Creating Hierarchies & Groupings In Excel – One Click Solution

A lot of users like to see hierarchies in Excel and build groupings around these hierarchies so they can be collapsed and expanded easily. It is not a huge deal to do this for things that don't change a lot, like months rolling to a quarter, but it can be extremely cumbersome to maintain for organizational or account hierarchies that are large or change frequently.

By adding some VBA code (a macro) to your workbook, managing groupings can be completely automated. This can be customized for a plethora of different scenarios. Below are 2 examples that Hyperion users will encounter. One caveat to this is that Excel limits the number of grouping levels to 8. If the worksheet has more than 8 levels, the following logic would not provide the expected result.

Creating a Hierarchy Based On Excel Indents

If a spreadsheet exists where the hierarchy is created with the indent (not multiple columns) feature of Excel, select the range for the groupings to be applied. Execute the following script. Basically, this loops through the cells you have selected and will create the groupings based on the number of indents in the cell.

```
Sub CreateGroupingsOnIndents()

Dim cell As Range
For Each cell In Selection
    If cell.IndentLevel <> 0 Then
        cell.EntireRow.OutlineLevel = cell.IndentLevel
```

```
    Else
        cell.EntireRow.ClearOutline
    End If
Next

End Sub
```

Creating a Hierarchy Based On SmartView/Excel Add-In Indents

When retrieving from Essbase, cells are indented by adding 5 spaces to the member name. By getting the length of the cell, subtracting the number of spaces preceding the member name, and dividing the result by 5, the level of the indent is identified. Select the cells with the member names and execute the following.

```
Sub CreateGroupingsOnSpaces()

Dim cell As Range
Dim iLength1 As Integer
Dim iLength2 As Integer
Dim iIndent As Integer

For Each cell In Selection
    iLength1 = Len(cell.Value)
    iLength2 = Len(LTrim(cell.Value))
    iIndent = (iLength1 - iLength2) / 5
    If iIndent <> 0 Then
        cell.EntireRow.OutlineLevel = iIndent
    Else
        cell.EntireRow.ClearOutline
    End If
Next

End Sub
```

Setup a Module

If you are unfamiliar with adding custom code to an Excel

workbook, follow the steps below.

Excel 2000 and below

1. Select Tools/Macro/Visual Basic Editor
2. Right click on the workbook in the Project window, and select Insert/Module
3. Expand the module folder and open the new module (likely module1)
4. Paste the example above in this window to the right
5. Execute it by clicking F5 or the green play triangle in the toolbar

Excel 2003 and greater

1. Select the Navigation Wheel, and check the “Show Developer tab in the Ribbon” checkbox in the Popular tab
2. Select the Developer Ribbon and click Visual Basic
3. Right click on the workbook in the Project window, and select Insert/Module
4. Expand the module folder and open the new module (likely module1)
5. Paste the example above in this window to the right
6. Execute it by clicking F5 or the green play triangle in the toolbar

These can also be associated to a custom menu or toolbar if you choose to take the extra step!