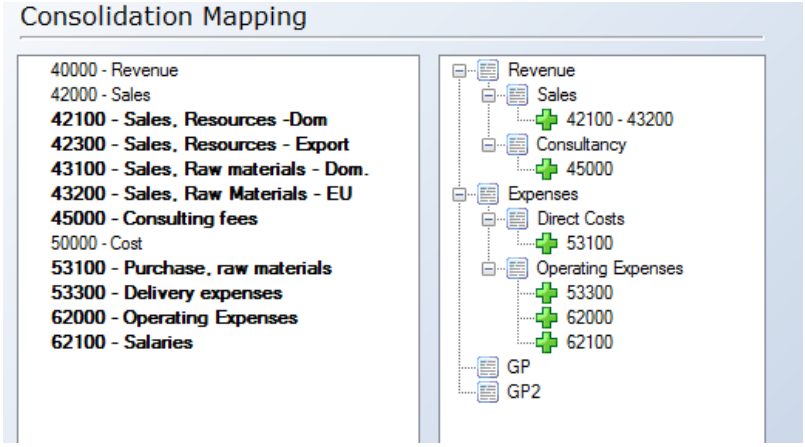
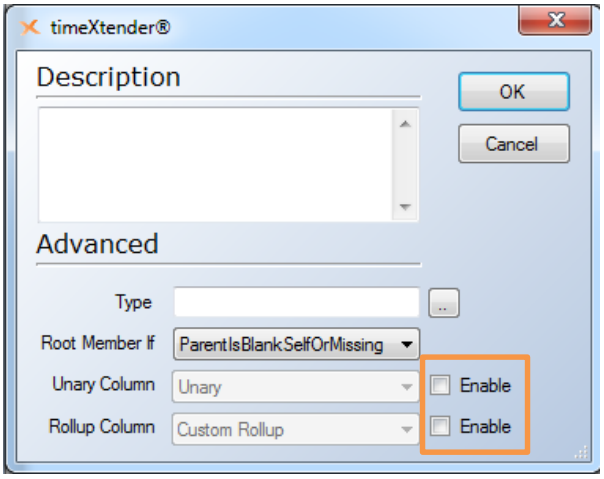
Multiple Parent-Child dimension rollup problems and Scopes

There is a problem with Analysis Services when it comes to Parent-Child dimensions, if you have multiple of these with custom rollups and unary operators, your cube will break. Not in the good way where it fails during processing or gives you an error in the reports, but in the sinister way where the results in your cubes will be wrong, with no warnings telling you that they are…

The following document will describe how you can use scopes to achieve the flexible results from custom rollups, without having to limit yourself to only one of these in a cube. This is not easy to maintain, since you must keep track of any changes in your parent child hierarchy structure, and keep the scope assignments up to date. But it will get the job done, and this will work for any parent-child or standard dimension.



I have created a very simple demo project to illustrate this issue; it contains a fact_finance table, a dim_accounts table and this consolidation table. The goal is to modify the table so the GP measures is Revenue – Direct Costs, GP2 is GP – Operating Expenses and the total (all member) is equal to GP2.



First of all I remove custom rollups and unary operators from the parent-child dimension to avoid the issues that this causes. This is done in the advanced section of the edit parent-child dimension dialog.

I will now create a *hidden* calculated measure, [AmountScoped], that we will use for the scope assignments; this is the MDX for it:

```
[Measures].[Amount], visible = 1
//Sets [Measures].[AmountScoped] measure to the standard measure amount, which we will
override later on in the scope.
//Also sets the measure to visible = 1, the one at the bottom (DummyTXScope) will be the
one getting hidden by the dimension setting.
;

SCOPE([Measures].[AmountScoped]);
//This is the measure we want to assign different values to.
            SCOPE([AccountsCons].[AccountsCons].[Level 02].[GP]);
            //For the root level member GP, give it the value from this calculation,
Revenue - Direct Costs.
                        THIS = ([AccountsCons].[AccountsCons].[Level
02].[Revenue],[Measures].[AmountScoped])-
                        ([AccountsCons].[AccountsCons].[Level 03].[Direct
Costs],[Measures].[AmountScoped]);
            END SCOPE;
            SCOPE([AccountsCons].[AccountsCons].[Level 02].[GP2]);
                        THIS = ([AccountsCons].[AccountsCons].[Level
02].[GP],[Measures].[AmountScoped])-
                        ([AccountsCons].[AccountsCons].[Level 03].[Operating
Expenses],[Measures].[AmountScoped]);
            END SCOPE;
            SCOPE([AccountsCons].[AccountsCons].[(All)]);
                        THIS = ([AccountsCons].[AccountsCons].[Level
02].[GP2],[Measures].[AmountScoped]);
            END SCOPE;
END SCOPE;

CREATE MEMBER CURRENTCUBE.[Measures].[DummyTXScope] AS Null
```

Please note that the order of the assignments is important, when assigning to GP2, I use GP, so this member must be calculated first and appear first in the script.

The scope must assign to all the members that needs different calculations. See the screenshot below for a comparison between the standard measure and the new scoped measure.

Row Labels	Amount	AmountScoped
Revenue	**52.000**	**52.000**
Sales	**42.000**	**42.000**
Sales, Resources -Dom	10.000	10.000
Sales, Resources - Export	12.000	12.000
Sales, Raw materials - Dom.	15.000	15.000
Sales, Raw Materials - EU	5.000	5.000
Consultancy	**10.000**	**10.000**
Consulting fees	10.000	10.000
Expenses	**43.300**	**43.300**
Direct Costs	**14.000**	**14.000**
Purchase, raw materials	14.000	14.000
Operating Expenses	**29.300**	**29.300**
Delivery expenses	8.000	8.000
Operating Expenses	1.300	1.300
Salaries	20.000	20.000
GP		**38.000**
GP2		**8.700**
Grand Total	**95.300**	**95.300**

In this example I only assign values to members that do not already contain data, the exact same approach can be used to override the value of any member that already has a value.