaded to ZPlant). The result is a valid hierarchy that doesn’t include any ZPlant nodes. Therefore, the ZPlant values are displayed under “not assigned”.
- Values in the hierarchy seem to adjust to the transaction data. The external representation adjusts to each other, but the internal representation differs. (For example, an ALPHA problem: external representation 1 but internal 1 and 0000001 are not equal). Check SID table and internal node name in the hierarchy maintenance.

Query Execution Details for Hierarchy
Structured field: `g_r_technfo->n_x_report`  
Initial Length (in Bytes): 696

<table>
<thead>
<tr>
<th>N.</th>
<th>Component name</th>
<th>T.</th>
<th>Ln.</th>
<th>Contents</th>
</tr>
</thead>
<tbody>
<tr>
<td>28</td>
<td>PARTPROV</td>
<td>h</td>
<td>8</td>
<td>Table[1x308]</td>
</tr>
<tr>
<td>29</td>
<td>MPH</td>
<td>h</td>
<td>8</td>
<td>Table[0x128]</td>
</tr>
<tr>
<td>30</td>
<td>RTIME</td>
<td>h</td>
<td>8</td>
<td>Table[0x36]</td>
</tr>
<tr>
<td>31</td>
<td>SEL</td>
<td>h</td>
<td>8</td>
<td>Table[6x376]</td>
</tr>
<tr>
<td>32</td>
<td>SFC</td>
<td>h</td>
<td>8</td>
<td>Table[10x244]</td>
</tr>
<tr>
<td>33</td>
<td>SFX</td>
<td>h</td>
<td>8</td>
<td>Table[3x212]</td>
</tr>
<tr>
<td>34</td>
<td>SH</td>
<td>h</td>
<td>8</td>
<td>Table[1x308]</td>
</tr>
<tr>
<td>35</td>
<td>SN</td>
<td>h</td>
<td>8</td>
<td>Table[1x76]</td>
</tr>
<tr>
<td>36</td>
<td>SUMM</td>
<td>h</td>
<td>8</td>
<td>Table[0x8]</td>
</tr>
<tr>
<td>37</td>
<td>VAR</td>
<td>h</td>
<td>8</td>
<td>Table[4x704]</td>
</tr>
</tbody>
</table>

ENDIF.

**Internal table**

| g_r_technfo->n_x_report-SH | | | | | |
|-----------------------------|---|---|---|---|
| SHID | CHARASID | CHANMID | CHANK |
| 1   | 1 | 287 | 287 | CMATERIAL  |

**Internal table**

| g_r_technfo->n_x_report-SN | | | | | |
|-----------------------------|---|---|---|---|
| SHID | SNID | SID | LEVEL | CHANMID | SIGN LISTING |
| 1   | 1 | 3 | 0 | 0 | 287 | I |
### Hierarchy Setting for Reporting Purpose

**Hierarchy Attributes:**

<p>| | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>A) Do not display leaves for inner nodes in the query</strong></td>
<td>There are different ways to display an inner node that can be posted in the query that is a node that can be posted and has lower-level children. Either a leaf is created for each inner node that is inserted directly below the inner node or these additional leaves are not displayed.</td>
</tr>
<tr>
<td><strong>B) Display behaviour for leaves of inner nodes can be changed</strong></td>
<td>You can set whether or not this display behaviour can be changed in the query at runtime. You have the following options: ' ' : Display behaviour can not be changed in the query 'X' : Display behaviour can be changed in the query</td>
</tr>
<tr>
<td><strong>C) Start Drilldown Level</strong></td>
<td>You can determine here to what level the hierarchy in the query should be drilled down, when it is first called up. This setting can be changed in the Query Builder.</td>
</tr>
<tr>
<td><strong>D) Suppressing the unassigned node</strong></td>
<td>You can make the setting here for suppressing the node, which contains all the values that can be posted to and do not appear in the hierarchy.</td>
</tr>
<tr>
<td><strong>E) Root/sum position</strong></td>
<td>You can determine here whether in the query the root, and thereby the totals item for the hierarchy, are displayed at the end of the query, and whether the leaves appear at the top. Whether the root, and thereby the totals item for the hierarchy, are displayed at the start of the query, and whether the leaves appear at the bottom.</td>
</tr>
</tbody>
</table>

**Hierarchy 'Sales Rep Hierarchy' Display: 'Active Version'**

![Hierarchy Attributes Settings](image)

**Display Parameters for Hierarchy Display in Query:**

- [ ] Do not Display Leaves for Inner-Nodes in the Query
- [ ] Display Behavior for Leaves of Internal Nodes Not Changeable
- [ ] Suppress 'Unassigned' Node

- [ ] Root/Totals Item Appears Above
- [ ] Root/Totals Item Appears Below

**Drilldown Start Lev.**
**NODE Attribute**

Time-Dependent Hierarchies:

**Use:**
In the Infoobject characteristic maintenance you determine whether or not a hierarchy may be time-dependent. You define either that the entire hierarchy should be time-dependent or that the hierarchy structure (each individual node relationship) should be time-dependent.

If you only need the hierarchy each month, you should create the entire hierarchy as time-dependent. However, if you only need the current view on the hierarchy, you should not create it as time-dependent. Of course you can activate it as often as you like.

**Entire Hierarchy Time-dependent:**
If an entire hierarchy is time-dependent, this means that there are versions of the hierarchy that are valid for one time interval each. The system automatically uses the current valid version in this case.

**Hierarchy Version**

**Requirements**
You have specified in the Info object maintenance, that the corresponding hierarchy characteristic is hierarchy version-dependent.

**Use**
Characteristic hierarchies can be used in different hierarchy versions

**Functions**
You can create different hierarchy versions under the same technical name. You then compare the different hierarchy versions in reporting.

**Intervals**
A node can be defined as an interval (INTERVFL in /BI*/H<IOBJNM>)

The interval value always refers to the hierarchy basic characteristic

The from-value is saved in the LEAFFROM field, and the to-value is saved in the LEAFTO field from the /BI*/J<IOBJNM> table

---

**Hierarchy ‘Lists for vendor dis’ Change: ‘Modified Version’**

<table>
<thead>
<tr>
<th>Lists for vendor distribution</th>
<th>InfoObject</th>
<th>Node Name</th>
<th>L</th>
<th>Valid for</th>
</tr>
</thead>
<tbody>
<tr>
<td>0HIER_NODE_ALV</td>
<td></td>
<td></td>
<td></td>
<td>03.06.2</td>
</tr>
<tr>
<td>ALV/CREDITOR_SUSMM</td>
<td>0CLASS_NUM</td>
<td>ALV/CREDITOR_SUSMM</td>
<td></td>
<td>03.06.2</td>
</tr>
<tr>
<td>ALV/Z_CRED_SUSMM_TRAIN</td>
<td>0CLASS_NUM</td>
<td>ALV/Z_CRED_SUSMM_TRAIN</td>
<td></td>
<td>03.06.2</td>
</tr>
</tbody>
</table>
Important Tables

1) RSHIEDIR
Table that contains the header for a hierarchy.

2) RSHIEDIRT
It’s just the text table for the table rshiedir with an index on HieID and Objvers to avoid deadlocks if the texts for more than one hierarchy are deleted parallel.

3) RSHIELVT
Level table to store the texts for the levels. This table is only filled if texts exist otherwise the default texts for the levels are used.

4) RSMHIERNODE
The textnodes are saved in this table.

5) RSTHIERNODE
This table contains the text of the text nodes of a hierarchy for each language. With the index on hieid and objvers the data for one hierarchy can be accessed fast.

6) H Table /BI*/H<OBJNM>
In this table the entire structure of the hierarchy is stored. The key for each node is the nodeID.

7) J Table /BI*/J<OBJNM>
In this table the intervals are stored for the node with the corresponding nodeid in the H-table.

Important Temporary Hierarchy Tables for IDOC

RSHIENODETMP
This is the temporary table where the structure of the hierarchy is stored (only IDOC – Upload).

RSHIEFOLTTMP
This is the temporary table where the texts for the texts nodes are stored (only IDOC Upload).

RSHIEINTVLTMP
This is the temporary table where the Intervals are stored (only IDOC Upload).

RSHIETXTTMP
Temporary table where the texts for the hierarchy header are stored (only IDOC Upload).
Important Packages

RSSH

Important Transactions

A) RSH1
Main transaction to start the maintenance for a special hierarchy or to get an overview over the existing hierarchies.

B) RHIERSIM
Start the simulation for IDOC hierarchies for an Oltp or BW-system that has an RFC connection to this BW system.

Classes

A) CL_RSSH_HIERARCHY_BASE

This is the most important and central class for the BW hierarchies!

CL_RSSH_HIERARCHY_BASE is the Base class that is used for the most hierarchy operations. An instance represents one hierarchy. On this hierarchy different operations can take place like checking, saving, subtree insert, create new nodes, delete nodes or subtrees etc. For every action that changes the hierarchy on the database this class is used.

Most Important Attributes:

P_SX_HIEPROPS
Protected Type RSSH_S_HIEPROPS
Contains the most important properties that are relevant for this hierarchy instance, like time dependence, hierarchy basis characteristic and it’s properties.

P_SX_HTABS
Protected Type RSSH_SX_HTABS
This complex structure contains all data structures and data tables that are relevant for this hierarchy. That are exactly the structures and tables that are stored in the database if the hierarchy is saved. Each structure and table has a fixed type that corresponds to the data type on the database. An exception is the H-table. This table is stored in a data reference that is created in the constructor of the class because the concret type of this table is only known during runtime and depends on the settings in the base characteristic.

P_S_STATUS
Protected Type RSSH_S_STATUSFLGS
Contains some status information. If an m-version exists on the database, if an A-version exists on the database. If the hierarchy is enqueued and if the change is allowed at this point of time. During runtime of the class before each changing of a hierarchy it is verified if the hierarchy is enqueued and if the change is allowed to avoid inconsistencies.

P_S_CHANGED
Protected Type RSSH_S_CHANGEDFLGS
In changed-structure is marked which structure or table was changed. Only the changed tables are updated on the database. An exception to this rule occurs if the field ALLTABS is set in the structure. That means the whole hierarchy has to be stored e.g. if a hierarchy is changed where only an active version is available a new M-version is created. That means the whole hierarchy (the new m-version) has to be stored.
B) CL_RSSH_HIERARCHY_FUNC

This class is a collection of static methods that allow external access to the hierarchies in the Warehouse Management.

Most Important Attributes:

P_R_HIER Type ref to CL_RSSH_HIERARCHY_BASE
Contains the hierarchy instance that is created to access a special hierarchy to read the hierarchy data.

Important Methods:

CHECK
This static method can be used to check a hierarchy. The whole hierarchy structure can be passed to the method or only the hieid is passed to the method, then the hierarchy is loaded from the database using the hieid. In any case an instance of the class cl_rssh_hierarchy_base is created. Then the method check of the instance is called. If an error log exists it is passed back to the caller or in dialog mode it is displayed in an amodal window.

CHECK_AND_ACTIVATE
Calls simply Function RSHIER_HIER_CHECK_AND_ACTIVATE

COPY
This method copies a hierarchy. The header data for the target hierarchy could be given by an import parameter or is inserted in a dialog that depends on the import parameter I_with_dialog. To copy the hierarchy two instances of the class cl_rssh_hierarchy_base are used. Enqueues and authority checks for both hierarchies source and target are done.

GET
External access to get all tables of a hierarchy or a subtree or only the hierarchy header that depends of the filling of the import parameters. The hierarchy can be selected by the hieID or the semantical key (unique index of rshiedir).

NDI_GET
This method is the basis for the RSNDI_SHIE-RFC-functions that provide hierarchy data. It calls the GET method and provides the node attributes of the hierarchy in a separate table so that the data is provided in a fixed table format that can be used for RFC.

NDI_UPDATE
Method that is executed by the RSNDI - RFC – Framework to update hierarchy data. It just prepares the parameter in that way that the real UPDATE method can be called.

UPDATE
This is the main Method that is called during the Upload of a hierarchy and uses an instance of class cl_rssh_hierarchy_base to save a hierarchy or subtree in the database.
C) CL_RSSH_HIERARCHY_SEL_ALV
Class that handles the main behavior of transaction RSH1 (Entry for hierarchy maintenance).

**Most Important Attributes**

<table>
<thead>
<tr>
<th>Attribute</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>P_R_ALV_GRID</td>
<td>Contains the instance of the ALV</td>
</tr>
<tr>
<td>P_R_CUSTOM_CONTAINER</td>
<td>Contains the instance of the custom container for the ALV</td>
</tr>
<tr>
<td>P_T_HIEDIRTXT</td>
<td>Contains the hierarchy headers with texts</td>
</tr>
<tr>
<td>P_IOBJNM</td>
<td>Contains the hierarchy basis characteristic that is used to filter the data</td>
</tr>
</tbody>
</table>

D) CL_RSSH_HIERDATA
This class is used for the data transfer of hierarchy data / subtrees for drag & drop.

**Most Important Attributes**

<table>
<thead>
<tr>
<th>Attribute</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>O_TX_HTABS</td>
<td>Contains the hierarchy data/ subtrees that should be transferred</td>
</tr>
<tr>
<td>O_SX_HIEPROPS</td>
<td>Contains the hierarchy properties</td>
</tr>
<tr>
<td>O_T_NODEID</td>
<td>Contains the root nodeIDs of the subtrees</td>
</tr>
</tbody>
</table>

E) CL_RSSH_HIERMAINTAIN
This is the Main class for the hierarchy maintenance that is managing the Main dynpro that means it manages and creates the control containers for the Hierarchy itself and for the main toolbar and it reacts on fcodes that are raised from the main toolbar or the main dynpro.

**Important Attributes**

<table>
<thead>
<tr>
<th>Attribute</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>P_R_HIERMTF</td>
<td>Type Ref To CL_RSSH_HIERMAINTAIN_FRONT</td>
</tr>
<tr>
<td>Instance of the special hierarchy instance inclusive representation in a tree control</td>
<td></td>
</tr>
<tr>
<td>P_R_MAINCONTAINER</td>
<td>Type Ref To CL_GUI_CUSTOM_CONTAINER</td>
</tr>
<tr>
<td>Main custom container of dynpro 100</td>
<td></td>
</tr>
<tr>
<td>P_R_HIERMT</td>
<td>Type Ref To CL_RSSH_HIERMAINTAIN</td>
</tr>
<tr>
<td>Instance of the hierarchy maintenance framework</td>
<td></td>
</tr>
<tr>
<td>P_R_TOOLBAR</td>
<td>Type Ref To CL_GUI_TOOLBAR</td>
</tr>
<tr>
<td>Instance of the main Toolbar</td>
<td></td>
</tr>
</tbody>
</table>
F) CL_RSSH_HIERMAINTAIN_BASE
This class is a Subclass of CL_RSSH_HIERARCHY_BASE.
This class extends the hierarchy basis class with methods that are needed if a hierarchy is changed manually. E.g. Insertion of single nodes, display a single node, move a subtree etc.

Important Attributes

P_CHANGE_MODE Type RS_BOOL
Flag to check if the hierarchy can be changed (change modus)
P_INSERTASNEXT Type RS_BOOL
Behavior if a node is inserted as the next or child node of the node where the insert action is performed on.
P_T_FIELDINDEXNODE Type RSSH_T_FIELDINDEX
Index table where relation between columns of the model tree and fields of the H-table is stored.

G) CL_RSSH_HIERMAINTAIN_FRONT
This is the Main class for the hierarchy maintenance that handles the representation and user interaction for a single hierarchy.

H) CL_RSSH_HIERMAINTAIN_LEVEL
This class displays the texts of the level of a hierarchy instance in a docking container for a hierarchy that is displayed in the hierarchy maintenance. The level texts can be maintained in up to the actual maximal level. When the hierarchy is stored the level texts are stored as well. To read the level texts the function RSSH_LEVEL_READ can be used that provides the texts for a hierarchy.

I) CL_RSSH_MESSAGE_HANDLER
This class can be used to be able to use error messages though you are using a control (e.g. ALV) that raises error messages while not in PAI.

J) CL_RSSH_SERVICE
This class contains hierarchy services that can be used from external programs. In the moment only the method FILL_CHILD_AND_NEXT is available. If you pass a hierarchy structure to the method the corresponding Child- and next-Ids are created and stored in the fields’ childID and nextID of that structure.

K) CL_RSSH_SHOW_LOG
With the statically method SHOW_MESSAGES_WRITE_LOG you can pass a message table to the class that is shown in an a modal window (cl_gui_dialogbox_container). You can close the a modal window using the method DELETE_WINDOW otherwise the window will pop up again if the same dynpro is called in the same internal modus.

L) CL_RSSH_TIMEHIERARCHY
This class is used from the time hierarchy framework (Transaction RSRHIERARCHYVIRT) to save the header data and level table of a time hierarchy in the header table rshiedir. This class uses the normal hierarchy framework (cl_rssh_hierarchy_base) to store the data.

Important Attributes

P_R_HIER Type Ref To CL_RSSH_HIERARCHY_BASE
The hierarchy basis instance that is used to access the hierarchy framework and to store the header data.
Related Content
https://www.sdn.sap.com/irj/sdn/index
For more information, visit the EDW homepage
Disclaimer and Liability Notice

This document may discuss sample coding or other information that does not include SAP official interfaces and therefore is not supported by SAP. Changes made based on this information are not supported and can be overwritten during an upgrade.

SAP will not be held liable for any damages caused by using or misusing the information, code or methods suggested in this document, and anyone using these methods does so at his/her own risk.

SAP offers no guarantees and assumes no responsibility or liability of any type with respect to the content of this technical article or code sample, including any liability resulting from incompatibility between the content within this document and the materials and services offered by SAP. You agree that you will not hold, or seek to hold, SAP responsible or liable with respect to the content of this document.