SAS® Financial Management 5.6

Formula Guide

SAS® Documentation

The correct bibliographic citation for this manual is as follows: SAS Institute Inc. 2019. *SAS® Financial Management 5.6: Formula Guide*. Cary, NC: SAS Institute Inc.

SAS® Financial Management 5.6: Formula Guide

Copyright © 2019, SAS Institute Inc., Cary, NC, USA

All rights reserved. Produced in the United States of America.

For a hard-copy book: No part of this publication may be reproduced, stored in a retrieval system, or transmitted, in any form or by any means, electronic, mechanical, photocopying, or otherwise, without the prior written permission of the publisher, SAS Institute Inc.

For a web download or e-book: Your use of this publication shall be governed by the terms established by the vendor at the time you acquire this publication.

The scanning, uploading, and distribution of this book via the Internet or any other means without the permission of the publisher is illegal and punishable by law. Please purchase only authorized electronic editions and do not participate in or encourage electronic piracy of copyrighted materials. Your support of others' rights is appreciated.

U.S. Government License Rights; Restricted Rights: The Software and its documentation is commercial computer software developed at private expense and is provided with RESTRICTED RIGHTS to the United States Government. Use, duplication or disclosure of the Software by the United States Government is subject to the license terms of this Agreement pursuant to, as applicable, FAR 12.212, DFAR 227.7202-1(a), DFAR 227.7202-3(a) and DFAR 227.7202-4 and, to the extent required under U.S. federal law, the minimum restricted rights as set out in FAR 52.227-19 (DEC 2007). If FAR 52.227-19 is applicable, this provision serves as notice under clause (c) thereof and no other notice is required to be affixed to the Software or documentation. The Government's rights in Software and documentation shall be only those set forth in this Agreement.

SAS Institute Inc., SAS Campus Drive, Cary, North Carolina 27513-2414.

June 2019

SAS provides a complete selection of books and electronic products to help customers use SAS® software to its fullest potential. For more information about our offerings, visit **sas.com/store/books** or call 1-800-727-0025.

SAS® and all other SAS Institute Inc. product or service names are registered trademarks or trademarks of SAS Institute Inc. in the USA and other countries. ® indicates USA registration.

Other brand and product names are trademarks of their respective companies.

Contents

Chapter 1 Introduction 1
Included in This Document 1
Additional Documentation 1
Example Data 1
Chapter 2 Formula Types 7
Overview of Formula Types and Formulas 7
Reporting Formulas and Excel-Based Calculated Members 8
Driver and Modeling Formulas 10
Chapter 3 Formula Computation 13
Order of Execution 13
Functional Currency Approach 17
Reporting Currency Approach 20
Chapter 4 Formula Basics for Server-Side Calculated Members 23
Introduction 23
Creating a Server-Side Calculated Member 23
Adding Formulas to a Member 37
Removing Formulas from a Member 39
Editing a Calculated Member 41
Account Types for Calculated Members 42
Resolving Conflicts between Dimensions 42
Virtual Child Members 46
Viewing Formula Information in SAS Financial Management Studio 46
Viewing Formula Information in Excel 49
Chapter 5 Formula Basics for Excel-Based Calculated Members 53
Introduction 54
Creating an Excel-Based Calculated Member 54
Removing an Excel-Based Calculated Member 59
Editing an Excel-Based Calculated Member 60
Viewing Formula Information for Excel-Based Calculated Members 63
Pivoting an Excel-Based Calculated Member 63
Resolving Conflicts between Excel-Based Calculated Members 65
Examples of Excel-Based Calculated Member Formulas 67
Chapter 6 Advanced Formula Concepts 85
Fixed Members 85
Formula Scope 92
Formula Scope and Member Selection Rules 99
Defining Multiple Formulas on One Member 103
Ranking Multiple Formulas 105

Chapter 7 Calculated Members Restrictions 109
Introduction 110
Ignoring Formulas on Rollup Members 110
Time Offsets: Formulas That Run Out of Bounds 112
Balance Account Calculated Members Scoped to Time 113
Referencing a Dimension and/or Member Not in the Model 115
Referencing Only Constants 118
Circular References 120
Dividing by Zero 122
Chapter 8 Statistical Account Behavior 127
Introduction 127
Statistical Accounts for Data Entry 128
Calculated Statistical Accounts 131
Chapter 9 Commonly Used Functions 135
ANCESTOR Function 136
CLOSINGPERIOD Function 138
CURRENT Function 138
DRATE Function 141
IF Function 142
NESTIF Function 143
OPENINGPERIOD Function 145
PARENT Function 146
PREVIOUS Function 146
PROPERTY Function 149
ROUND Function 153
SUM Function 154
VIRTUALCHILD Function 157
Adding Comments within a Formula Expression 157
Using String Functions in a Bracketed Member Reference 157
Using Dates and Numbers in Formula Syntax 158
Codes Containing Single and Double Quotation Marks 158
Chapter 10 Interactions with Load and Delete Data Options 159
Options for Loading Data 159
Loading Data and the Source Dimension 160
Loading Driver Formulas Facts to the Base Member of the Source Dimension 168
Deleting Data and Driver Facts 169
Chapter 11 Using Driver Formulas 171
Introduction 171
Range of Execution 171
What Triggers Execution of a Driver Formula 172
Form Design 173
Form Design Restrictions 173

Delete Behavior with Run driver formulas for this form set Option 173
Number of Driver Formulas in a Model 174
Chapter 12 Optimizing Formula Performance 175
Overview 175
Comparing the Execution of Driver Formulas and Modeling Formulas 175
Managing Formulas for Performance 178
Using CubeFormulasInfo and QueryStats Logging 181
Applying Performance Improvements 182
Chapter 13 Troubleshooting 193
Introduction 193
Troubleshooting in SAS Financial Management Studio 193
Troubleshooting in Microsoft Excel 194
Verifying Formula Results with the Excel Add-In 195
Troubleshooting Excel-based Calculated Members 198

CHAPTER

1 Introduction

Included in This Document 1
Additional Documentation 1
Example Data 1

Included in This Document

This document covers the following SAS Financial Management topics:

- formula types
- formula computations
- basic and advanced formula concepts
- formula restrictions
- statistical account behavior
- commonly used functions
- interactions with load and delete data options
- using driver formulas
- optimizing formula performance
- troubleshooting

This document is intended for process and budget administrators who have a fundamental understanding of accounting concepts and financial reporting, planning, and forecasting. It is assumed that the reader is familiar with the general concepts and capabilities of SAS Financial Management.

Additional Documentation

For additional information about using the SAS Financial Management clients (SAS Financial Management Studio, SAS Financial Management Add-In for Microsoft Excel and the web-based Financial Form Manager software), refer to *SAS Financial Management: Process Administrator's Guide* or the online Help for each client.

Example Data

The examples provided in this document were performed using SAS Financial Management 5.6. The query results are displayed in the SAS Financial Management Add-In for Microsoft Excel.

The examples use a standard set of dimensions, hierarchies, and members.

The dimensions used in the examples are as follows:

- Organization
- Account
- Analysis
- Currency
- Product
- Customer
- Time

The hierarchies are as follows:

Note: Bold text denotes roll-up points.

ORGANIZATION

- **Worldwide Operations**
  - **U.S. Operations**
    - **Sales**
      - Central
      - Eastern
      - Western
    - **Administration**
      - Corporate Comm
      - Facilities
      - Human Resources
    - **Finance**
      - Accounting
      - Corporate Legal
      - Payroll
      - Purchasing
    - **Product Delivery**
    - **Telemarketing**
    - **Technology**
      - Information Systems
      - Publications
      - Quality Assurance
      - Research
      - Technical Support
  - **European Operations**
    - European Sales
    - Admin Europe
    - HR Europe
    - IT Support - Europe
  - **Canada Operations**
    - Canada Sales
    - Admin Canada
    - HR Canada
    - IT Support - Canada
  - **Mexico Operations**
    - Mexico Sales
    - Admin Mexico
    - HR Mexico
    - IT Support - Mexico

ACCOUNT

- **Net Income**
  - **Income Tax**
  - **Income before Taxes**
    - **Operating Expense**
      - **Administrative Expense**
        - Office Supplies
        - Postage
        - Other Administrative Expenses
        - **Facilities**
          - Rent
          - Electric
          - Water
          - Repairs & Maintenance
          - Telecom
          - I/S Expenses
          - Other Facilities Expenses
      - **Marketing Expense**
        - Advertising
        - Promotions
        - Other Marketing Expenses
      - **Selling Expense**
        - **Staff Expenses**
          - Salaries
          - Benefits
          - Travel
          - Other Selling Expenses
      - **Other Operating Expense**
    - **Gross Profit**
      - Sales
      - Cost of Sales
- **Balance Sheet**
  - **Assets**
    - Cash and Cash Equivalents
    - **Net Accounts Receivable**
      - Accounts & Notes Receivable
      - Allowance for Doubtful Accounts
    - Inventory
    - Investment in Subs
  - **Liabilities**
    - Accounts Payable
    - Notes Payable
    - Other Accrued Liabilities
  - **Stockholder's Equity**
    - Common Stock
    - Additional Paid-in Capital
    - Retained Earnings
    - Pr Year Retained Earnings
    - Cumulative Translation Adjustment

Ending Headcount
Units Sold

ANALYSIS

Actual
Budget
Forecast

CURRENCY

USD
CAD
EUR
MXN

PRODUCT

Product Total
- **Video Games**
  - Action
  - Simulation
  - Arcade
  - Puzzle
- **Hardware**
  - Game Controller
  - Joy Stick
  - Flight Stick
- **Publications**
  - How to Use Simulator
  - Arcade Secrets
  - Puzzle Tricks

CUSTOMER

Customer Total
- **Partner**
  - HAL
  - Accents
  - Grand Hampton
- **Direct**
  - Buy Best
  - SW Mart
  - Radio City
- **Resellers**
  - Jones Distributing
  - Westco
- **Government**
  - DOD
  - Civilian
  - State & Local

TIME

- **All Years**
  - **2012**
    - **Q1 2012**
      - Jan 2012
      - Feb 2012
      - Mar 2012
    - **Q2 2012**
      - Apr 2012
      - May 2012
      - Jun 2012
    - **Q3 2012**
      - Jul 2012
      - Aug 2012
      - Sep 2012
    - **Q4 2012**
      - Oct 2012
      - Nov 2012
      - Dec 2012
  - **2013**
    - **Q1 2013**
      - Jan 2013
      - Feb 2013
      - Mar 2013
    - **Q2 2013**
      - Apr 2013
      - May 2013
      - Jun 2013
    - **Q3 2013**
      - Jul 2013
      - Aug 2013
      - Sep 2013
    - **Q4 2013**
      - Oct 2013
      - Nov 2013
      - Dec 2013

CHAPTER 2

Formula Types

Overview of Formula Types and Formulas 7
Reporting Formulas and Excel-Based Calculated Members 8
Introduction 8
Calculation Method 8
Currency Conversion Method 8
Fact Storage 9
Treatment of NaNs (Not a Number) 9
Dimension Type 9
Reuse 9
Choosing Whether to Use a Reporting Formula or an Excel-Based Calculated Member 9
Driver and Modeling Formulas 10
Introduction 10
Calculation Method 10
Currency Conversion Method 10
Fact Storage 10
Treatment of NaNs 10
Dimension Type 11
Reuse 11
Choosing Whether to Use a Driver Formula or a Modeling Formula 11

Overview of Formula Types and Formulas

SAS Financial Management supports four types of *formulas*: reporting formulas, modeling formulas, driver formulas, and Excel-based calculated member formulas. A formula is a mathematical expression that returns a value when it runs. A formula can be created on a *calculated member* in the Dimensions workspace in SAS Financial Management Studio.

Formula types derive their names based on when they run and where they are most commonly used in financial reporting and planning. Reporting formulas and Excel-based calculated member formulas run post-query. This means that the formulas typically are based on data inputs provided in a query. The most common examples are ratios such as Gross Profit Percentage and Earnings per Share.

Modeling formulas and driver formulas can be thought of as data-creation formulas; they generate data that can be consumed by reporting formulas and Excel-based calculated member formulas. These formulas also generate accounting logic such as Retained Earnings and Cumulative Translation Adjustment (CTA) accounts. For this reason, they are often termed pre-query formulas. Typical examples include Sales based on Price and Units, Estimated Benefits Expense based on Salaries, and Training Expense based on Headcount.

Choosing the formula type of a calculated member requires consideration of the following properties:

- calculation method
- currency conversion method
- fact storage
- treatment of NaNs (Not a Number)
- dimension type
- reuse

The following sections describe the various formula types and their properties.

Reporting Formulas and Excel-Based Calculated Members

Introduction

Reporting formulas and Excel-based calculated members are generally recommended for ratios and other values that do not need to be aggregated, such as key performance indicators.

Calculation Method

Reporting formulas and Excel-based calculated members run at each member in a hierarchy and do not distinguish between leaf members and roll-up points. The following illustrates their calculation behavior:

Dimension ABC	Dimension XYZ	Training Expense - RF
Parent Member A	Parent Member X	**Calculate formula**
	Leaf Member Y	**Calculate formula**
	Leaf Member Z	**Calculate formula**
Leaf Member B	Parent Member X	**Calculate formula**
	Leaf Member Y	**Calculate formula**
	Leaf Member Z	**Calculate formula**
Leaf Member C	Parent Member X	**Calculate formula**
	Leaf Member Y	**Calculate formula**
	Leaf Member Z	**Calculate formula**

Currency Conversion Method

Reporting formula and Excel-based calculated member results are converted using the currency and frequency specified on the table. This method implies that it is not necessary to determine functional currency values before returning formula results. This method is called the reporting currency approach. For more information, see Chapter 3, “Formula Computation.”

Fact Storage

Results for reporting formulas and Excel-based calculated members are computed at the time of a query. Their values are not stored in the database, but are generated on an as-needed basis.

Treatment of NaNs (Not a Number)

Invalid Reporting and Excel-based calculated member formula expressions result in a NaN (Not a Number) on a table when queried. For these two formula types, a calculated member that results in a NaN returns a red cell. Cell Information provides a message explaining why the query for a particular crossing failed. Examples of invalid formula expressions and calculated member restrictions are discussed further in Chapter 7, "Calculated Members Restrictions."

Dimension Type

Reporting and Excel-based calculated members can be created for the following dimension types:

- Account
- IntOrg
- Analysis
- Custom

In addition, Excel-based calculated members can be created on the Time dimension as well.

Reuse

Reporting formulas are centrally managed in the Dimensions workspace of SAS Financial Management Studio. Given appropriate security, reporting formula calculated members are available to many users because they are server-side calculations. As a result, the formula is considered reusable.

Excel-based calculated members are created and maintained in the Excel Add-In client. Because they are not server-side calculations, they are not considered reusable. They are available on a table-by-table basis only.

Choosing Whether to Use a Reporting Formula or an Excel-Based Calculated Member

The choice between a Reporting formula and an Excel-based calculated member should be based on the need for reuse and the dimension type of the calculated member. Excel-based calculated members are generally best for ad hoc calculations that are not required for ongoing reporting and analysis purposes. Reporting formulas are created on the server, so they're created by an Administrator and can be reused by models and cycles. Both formula types have minimal performance impact, but Excel-based calculated members are more efficient since they do not run on the server.

Driver and Modeling Formulas

Introduction

Driver formulas and modeling formulas are typically associated with calculations used in the budgeting and forecasting process. Examples include Estimated Sales based on Price and Units, Benefits Expense based on Salaries & Wages, and Training Expense based on Ending Headcount.

Calculation Method

Driver formulas and modeling formulas run at leaf members, and results are summed at roll-up points. The following illustrates their calculation behavior:

Dimension ABC	Dimension XYZ	Training Expense - DF
Parent Member A	Parent Member X	Sum of calculations
	Leaf Member Y	Sum of calculations
	Leaf Member Z	Sum of calculations
Leaf Member B	Parent Member X	Sum of calculations
	Leaf Member Y	**Calculate formula**
	Leaf Member Z	**Calculate formula**
Leaf Member C	Parent Member X	Sum of calculations
	Leaf Member Y	**Calculate formula**
	Leaf Member Z	**Calculate formula**

Currency Conversion Method

Driver formula inputs and modeling formula inputs are converted using the functional currency of the Organization member and the PTD frequency. Then, the PTD results are converted to PA results, and the PA results are finally converted using the currency and frequency as specified in the table. This method is called the functional currency approach. For more information about the functional currency approach, see Chapter 3, “Formula Computation.”

Fact Storage

Results for driver formulas are stored in the database as facts for the BaseForm member of the Source dimension. Results for modeling formulas are computed as needed at the time of a query. They are not stored in the database.

Treatment of NaNs

For invalid formula expressions on driver and modeling formula types, query results and available Cell Information differ significantly. Modeling formulas perform similarly to reporting formulas and Excel-based calculated members in that the crossing displays

a red cell and Cell Information provides a message explaining why the query for a particular crossing failed.

To minimize the number of NaN facts generated and stored for Driver formulas, the query results for a driver formula type with an invalid formula expression return gray, non-writeable cells with a value of zero displayed. Also, Cell Information says that the query succeeded.

Dimension Type

Driver formulas and modeling formulas are available only in the Account dimension.

Reuse

Driver formulas and modeling formulas are centrally managed in the Dimensions workspace of SAS Financial Management Studio. Given appropriate security, these calculated members are available to many users because they are server-side calculations. As a result, driver and modeling formula types are considered reusable.

Choosing Whether to Use a Driver Formula or a Modeling Formula

The choice between a driver formula and a modeling formula should be based on whether significant changes are expected to the formula expression inputs, exchange rates, and driver rates. If minimal change is expected, you should assign calculated members as driver formulas to maximize formula performance.

CHAPTER 3

Formula Computation

Order of Execution *13*
Formulas *14*
Facts *14*
Rates *14*
Results *15*
Functional Currency Approach *17*
Formula *17*
Facts *17*
Rates *18*
Step 1 ..... *18*
Step 2 ..... *18*
Step 3 ..... *19*
Step 4 ..... *19*
Reporting Currency Approach *20*
Formula *20*
Facts *20*
Rates *20*
Step 1 ..... *21*
Step 2 ..... *21*

Order of Execution

Introduction

The order of execution is as follows:

1. Facts
2. Intercompany eliminations
3. Driver formulas
4. Modeling formulas
5. Retained Earnings and Cumulative Translation Adjustment accounts
6. Reporting formulas

This execution order allows both modeling formulas and driver formulas to be indirectly referenced or directly referenced as source accounts of Retained Earnings accounts and Cumulative Translation Adjustment accounts. Reporting formulas are calculated after Retained Earnings accounts and Cumulative Translation Adjustment accounts. Therefore, reporting formulas do not contribute to the results of Retained Earnings accounts and Cumulative Translation Adjustment accounts.

Simply stated, each level can consume its own level as well as the level(s) before it. Note that currency conversion occurs at each level of execution.

To illustrate execution order, the following example includes a reporting formula, a modeling formula, a driver formula, and an Excel-based calculated member. The formulas, facts, rates, and results are as follows:

Formulas

Reporting Formula (RF):
Training Expense - RF = ["ACCOUNT"="Ending Headcount"]*200

Modeling Formula (MF):
Training Expense - MF = ["ACCOUNT"="Ending Headcount"]*300

Driver Formula (DF):
Training Expense - DF = ["ACCOUNT"="Ending Headcount"]*400

Excel-based Calculated Member (XB):
Training Expense - XB =fmValue("Ending Headcount")*500

Facts

The following PTD facts have no currency assigned (NONE) for Ending Headcount. The Ending Headcount is a Statistical Balance account type.

Jan 2013	2.0
Feb 2013	2.5
Mar 2013	3.0

Rates

The following Period Average rates apply for currency conversion from CAD to USD:

Jan 2013	0.90
Feb 2013	0.91
Mar 2013	0.92

The following Period Close rates apply for currency conversion from CAD to USD:

Jan 2013	0.895
Feb 2013	0.905
Mar 2013	0.915

Results

In CAD, the results are displayed as follows:

Organization HR Canada
Product Product Total
Customer Customer Total
Analysis Budget
Currency CAD
Frequency PTD

	Jan 2013	Feb 2013	Mar 2013	Q1 2013
Net Income	1,400.00	1,750.00	2,100.00	5,250.00
Income before Taxes	1,400.00	1,750.00	2,100.00	5,250.00
Operating Expense	1,400.00	1,750.00	2,100.00	5,250.00
Administrative Expense	1,400.00	1,750.00	2,100.00	5,250.00
Training Expense - RF	400.00	500.00	600.00	600.00
Training Expense - MF	600.00	750.00	900.00	2,250.00
Training Expense - DF	800.00	1,000.00	1,200.00	3,000.00
Balance Sheet	1,400.00	3,150.00	5,250.00	5,250.00
Stockholder's Equity	1,400.00	3,150.00	5,250.00	5,250.00
Retained Earnings	1,400.00	3,150.00	5,250.00	5,250.00
Ending Headcount	2.00	2.50	3.00	3.00

In this table, note the following:

- Roll-up points and Retained Earnings do not include the results of the Training Expense - RF and Training Expense - XB members.
- Results for the Training Expense - RF and Training Expense - XB members for Q1 2013 are calculated using formulas. (This is not realistic behavior.)
- The Source account to the Retained Earnings account member is Net Income.
- The Source account to the Cumulative Translation Adjustment account member is Stockholder's Equity.

When converted to USD, the results are as follows:

Organization	HR Canada
Product	Product Total
Customer	Customer Total
Analysis	Budget
Currency	USD
Frequency	PTD

	Jan 2013	Feb 2013	Mar 2013	Q1 2013
Net Income	1,260.00	1,592.50	1,932.00	4,784.50
Income before Taxes	1,260.00	1,592.50	1,932.00	4,784.50
Operating Expense	1,260.00	1,592.50	1,932.00	4,784.50
Administrative Expense	1,260.00	1,592.50	1,932.00	4,784.50
Training Expense - RF	400.00	500.00	600.00	600.00
Training Expense - MF	540.00	682.50	828.00	2,050.50
Training Expense - DF	720.00	910.00	1,104.00	2,734.00
Balance Sheet	1,253.00	2,850.75	4,803.75	4,803.75
Stockholder's Equity	1,253.00	2,850.75	4,803.75	4,803.75
Retained Earnings	1,260.00	2,852.50	4,784.50	4,784.50
Cumultative Translation Adjustement	(7.00)	(1.75)	19.25	19.25
Ending Headcount	2.00	2.50	3.00	3.00

In this table, note the following:

- The Training Expense - RF and Training Expense - XB member results display the same values in CAD and USD due to the use of the reporting currency approach.
- The Cumulative Translation Adjustment account is computed as follows:

  Jan 2013

  (1400.00 * 0.895) - (1400.00 * 0.90) = 1253.00 - 1260.00 = (7.00)

  Feb 2013

  ((1400+1750) * 0.905) - ((1400 * 0.9) + (1750 * 0.91)) = 2850.75 -2852.50 = (1.75)

  Mar 2013 and Q1 2013

  ((1400+1750+2100) * 0.915) - ((1400 * 0.9) + (1750 * 0.91) + (2100 * 0.92)) = 4803.75 - 4784.50 = 19.25

Note: The assigned formula type for the calculated member determines the assumed currency of constants and DRATEs referenced in the formula expression:

- the IntOrg's functional currency for modeling formulas and driver formulas
- the currency specified by the query for reporting formulas and Excel-based calculated members

Functional Currency Approach

Introduction

In versions prior to SAS Financial Management 4.4, all formula results were computed using the currency and frequency as specified in the table. Beginning with SAS Financial Management 4.4, only reporting formulas and Excel-based calculated members continue to use this currency conversion method. Modeling formulas and driver formulas now use the functional currency approach. Modeling formula and driver formula calculations always take place in the functional currency of the IntOrg member. The results are then converted to the reporting currency, which might or might not be the same as the functional currency. The reporting currency is the currency that is specified in the table, either explicitly or through the table default read member.

Four Steps

The functional currency approach has four steps to accurately calculate and currency convert modeling formulas and driver formulas:

- **Step 1:** Convert data input(s) to the functional currency using the PTD frequency.
- **Step 2:** Solve for the calculated member in the functional currency using the PTD frequency.
- **Step 3:** Solve for the Period Activity frequency in the functional currency based on the PTD results that are derived in Step Two.
- **Step 4:** Convert results to the reporting currency and frequency as specified in the table based on the Period Activity results that are derived in Step Three.

Note: Any constants are considered to be in the functional currency of the IntOrg member that they were entered on.

To illustrate these four steps, we use the following example with a driver formula. The formula, facts, rates, and results are given below.

Formula

Driver Formula:
Allowance for Doubtful Accounts = ["ACCOUNT"="Accounts & Notes Receivable"]*["ACCOUNT"="Doubtful Accounts %"]

Facts

The following PTD facts are in USD for Accounts Receivable—a balance account type. The facts are *not* in the functional currency of the Admin Canada organization member.

Dec 2012	100
Jan 2013	200
Feb 2013	250
Mar 2013	400

Rates

The following Period Close rates apply for currency conversion from EUR to USD:

Dec 2012	0.76
Jan 2013	0.77
Feb 2013	0.78
Mar 2013	0.79

Step 1

Convert data input(s) to the functional currency using the PTD frequency.

Organization	European Sales
Product	Product Total (vc)
Customer	Customer Total (vc)
Analysis	Budget
Currency	EUR
Frequency	PTD
Source	Form Data
TRADER	External

	Dec 2012	Jan 2013	Feb 2013	Mar 2013	Q1 2013
Accounts & Notes Receivable	131.58	259.74	320.51	506.33	506.33
Doubtful Accounts %	(0.25)	(0.25)	(0.25)	(0.25)	

Step 2

Solve for the calculated member in the functional currency using the PTD frequency.

Organization	European Sales
Product	Product Total (vc)
Customer	Customer Total (vc)
Analysis	Budget
Currency	EUR
Frequency	PTD
Source	Form Data
TRADER	External

	Dec 2012	Jan 2013	Feb 2013	Mar 2013	Q1 2013
Accounts & Notes Receivable	131.58	259.74	384.62	506.33	506.33
Allowance for Doubtful Accounts	(32.89)	(64.94)	(96.15)	(126.58)	(126.58)
Doubtful Accounts %	(0.25)	(0.25)	(0.25)	(0.25)	

Step 3

Solve for the Period Activity frequency in the functional currency based on the PTD results that are derived in Step 2.

Organization ... European Sales
Product ... Product Total (vc)
Customer ... Customer Total (vc)
Analysis ... Budget
Currency ... EUR
Frequency ... PA
Source ... Form Data
TRADER ... External

	Dec 2012	Jan 2013	Feb 2013	Mar 2013	Q1 2013
Accounts & Notes Receivable	131.58	128.16	124.88	121.71	374.75
Allowance for Doubtful Accounts	(32.89)	(32.04)	(31.22)	(30.43)	(93.69)
Doubtful Accounts %	(0.25)	(0.25)	(0.25)	(0.25)	

Step 4

Convert results to the reporting currency and frequency as specified in the table based on the Period Activity results that are derived in Step 3.

Organization ... European Sales
Product ... Product Total
Customer ... Customer Total
Analysis ... Budget
Currency ... USD
Account ... Allowance for Doubtful Accounts

	Dec 2012	Jan 2013	Feb 2013	Mar 2013	Q1 2013
PTD	(25.00)	(50.00)	(75.00)	(100.00)	(100.00)
LTD	(25.00)	(50.00)	(75.00)	(100.00)	(100.00)
YTD	(25.00)	(50.00)	(75.00)	(100.00)	(100.00)
QTD	(25.00)	(50.00)	(75.00)	(100.00)	(100.00)
MTD	(25.00)	(50.00)	(75.00)	(100.00)	
PA	(25.00)	(24.67)	(24.35)	(24.04)	(74.01)
LA	(25.00)	(50.00)	(75.00)	(100.00)	(100.00)
YA	(25.00)	(24.67)	(49.34)	(74.01)	(74.01)
QA	(25.00)	(24.67)	(49.34)	(74.01)	(74.01)
MA	(25.00)	(24.67)	(24.35)	(24.04)	

Reporting Currency Approach

The reporting currency approach is used by reporting formulas and Excel-based calculated members only. There basically are two steps required to calculate and convert reporting formulas and Excel-based calculated members:

- **Step 1:** Convert data input(s) to the currency and frequency as specified in the table.
- **Step 2:** Solve for the calculated member in the currency and frequency as specified in the table.

To illustrate these steps, we use an example of the calculation and conversion of a reporting formula. The formula, facts, rates, and results are as follows:

Formula

Reporting Formula:
Gross Margin Percentage = ["ACCOUNT"="Gross Margin”]/ ["ACCOUNT"="Sales”]

Facts

The following table displays the facts for Sales and Cost of Sales, which are both flow account types. The facts were entered in CAD, which is the functional currency for the Canada Sales organization member.

Organization	Canada Sales
Product	Product Total
Customer	Customer Total
Analysis	Budget
Currency	CAD
Frequency	YTD

	Dec 2012	Jan 2013	Feb 2013	Mar 2013	Q1 2013
Gross Profit	(35,000.00)	(1,800.00)	(4,150.00)	(7,250.00)	(7,250.00)
Sales	(90,000.00)	(5,000.00)	(11,000.00)	(19,500.00)	(19,500.00)
Cost of Sales	55,000.00	3,200.00	6,850.00	12,250.00	12,250.00

Rates

The following Period Average rates apply for currency conversion from CAD to USD:

Dec 2012	0.89
Jan 2013	0.90
Feb 2013	0.91
Mar 2013	0.92

Step 1

Convert data input(s) to the currency and frequency as specified in the table.

Organization ... Canada Sales
Product ... Product Total
Customer ... Customer Total
Analysis ... Budget
Currency ... USD
Frequency ... YTD

	Dec 2012	Jan 2013	Feb 2013	Mar 2013	Q1 2013
Gross Profit	(31,150.00)	(1,620.00)	(3,758.50)	(6,610.50)	(6,610.50)
Sales	(80,100.00)	(4,500.00)	(9,960.00)	(17,780.00)	(17,780.00)
Cost of Sales	48,950.00	2,880.00	6,201.50	11,169.50	11,169.50

Step 2

Solve for the calculated member in the currency and frequency as specified in the table.

Organization ... Canada Sales
Product ... Product Total
Customer ... Customer Total
Analysis ... Budget
Currency ... USD
Frequency ... YTD

	Dec 2012	Jan 2013	Feb 2013	Mar 2013	Q1 2013
Gross Profit	(31,150.00)	(1,620.00)	(3,758.50)	(6,610.50)	(6,610.50)
Sales	(80,100.00)	(4,500.00)	(9,960.00)	(17,780.00)	(17,780.00)
Cost of Sales	48,950.00	2,880.00	6,201.50	11,169.50	11,169.50
Gross Margin Percentage	0.39	0.36	0.38	0.37	0.37

CHAPTER

4

Formula Basics for Server-Side Calculated Members

Introduction *23*
Creating a Server-Side Calculated Member *23*
Adding Formulas to a Member *37*
Removing Formulas from a Member *39*
Editing a Calculated Member *42*
Account Types for Calculated Members *43*
Resolving Conflicts between Dimensions *43*
Introduction *43*
Scenario 1: Account Dimension Has Higher Rank than Analysis Dimension *45*
Scenario 2: Analysis Dimension Has Higher Rank than Account Dimension *46*
Virtual Child Members *47*
Viewing Formula Information in SAS Financial Management Studio *47*
Viewing Formula Information in Excel *50*

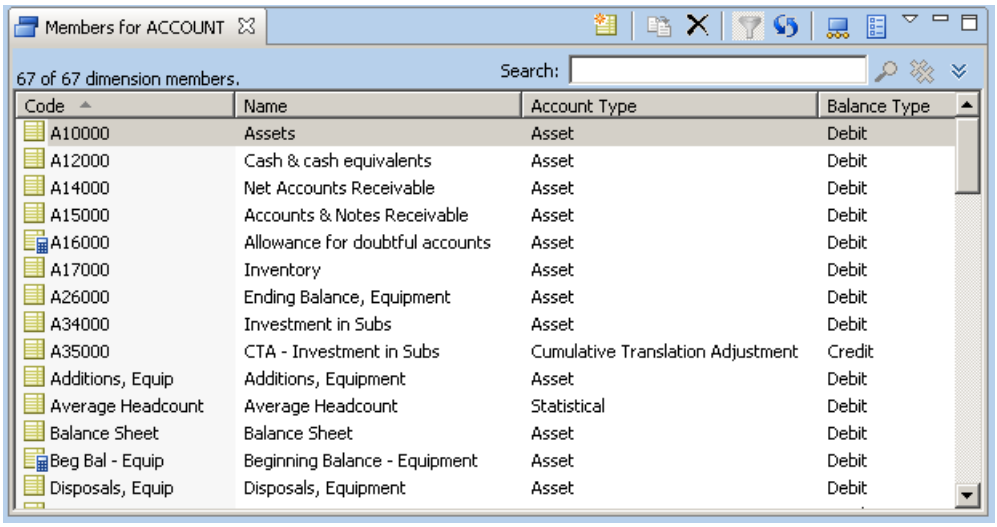
Introduction

This chapter covers the fundamentals for server-side formula creation and management as well as other relevant information.

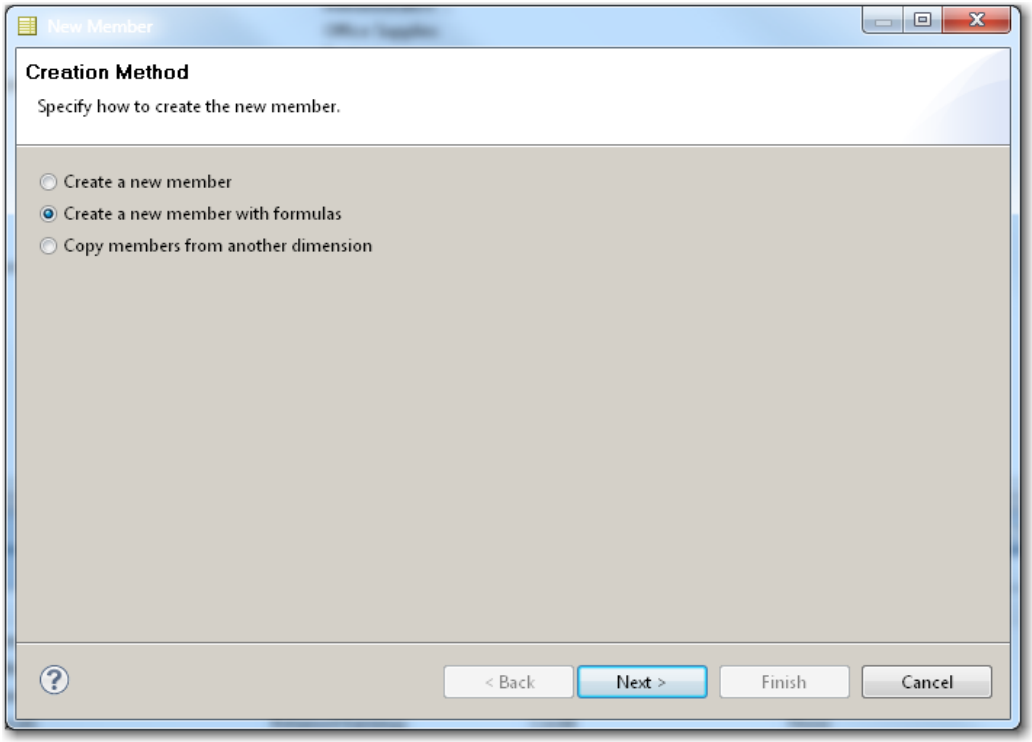
Creating a Server-Side Calculated Member

In SAS Financial Management Studio, you can create reporting formulas, modeling formulas, and driver formulas. You use the same action to create a new member with or without formulas. The following steps illustrate the creation of a calculated member in the Account dimension. Note that only the Account dimension offers three formula types. In all other dimensions, only reporting formulas are available. Also, note that the Account dimension includes additional member property windows such as Account Type and Intercompany.

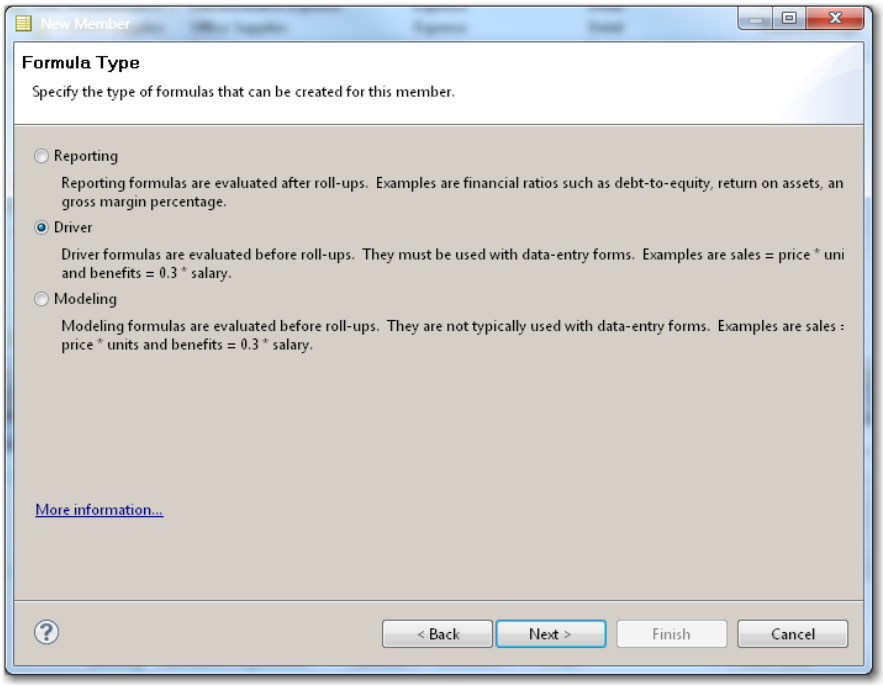
To create a new member in the Account dimension, click Add a member to this hierarchy on the **Hierarchies** tab or **Create a new member** on the **Members** tab (shown below).



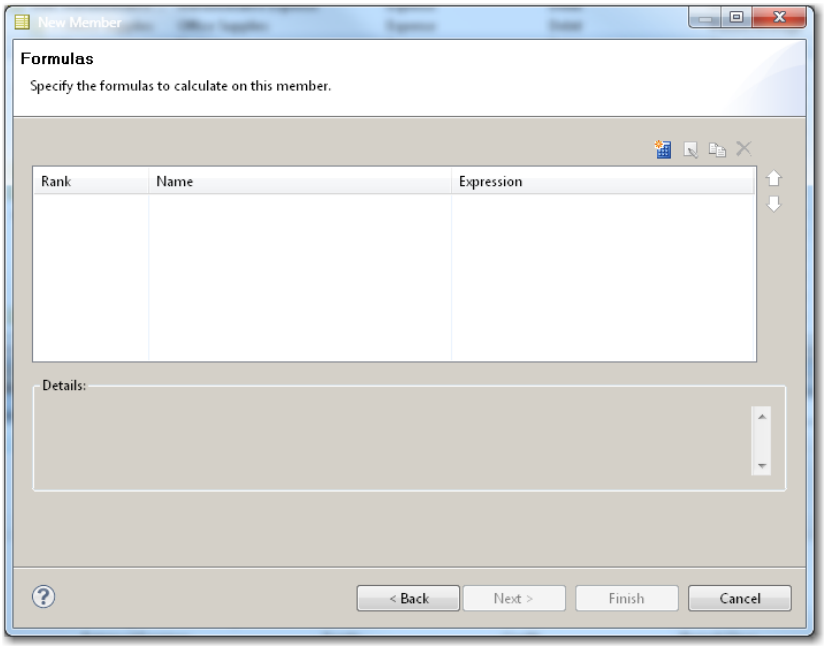
1. Select **Create a new member with formulas** and click **Next**.



2. After completing the required member property information, select the desired formula type and click **Next**.

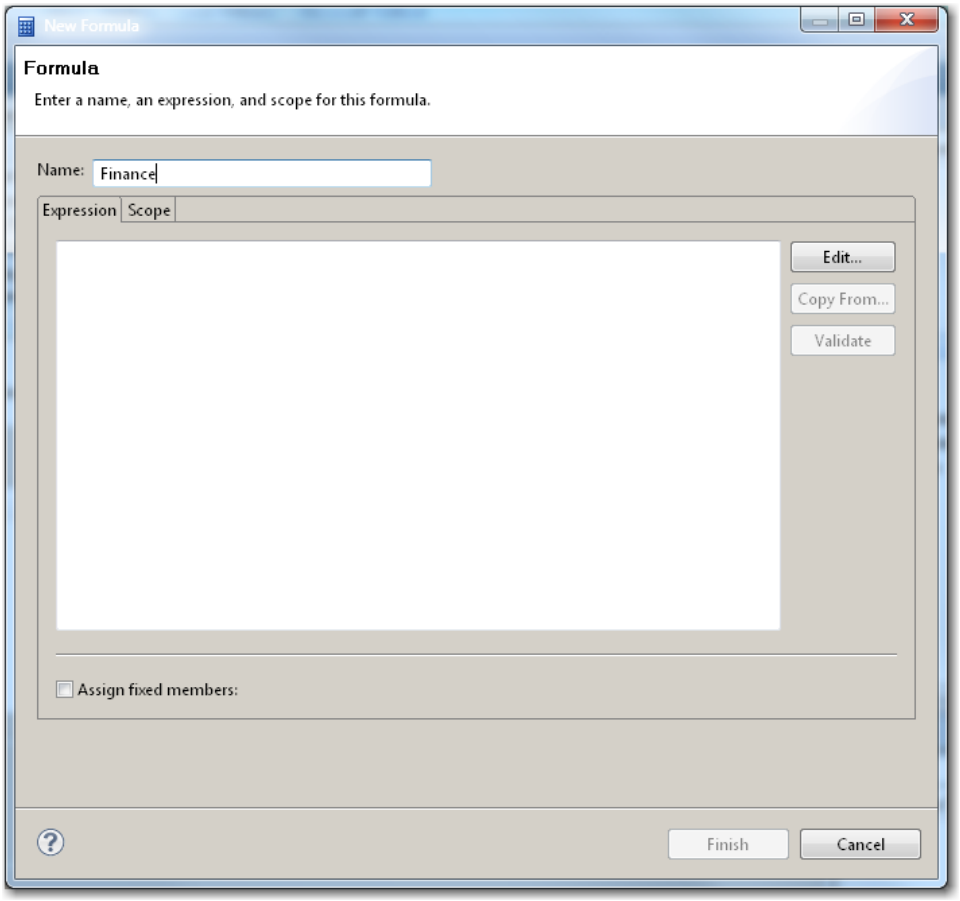


3. On the Formulas page, click the Create a new formula icon.



4. On the Formula page, name the formula.

The name given here is displayed via **Cell Information** in the Excel Add-In. Click **Edit** to open the Edit Expression page.



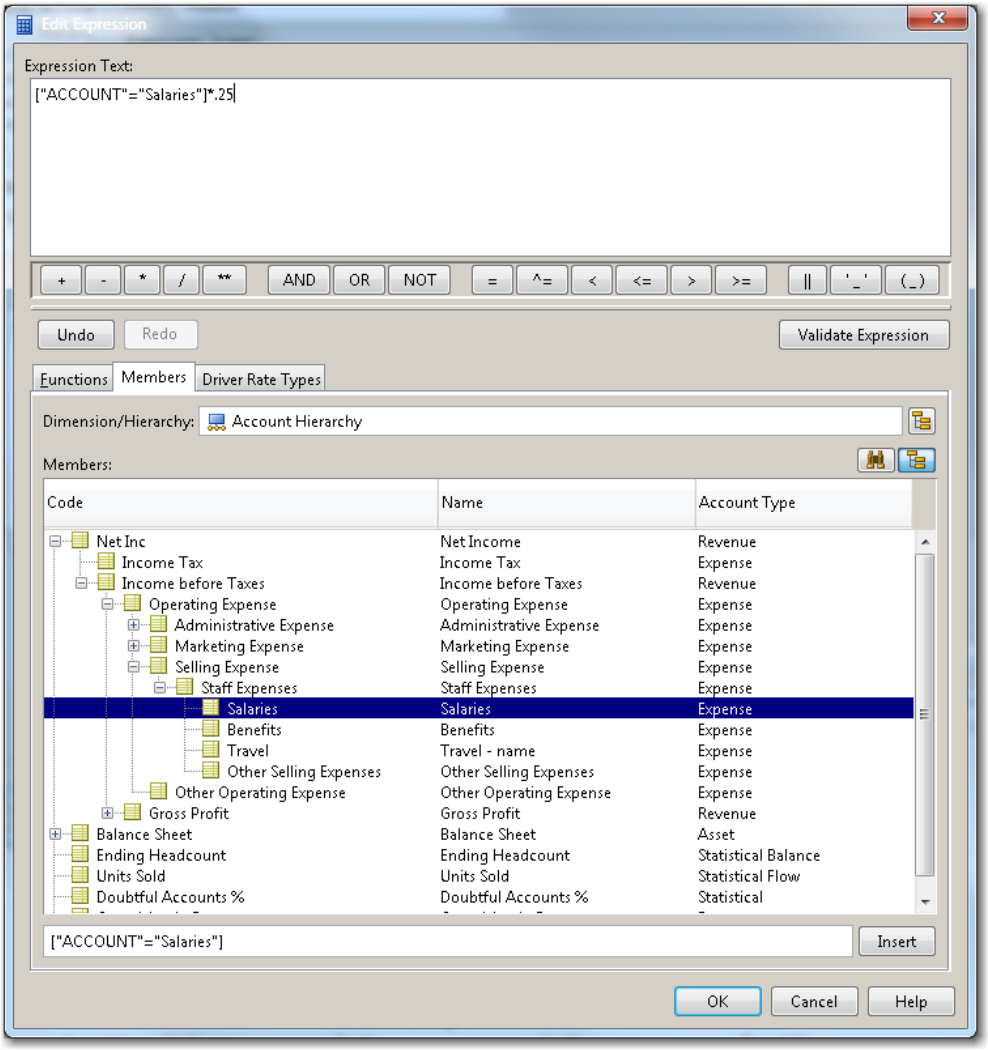
The Edit Expressions window has three tabs for creating and managing formula expressions:

- The **Functions** tab provides a comprehensive list of functions that are available in SAS Financial Management.
- The **Members** tab is the default tab that is displayed when opening the Formula Editor window. It provides the current hierarchical view of members based on the selected dimension and hierarchy.
- The **Driver Rate Types** tab offers a separate workspace for selecting rates.

Use these tabs to enter the expression text.

5. Highlight the member(s), function(s), and rate(s) to be included. Click **Insert**.

You can also type the formula expression directly into the **Expression Text** region.

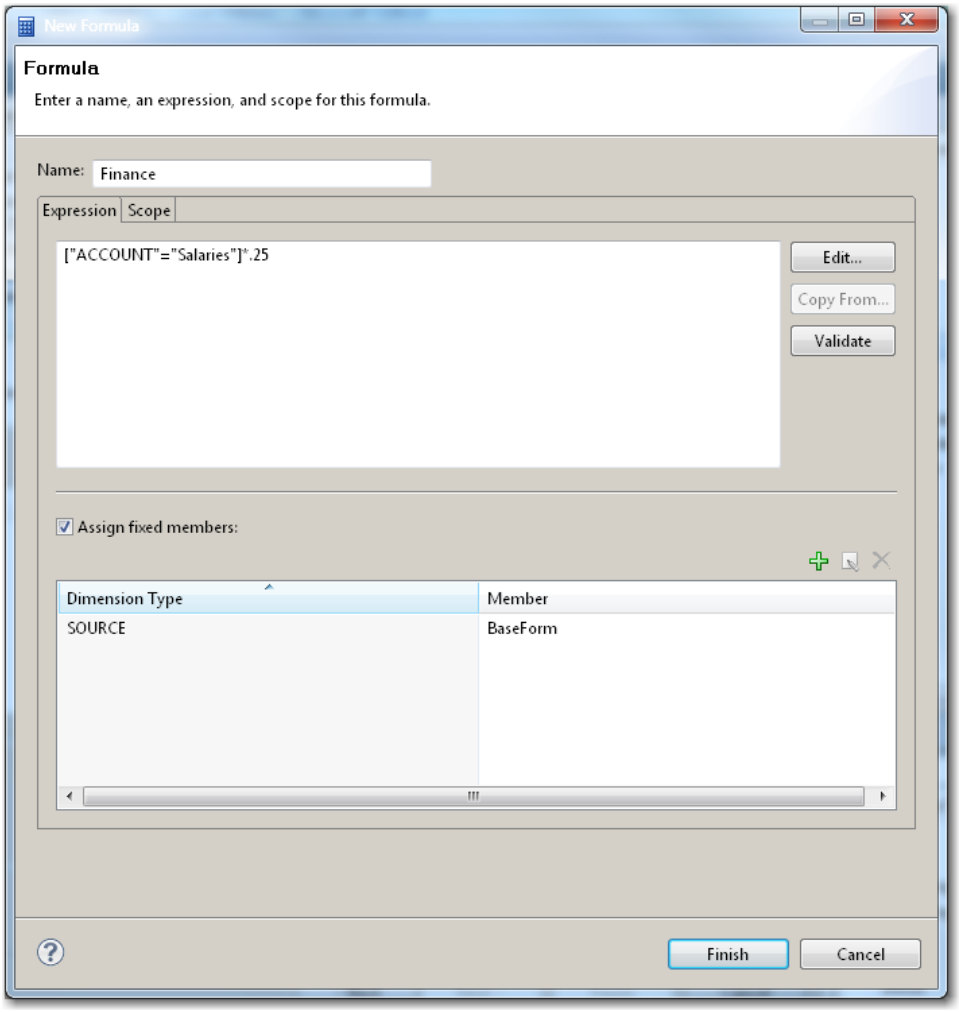


6. Click **Validate Expression** or **OK** to validate that the expression is correct.

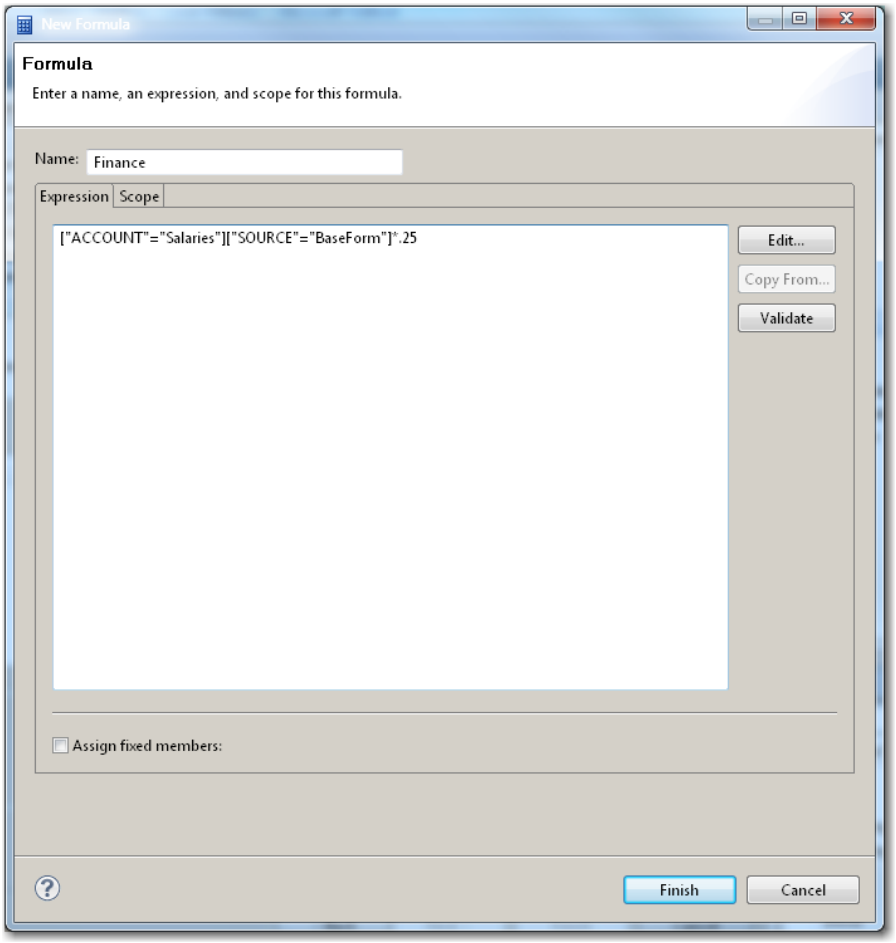
   Upon validation, you can assign fixed members to the formula expression. Fixed members are considered an extension of a formula expression. The selection of a default member is applied universally to all of the formula inputs for the current formula expression.

   To illustrate, you can create a formula that applies only to the BaseForm Source member. There are two ways to write the formula, with and without fixed members:

   Option 1: With Fixed Members

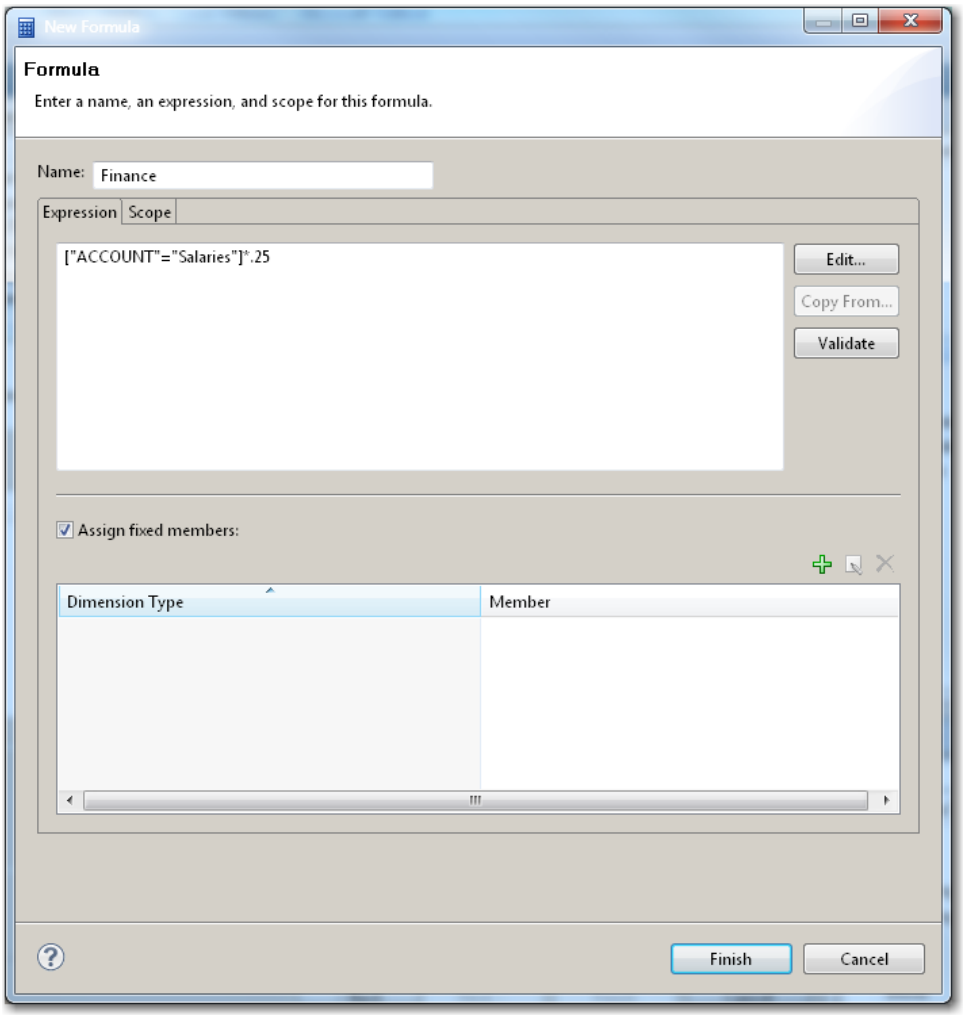


Option 2: Without Fixed Members



The two options return identical results. The only difference is in the setup and maintenance of the formula expression.

7. To use fixed members, select the **Assign fixed members** check box.

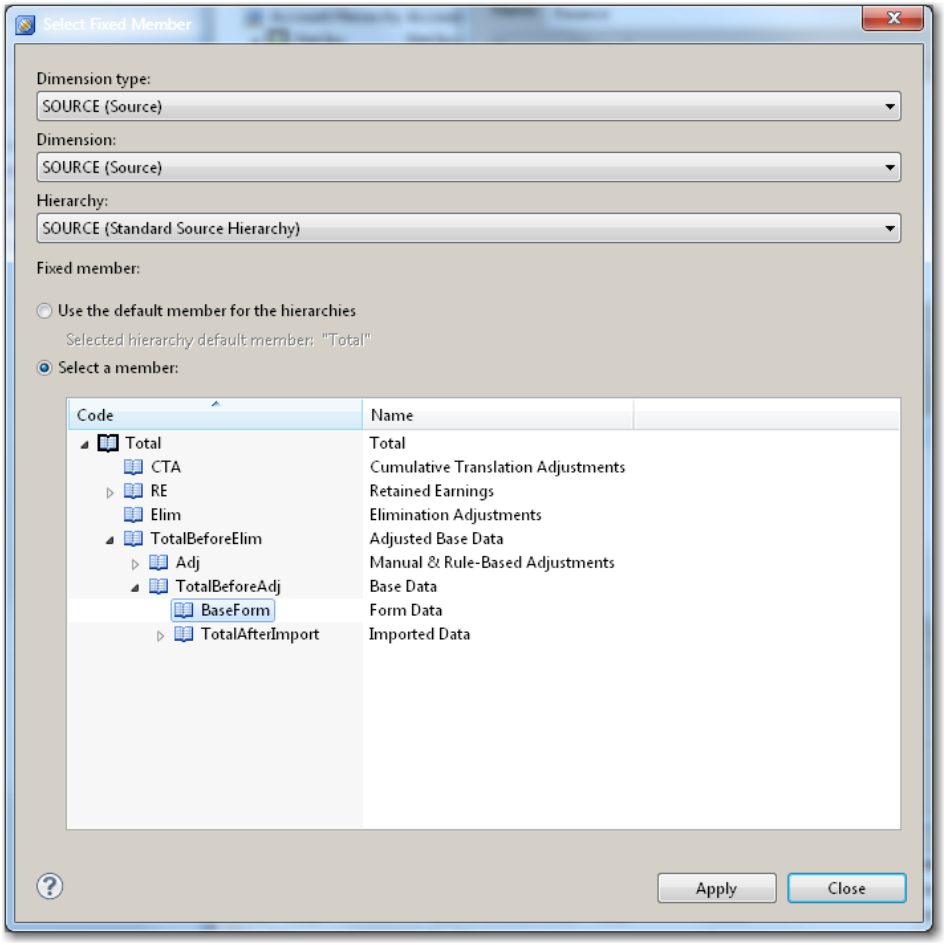


8. Click the green plus icon to select a fixed member.
9. Select the desired dimension type, dimension, and hierarchy.

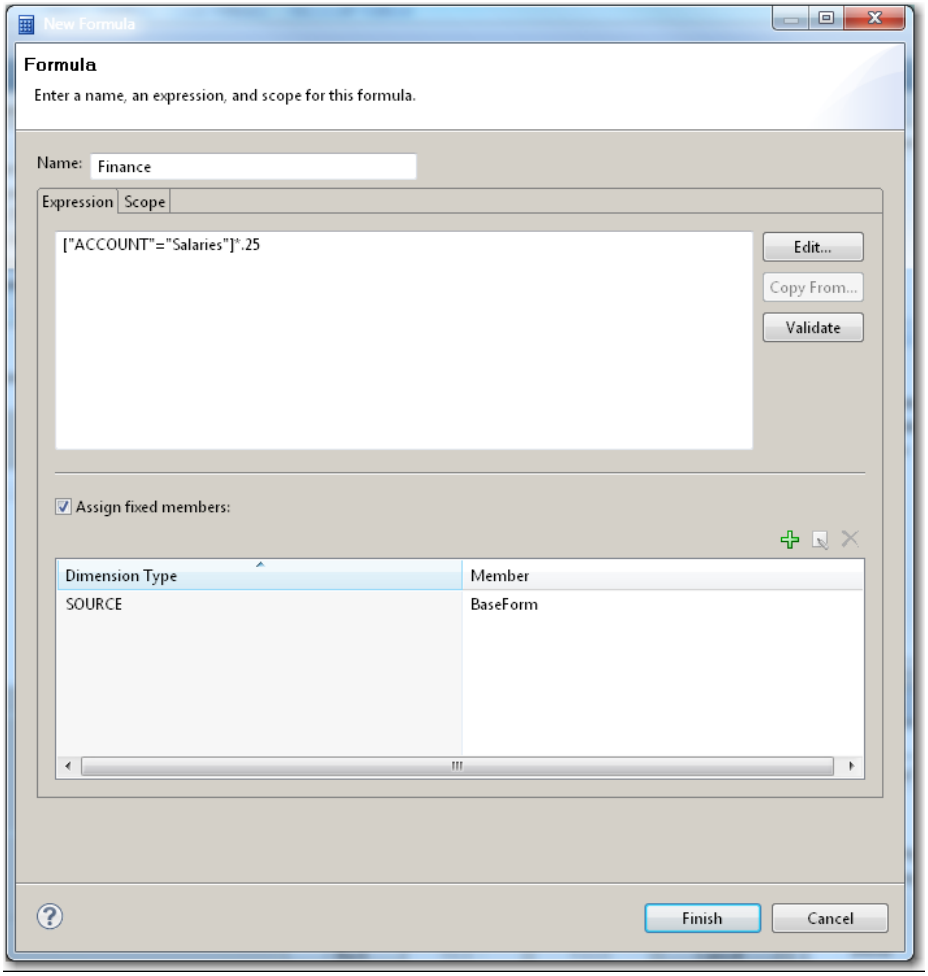
   Members are displayed in the context of the current view of the selected dimension and hierarchy.
10. Select the default member of a given hierarchy or a different member.

    Either option allows for only one default member per dimension type.

11. Highlight the desired member and click **Apply**.



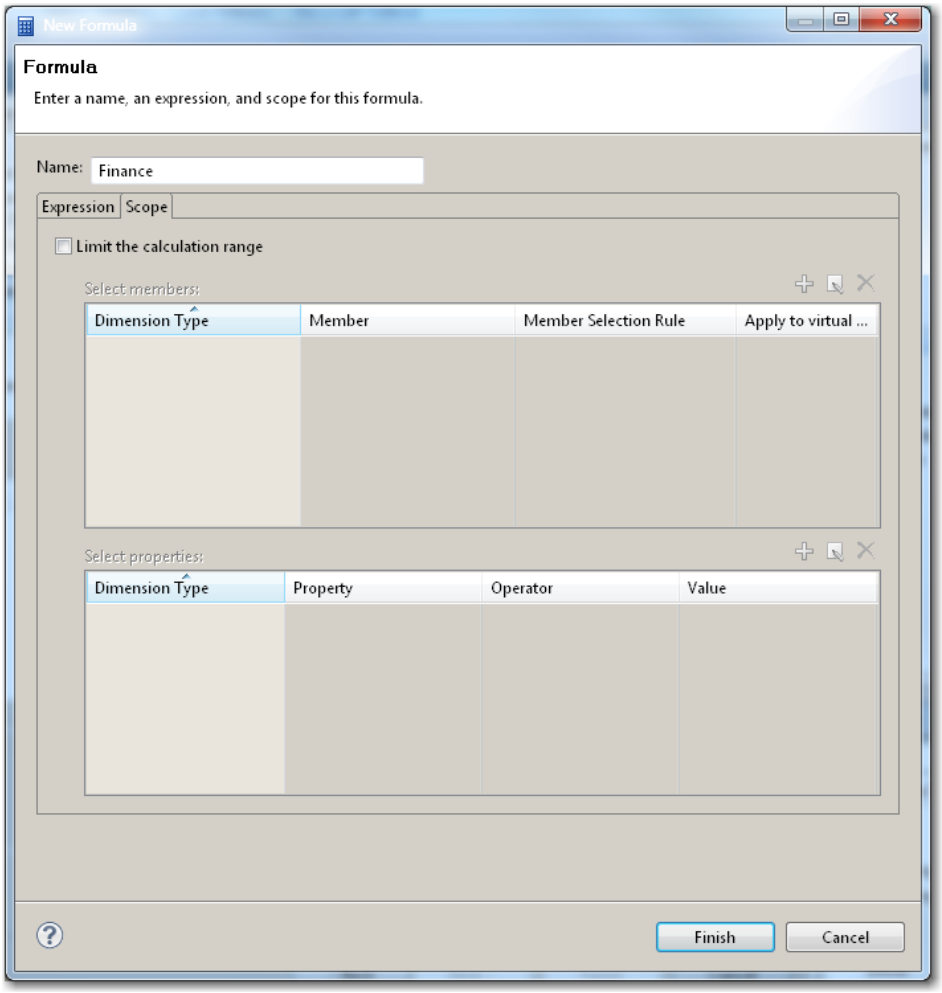
The member selected is displayed on the **Expression** tab. For more information about the use of fixed members with calculated members, see Chapter 6, "Advanced Formula Concepts."



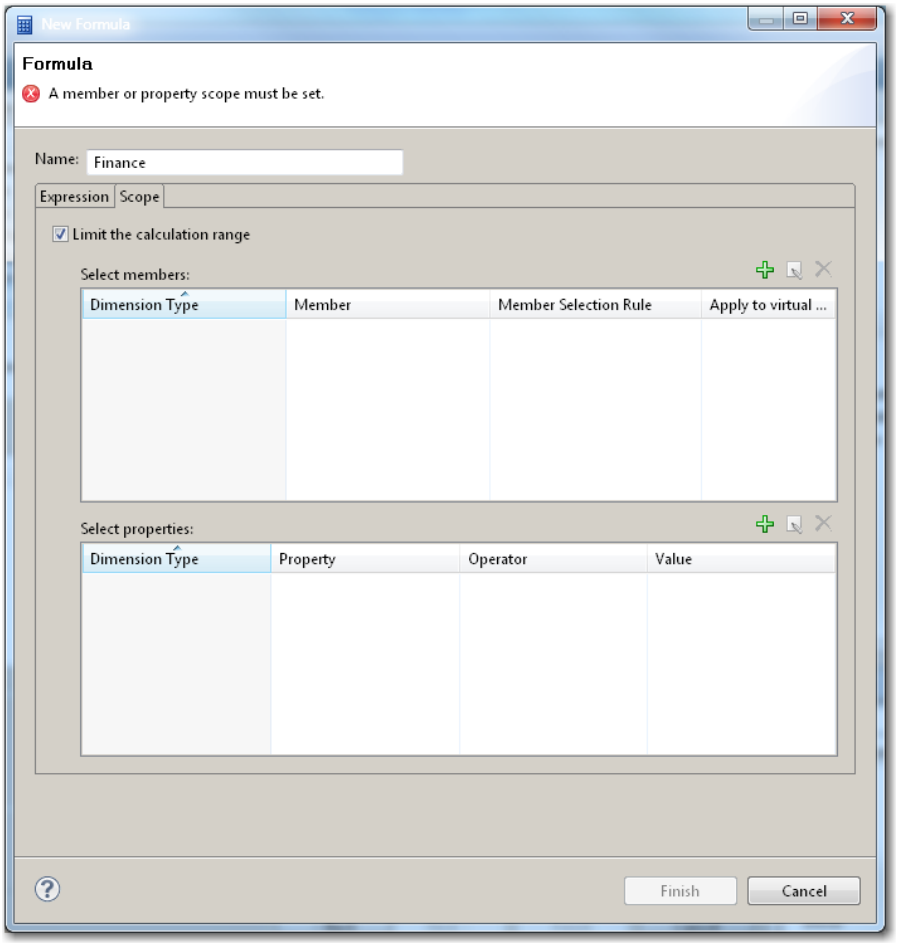
12. Click **Next** to advance to the **Scope** tab.

By specifying a scope for a formula, you can restrict the set of crossings where the formula runs. Depending on the formula type, this can improve formula performance. It is also important when you define multiple formulas on the same member. For details about defining multiple formulas on a member, see Chapter 6, "Advanced Formula Concepts."

Scoping can optionally be applied to dimension members, properties associated with members, or both.



13. To use the scope functionality, select the Limit the calculation range check box.

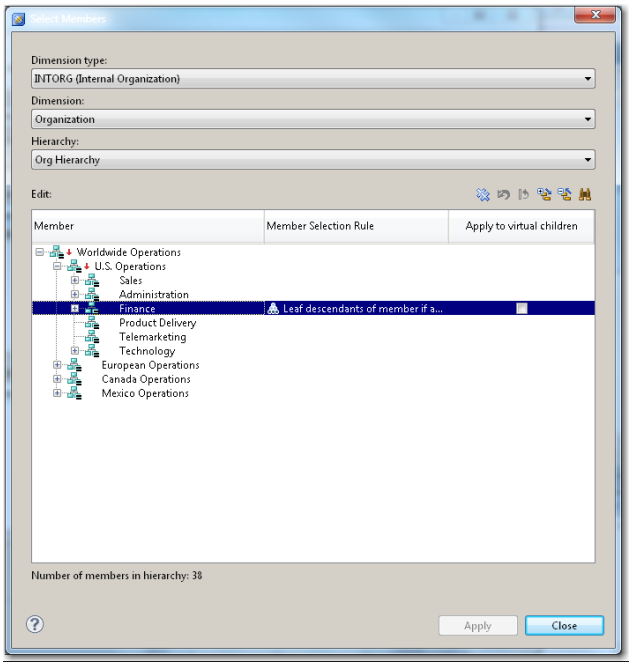


14. Similar to fixed members, click the green plus icon to select members or properties for scoping.

For member scoping, select the desired dimension type, dimension, and hierarchy.

Members are displayed in the context of the current view of the selected dimension and hierarchy. You can select from the available hierarchies and choose multiple members in each dimension.

15. Highlight the desired member(s) and select the desired Member Selection Rule associated with the member as well as whether to apply to the virtual children.



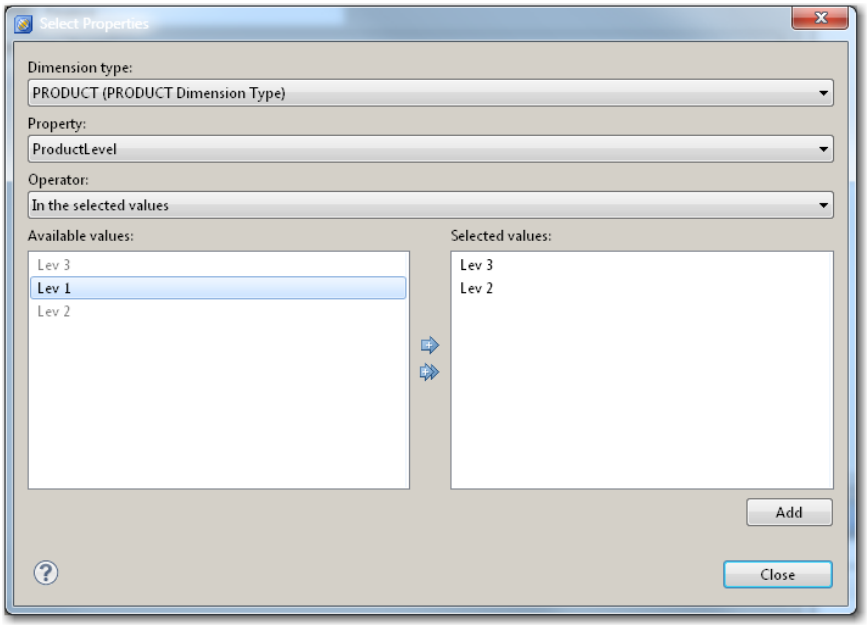
16. Click **Apply** and **Close** when member scoping is complete.

The results of specifying member scope for the formula are displayed on the **Scope** tab. You can specify additional members or select properties to scope by.

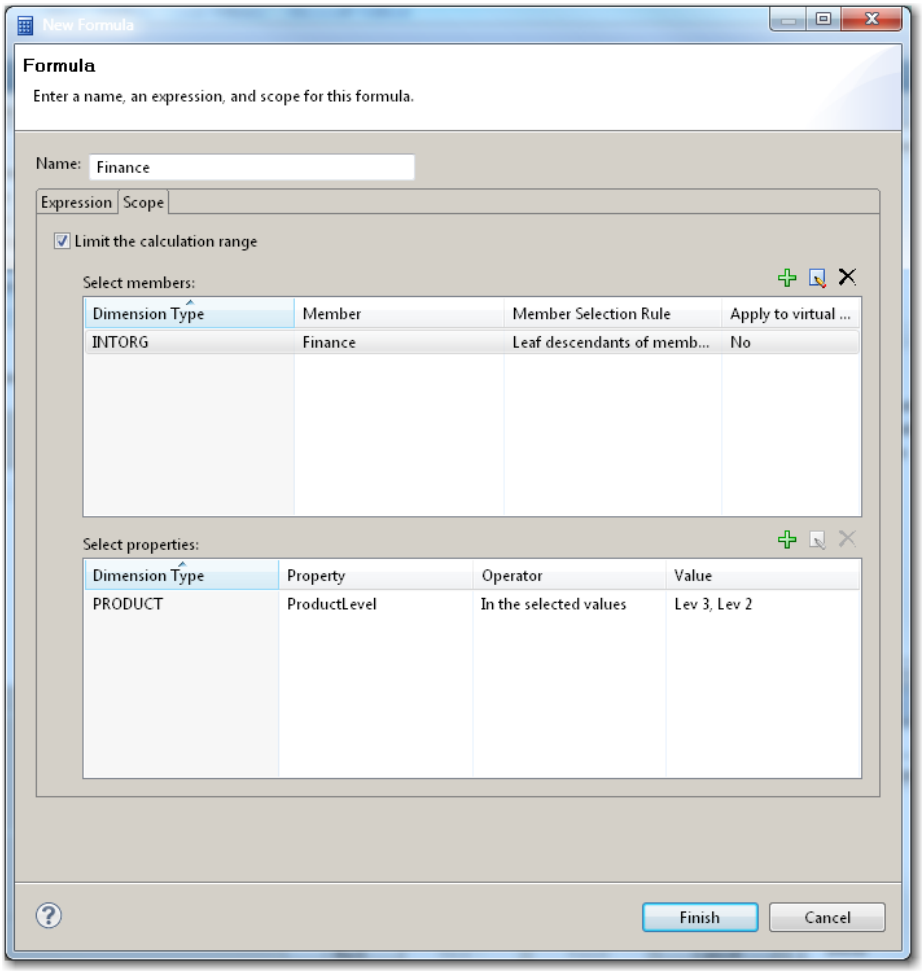
17. To select properties, click the green plus icon.

18. Select the desired dimension type, property, and operator.

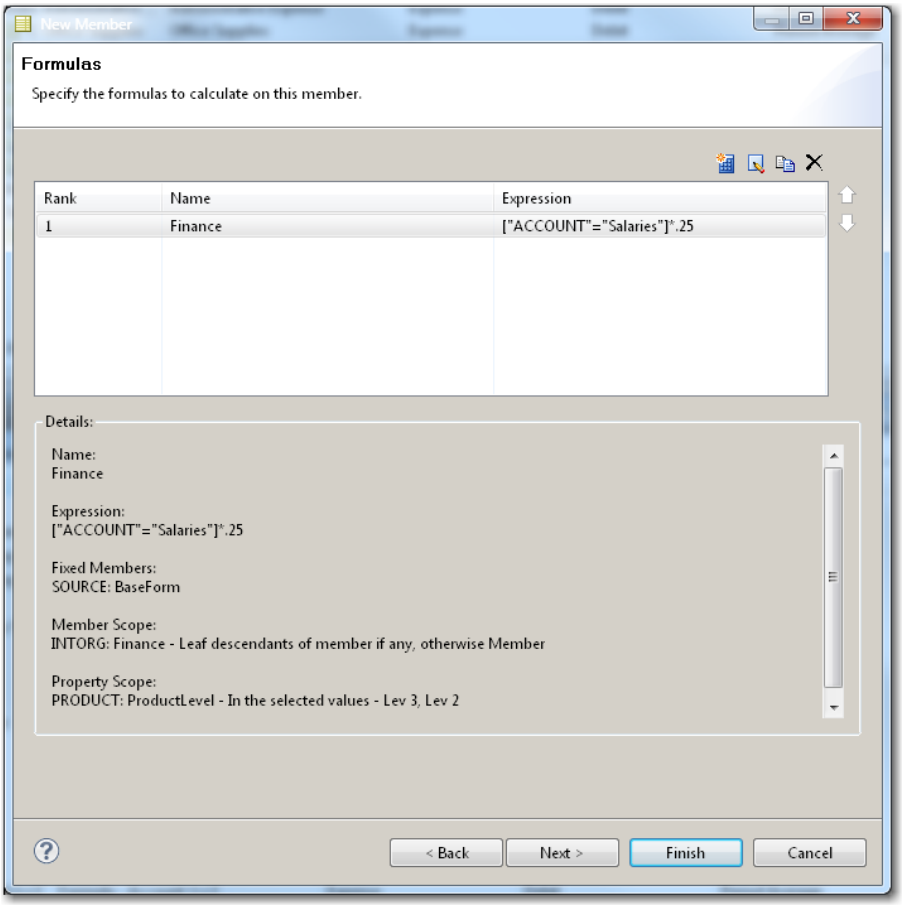
Property values are displayed in the context of the current view of the selected. You can select multiple property types and values in each dimension.



19. Click **Add** and **Close** to see the summary of member and property scoping on the **Scope** tab.



20. Click **Finish** when the formula scope specification is complete.



A summary window shows the expression text, fixed members, member, and property scope of the formula that you have created.

21. Click **Finish** if the information is correct.

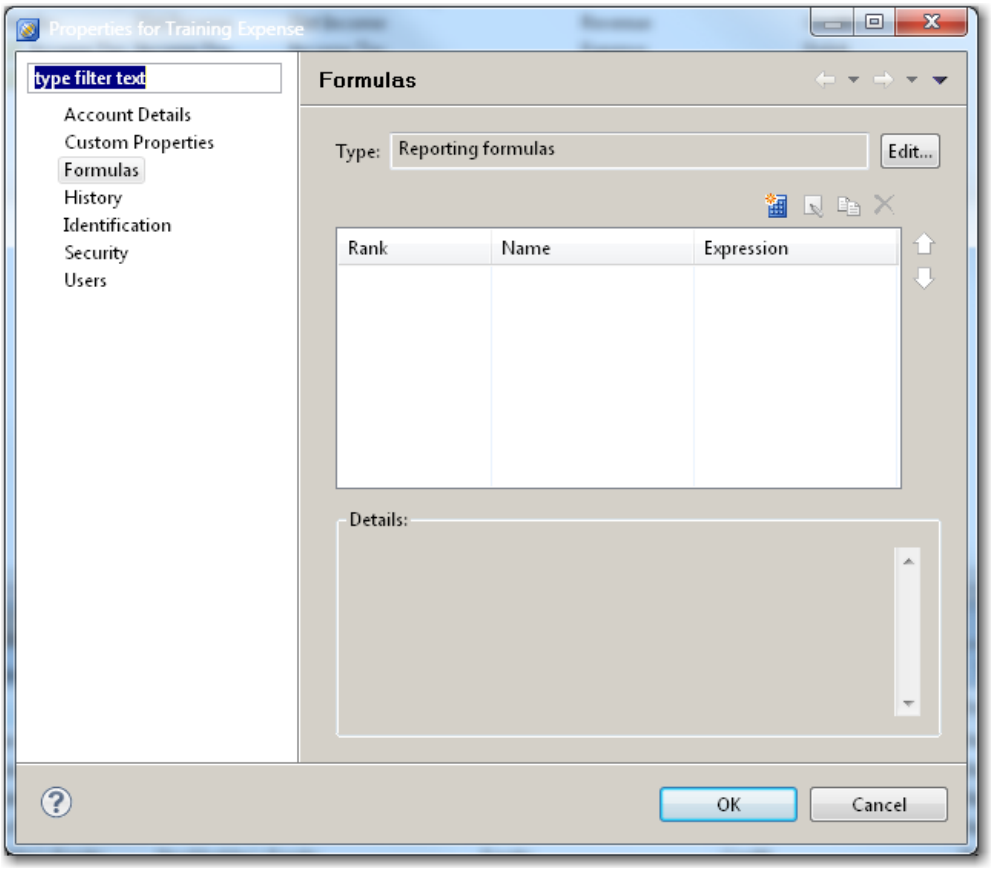
This completes the creation of a single formula on a member. At this point, you have not yet completed all the required steps for the addition of the calculated member. You are returned to the Formulas page. On the Formulas page, you can create additional formulas, edit existing formulas, or view the properties of existing formulas.

Repeat steps 3 (page 25) through 20 to create multiple formulas on the same member.

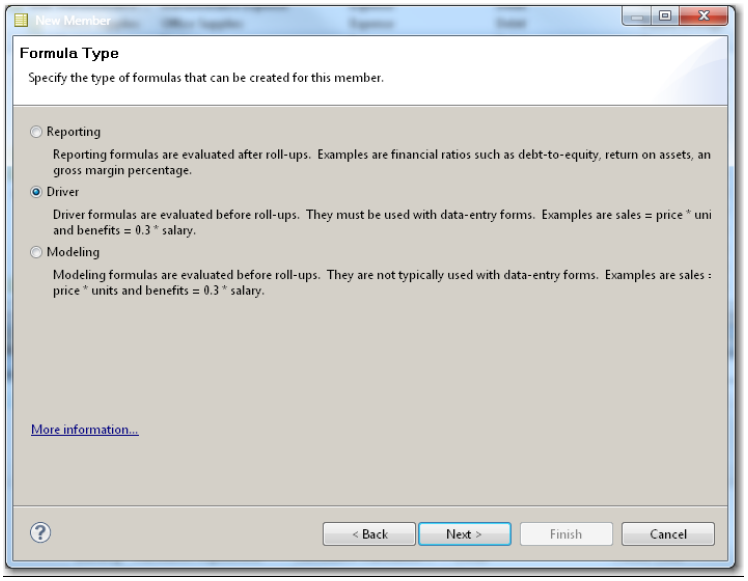
Adding Formulas to a Member

The following steps illustrate the addition of a formula to an existing member of the Account dimension. Note that only the Account dimension offers three formula types. In all other dimensions, only reporting formulas are available.

To add a formula to an existing member, highlight the member and either double-click the member or select the Show properties icon. On the Formulas page, the Reporting formula type is selected as the default. To select a different formula type, click **Edit**.

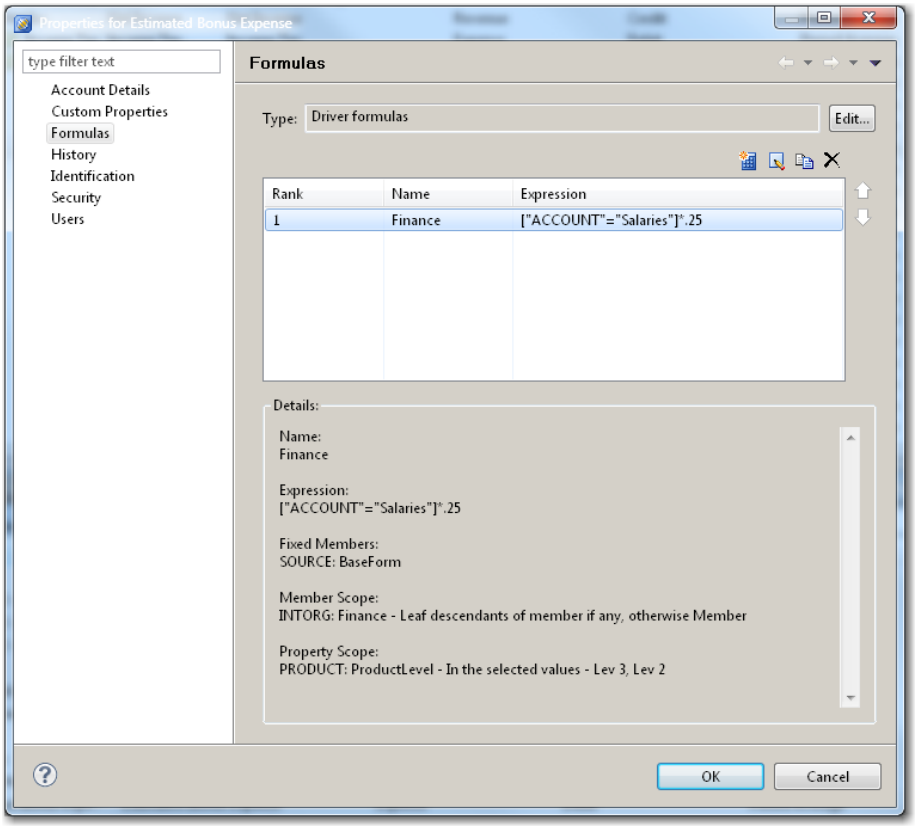


1. Select the desired formula type and click **Next**.



2. On the Formulas page, select the Create a new formula icon.

See steps 3 (page 25) through 20 in the preceding section for detailed information about creating formulas.

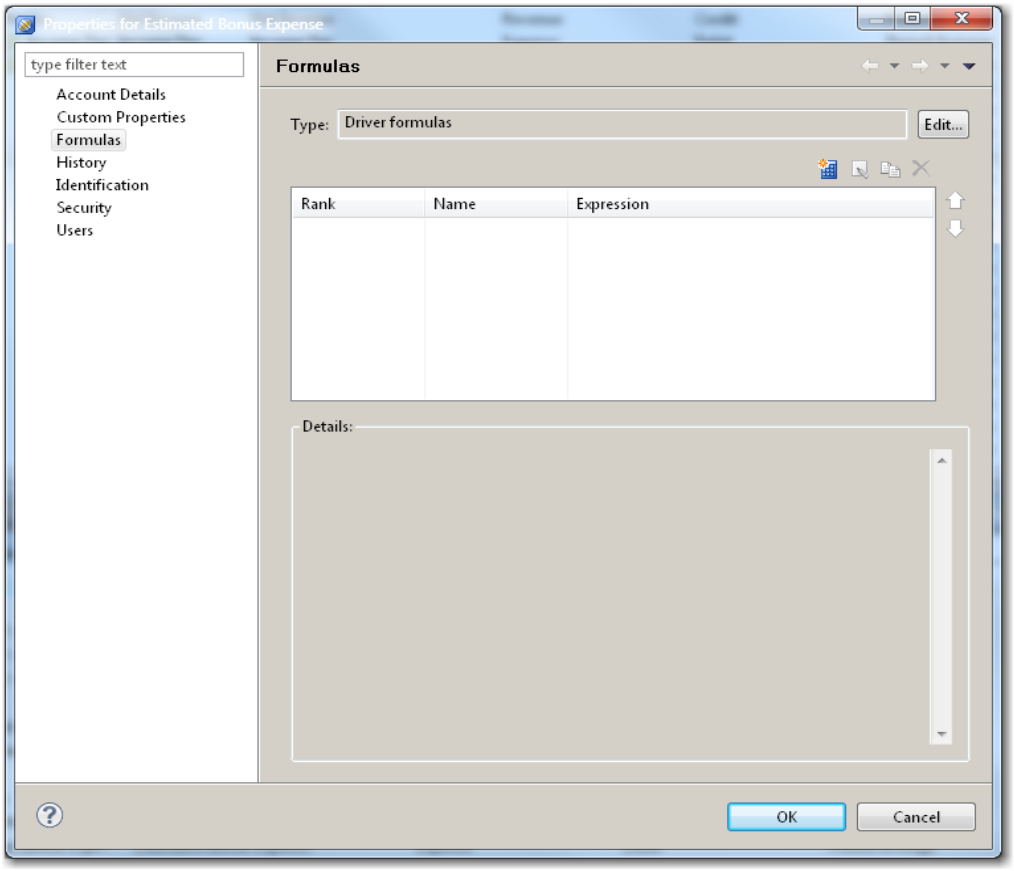


Removing Formulas from a Member

The following steps show how to remove a formula from a member of the Account dimension.

1. Highlight the member and either double-click or select the Show properties icon.
2. Click the Formulas page.

3. Highlight the formulas that you want to remove and click the Delete icon.



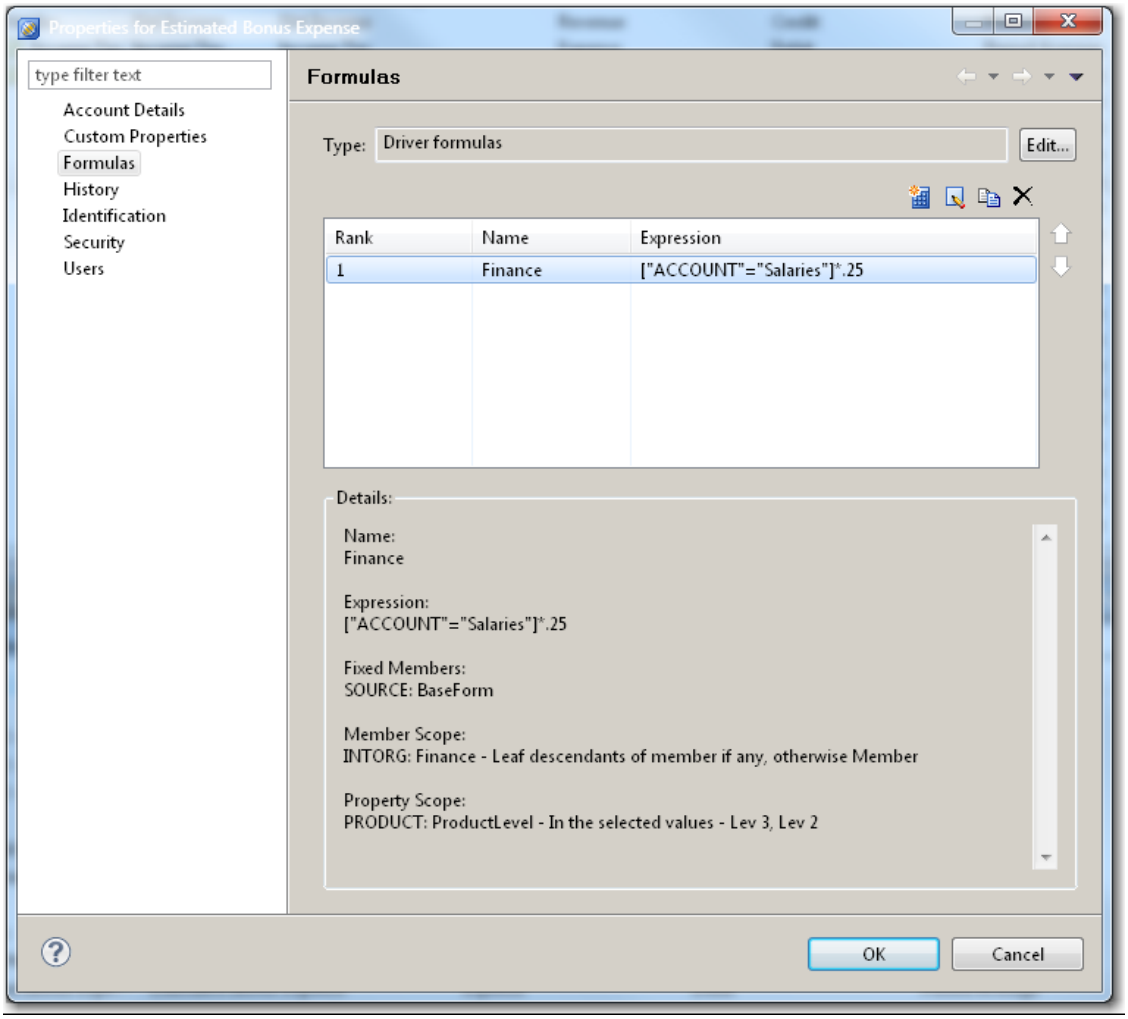
When you click the Delete icon, the formula expression(s) is removed. However, the deletion is not final until you click **OK**. If you click **Cancel**, then no changes are saved, and the formulas are not deleted. If you click **OK**, then changes are saved and the formula expressions are gone.

Note: If you delete a driver formula for which results have been computed and stored, the stored results are not deleted. To delete the stored results, select the Run driver formulas for this form set option for the appropriate form set.

Editing a Calculated Member

The following steps show how to edit formula properties for a calculated member of the Account dimension.

1. Highlight the member and either double-click or select the Show properties icon.
2. Click the Formulas page.



You can edit the following formula properties:

- Formula type (Account dimension only)
- Formula name
- Expression text
- Fixed members
- Scope

Detailed steps for editing these properties are in the first section of this chapter. As with all properties, changes to formula-related properties are subject to versioning. A hierarchy that has a time/date stamp that is earlier than the time when a certain change was made to the formula still uses the formula as it was before the change.

Account Types for Calculated Members

The behavior of calculated members in the Account dimension is influenced by the member's account type. Here are the available account types, grouped by category:

Balance Account Types
Asset
Liability
Equity
Statistical Balance

Flow Account Types
Revenue
Expense
Statistical Flow

Other Account Types
Statistical

The Retained Earnings and Cumulative Translation Adjustment account types cannot be associated with formulas and are therefore excluded from the preceding list.

Formula results are calculated by a distinct method for each account type category. The methods are:

- Balance account types aggregate and then convert results.
- Flow account types convert and then aggregate results.

Statistical accounts do not participate in aggregation or conversion.

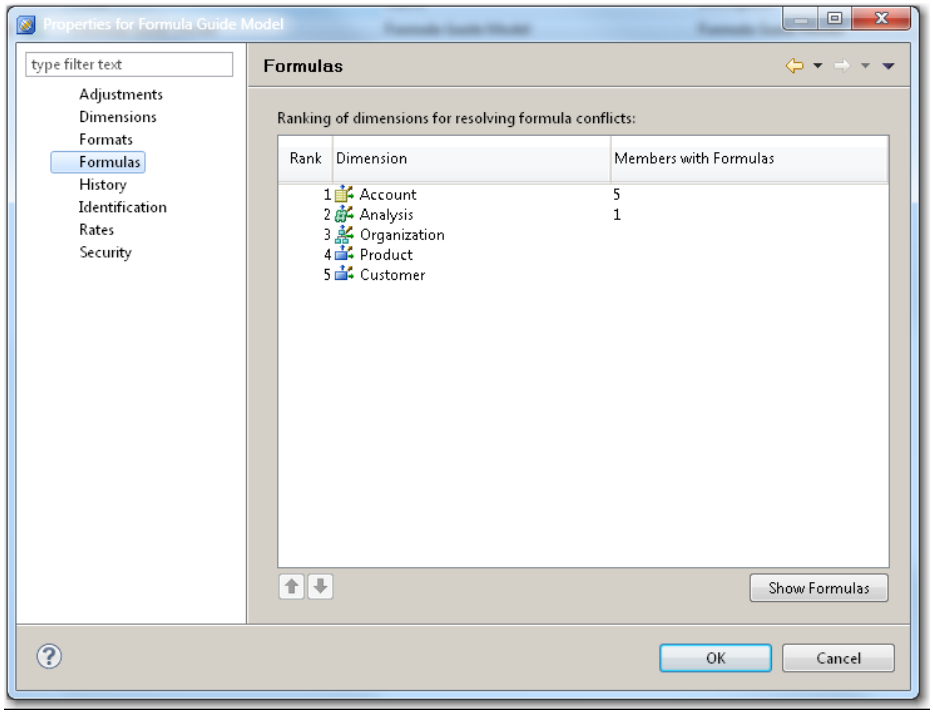
The calculation and currency conversion methods depend on the account type of the calculated member.

Resolving Conflicts between Dimensions

Introduction

Formula conflicts are limited to reporting formulas. Due to execution order, driver formulas and modeling formulas always run before reporting formulas and are available only in the Account dimension. Since only the Account dimension allows driver formulas and modeling formulas, there are never any conflicts with these formula types.

However, a model with reporting formulas in more than one dimension can have crossings that have two or more formulas associated with them. In such a case, the formula rank determines which formula runs at the crossing. The rank is managed through Model Properties on the Formulas page, as shown here:



To illustrate formula rank, consider the following two reporting formula expressions:

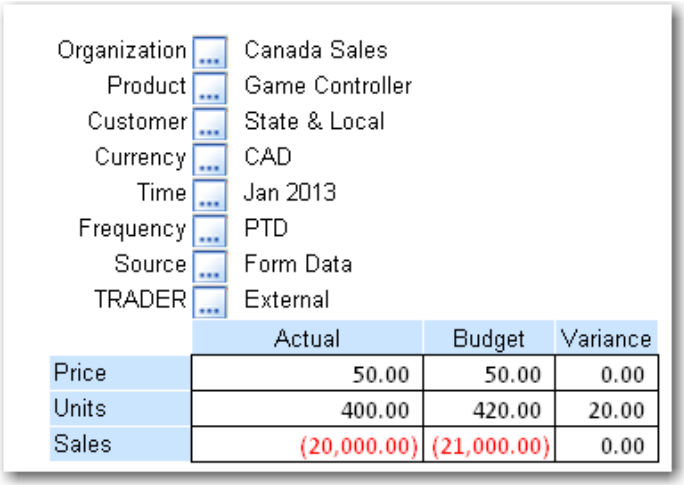
Account dimension: Sales = ["ACCOUNT"="Price"]*["ACCOUNT"="Units"]*-1
Analysis dimension: Variance = ["ANALYSIS"="Budget"]-["ANALYSIS"="Actual"]

We follow a scenario for each expression:

- "Scenario 1: Account Dimension Has Higher Rank than Analysis Dimension"
- "Scenario 2: Analysis Dimension Has Higher Rank than Account Dimension"

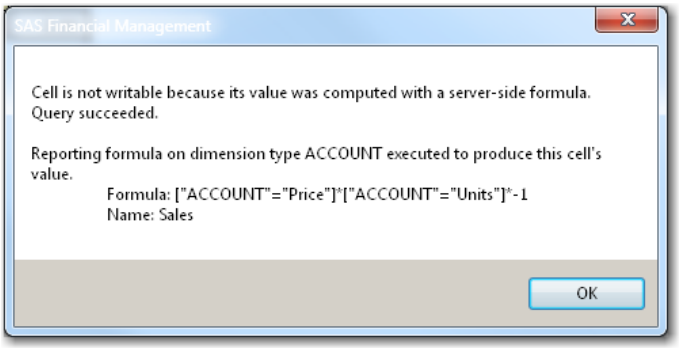
Scenario 1. Account Dimension Has Higher Rank than Analysis Dimension

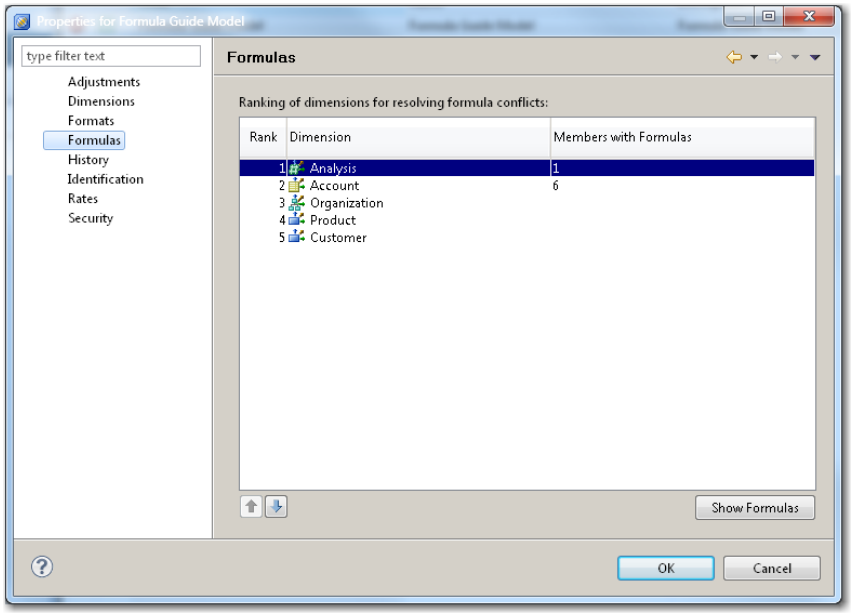
The Account dimension is ranked higher than the Analysis dimension, as shown in the preceding window. This means that the Account formula runs at the crossing where the formulas conflict, as shown here:



With the Account dimension ranked first, Variance for Sales is computed as 0 * 20 * -1 = 0.

1. At the crossing for the Sales Account and Variance Analysis members, select **Tools ▶ Cell Information** to view the following message:





Scenario 2: Analysis Dimension Has Higher Rank than Account Dimension

1. Update the formula rank such that the Analysis dimension is ranked first on the Formulas page, as shown here:

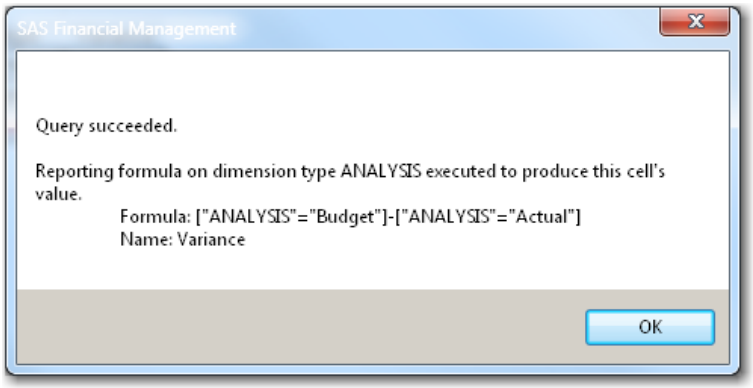
   The resulting read-only table appears as follows:

Organization	Canada Sales
Product	Game Controller
Customer	State & Local
Currency	CAD
Time	Jan 2013
Frequency	PTD
Source	Form Data
TRADER	External

	Actual	Budget	Variance
Sales	(20,000.00)	(21,000.00)	(1,000.00)
Price	50.00	50.00	0.00
Units	400.00	420.00	20.00

With the Analysis dimension ranked first, Variance for Sales is computed as (21,000.00) - (20,000.00) = (1,000.00).

2. At the crossing for the Sales Account and Variance Analysis members, select **Tools ▶ Cell Information** to view the following message:



Note that the cell information reflects the formula expression based on rank. The cell information includes the formula expression for the Variance Analysis member, because it is ranked first.

Virtual Child Members

In SAS Financial Management, a virtual child is automatically available to a member that is designated as a roll-up. A virtual child enables you to enter values at a roll-up point where less detail is required and/or spreading or allocations are involved. Virtual children are available for these dimension types:

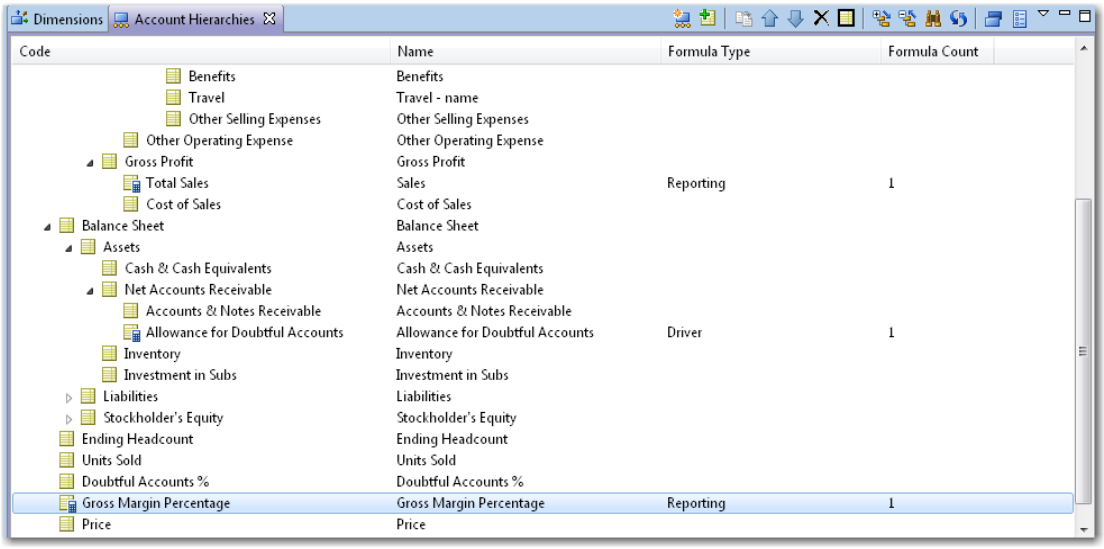
- IntOrg
- Account
- Custom

A virtual child is similar to an ordinary leaf member in that a formula can run on a virtual child. In contrast, a virtual child differs from an ordinary leaf member in that you cannot assign a formula to a virtual child. A formula on a roll-up member is always ignored, and the result is the sum of leaf values.

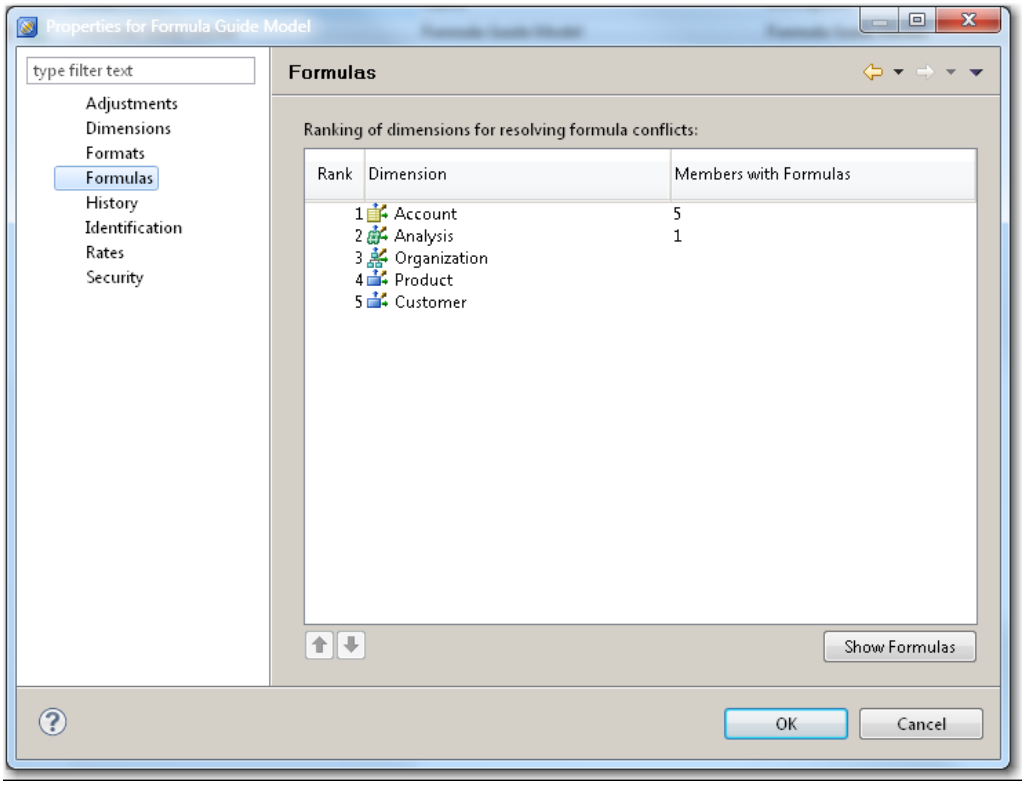
Viewing Formula Information in SAS Financial Management Studio

SAS Financial Management Studio provides formula information in the Dimensions and Models workspaces. In the Dimensions workspace, formula information is available on the **Members** and **Hierarchies** tabs for all dimension types that support calculated members. The Account dimension type offers optional informational headers and columns for both Formula Type and Formula Count. For Intorg, Analysis, and custom dimension types, only Formula Count is offered. The Formula Count header provides the number of formulas on a calculated member. For a detailed discussion of defining multiple formulas on the same member, see Chapter 6, “Advanced Formula Concepts.”

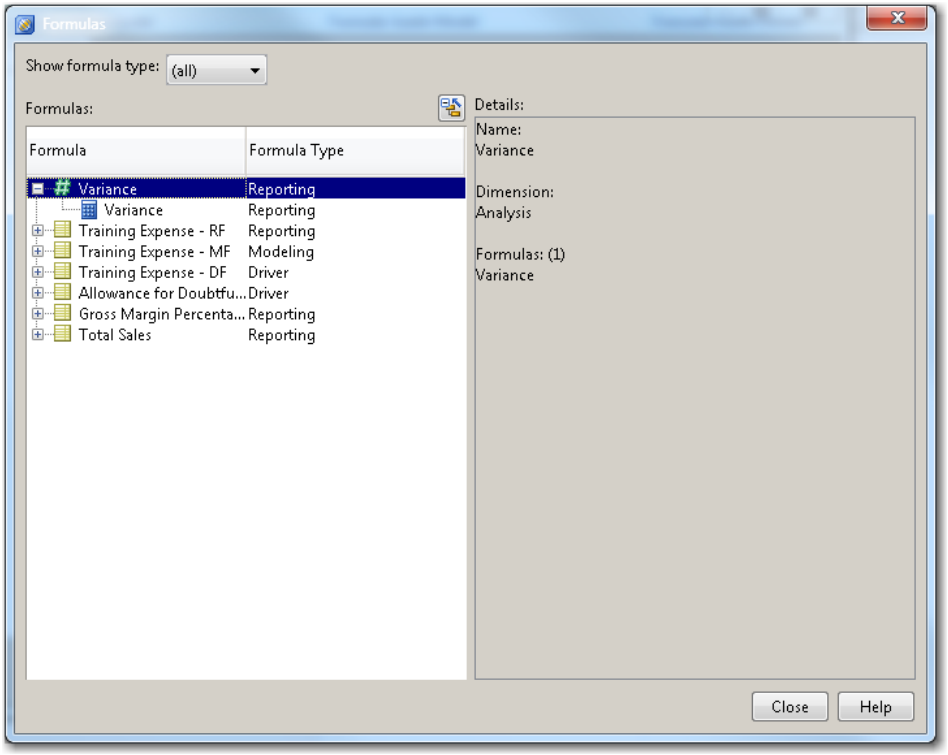
The following window illustrates the optional headers displayed for the Account dimension type:



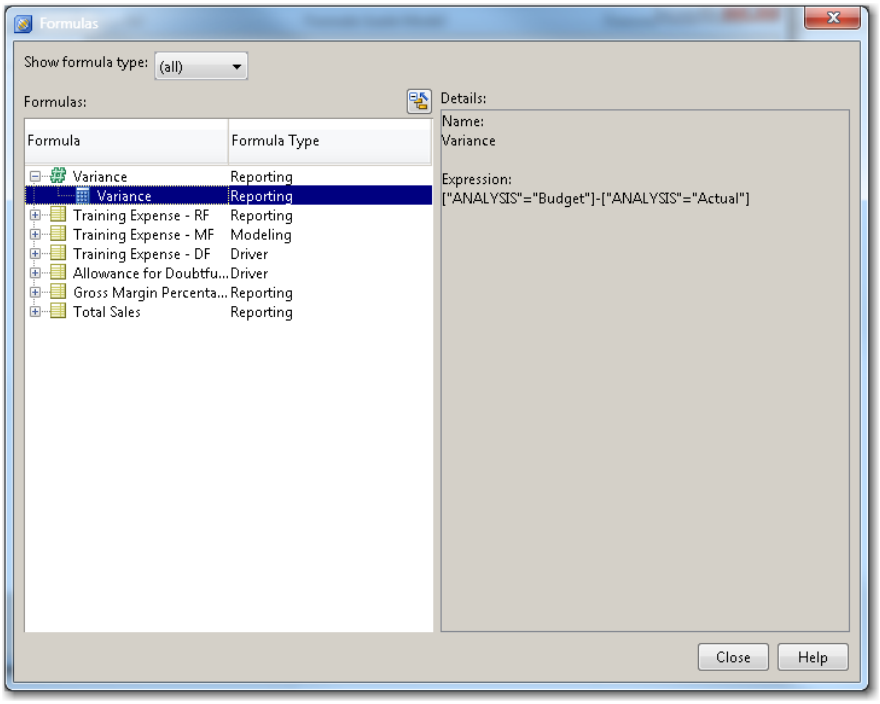
In the Models workspace, Model Properties displays the number of members with formulas on the Formulas page for each dimension in the model, based on the selected hierarchy and as-of-date.



For more information about the formulas, click the Show Formulas box. This provides information about all formulas in the model based on the selected hierarchies and as-of-dates.



Formula detail can be filtered based on the formula type and is displayed by dimension member code. Each formula displays the name of the calculated member, the Dimension that the calculated member is in, and the number of formula expressions on the member. Expanding the calculated member displays the various formula expressions on the member. Each formula expression displays the name as well the formula syntax. Any scoping and/or fixed member information is also included in the Details window.

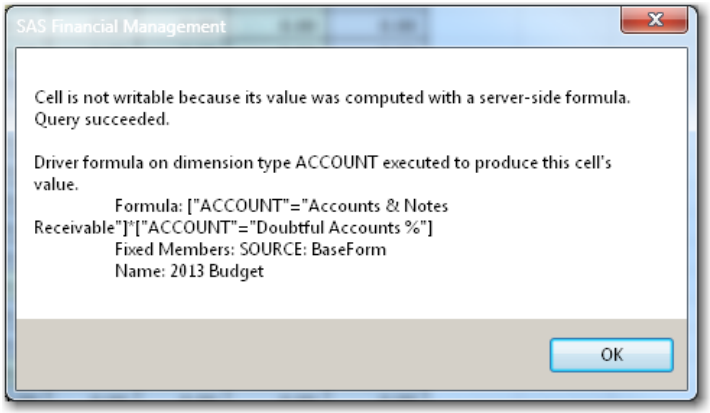


Viewing Formula Information in Excel

Formula information is available on a crossing-by-crossing basis in the Excel Add-In. For either a data-entry table or a read-only table, you can select **Tools ▶ Cell Information** from the **SAS Financial Management** menu to view formula information. The following formula information is displayed:

- Formula type
- Dimension of calculated member
- Formula expression
- Name of formula
- Fixed members, if any

To illustrate, the following message is displayed for the driver formula Allowance for Doubtful Accounts:



Here are some points to keep in mind when querying for formula details:

- In the event of a formula conflict involving two or more formulas on the same crossing, the formula expression that is ranked higher is displayed.
- For modeling formulas and driver formulas, *all* members of the selected crossing must be leaf members. Therefore, all dimensions, including Trader and Source, must have a leaf member selected. This means that you must select a leaf member for every dimension that is displayed in the table. In addition, the table default member must be a leaf member for any dimension that is in the model but not displayed in the table.

CHAPTER 5

Formula Basics for Excel-Based Calculated Members

Introduction 54
Creating an Excel-Based Calculated Member 54
Removing an Excel-Based Calculated Member 59
Editing an Excel-Based Calculated Member 60
Viewing Formula Information for Excel-Based Calculated Members 63
Pivoting an Excel-Based Calculated Member 63
Resolving Conflicts between Excel-Based Calculated Members 65
- *Introduction* 65
- *Excel-Based Calculated Member Expressions* 65
- *Results* 65

Examples of Excel-Based Calculated Member Formulas 67
- *Introduction* 67
- *fmValue Function* 67
  - *Use* 67
  - *Syntax* 67
  - *Example* 67
- *fmCode Function* 67
  - *Use* 67
  - *Syntax* 67
  - *Example* 68
- *fmProperty Function* 68
  - *Use* 68
  - *Syntax* 69
  - *Example* 69
  - *Example* 69
- *fmRate Function* 70
  - *Use* 70
  - *Driver Rate Table in SAS Financial Management Studio* 70
  - *Syntax* 70
  - *Example* 71
- *fmXRate Function* 71
  - *Use* 71
  - *Syntax* 71
  - *Example* 71
  - *Example of Resulting CDA Formula* 72
- *fmCXRate Function* 72
  - *Use* 72
  - *Syntax* 72
  - *Example* 73
  - *Example* 74
  - *Examples* 75
  - *Reference to Another Excel-Based Calculated Member in the Same Dimension* 75
- *Absolute References to Cells in the Same Workbook* 78
- *Using Any Excel Function or Valid Excel Expression* 78
  - *Introduction* 78
  - *Example* 79
- *Excel-Based Calculated Members and Display Styles* 80
  - *Introduction* 80
  - *Scenario 1: Excel Styles Set to Display Default for Debit and Credit Balances* 80
  - *Scenario 2: Excel Styles Set to Display Credit Balances as Positive* 80

Scenario 3: Excel Styles Set to Display Credit Balances as Positive and Referenced Members Not Displayed on the Table *81*
Formulas That Run Out of Bounds *82*
How Calculated Members are Displayed in Excel? *84*
Scenario 1. Time Offsets *84*
Scenario 2. Referencing a Member or Hidden Member Not on the Table *84*
Scenario 3. Placing an Excel-Based Calculated Member Before or After a Member or Hidden Member Not on the Table *84*
Excel-Based Calculated Members and Supplemental Tables *84*

Introduction

Using the SAS Financial Management for Microsoft Excel Add-In, you can create Excel-based calculated members on a table-by-table basis. Excel-based calculated members look the same as members that are created in SAS Financial Management Studio. They are most similar in behavior to reporting formulas, sharing similar calculation and currency conversion methods. They differ primarily in reuse because formulas are created and saved locally, not on the server.

Excel-based calculated members support any combination of the following in a formula expression:

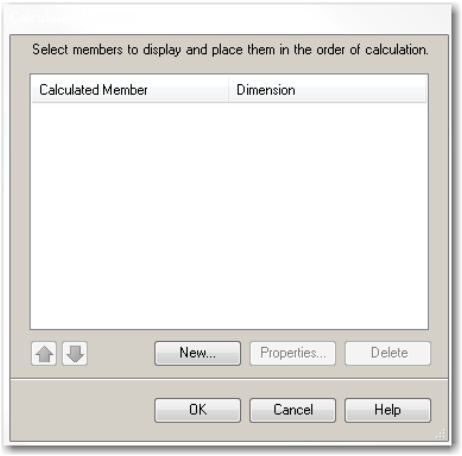
- reference to any member in the same dimension
- absolute reference to any crossing in a table in the same workbook
- absolute reference to any cell in the same workbook
- any Excel function or valid Excel expression

Creating an Excel-Based Calculated Member

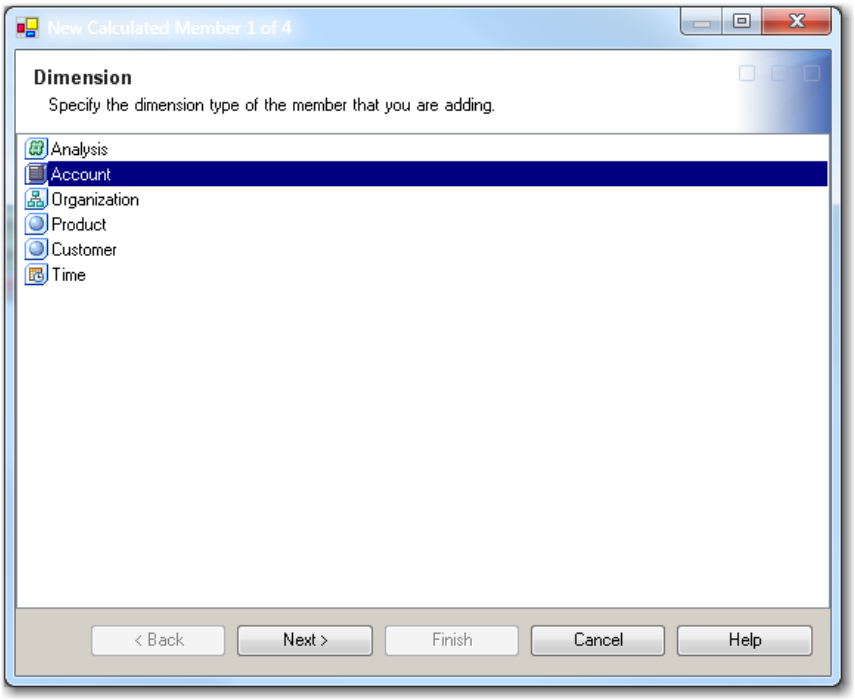
The following steps show how to create an Excel-based calculated member in the Account dimension.

1. Place the cursor in any cell of an active table and select **Members ▶ Calculated Members** from the **SAS Financial Management** menu.

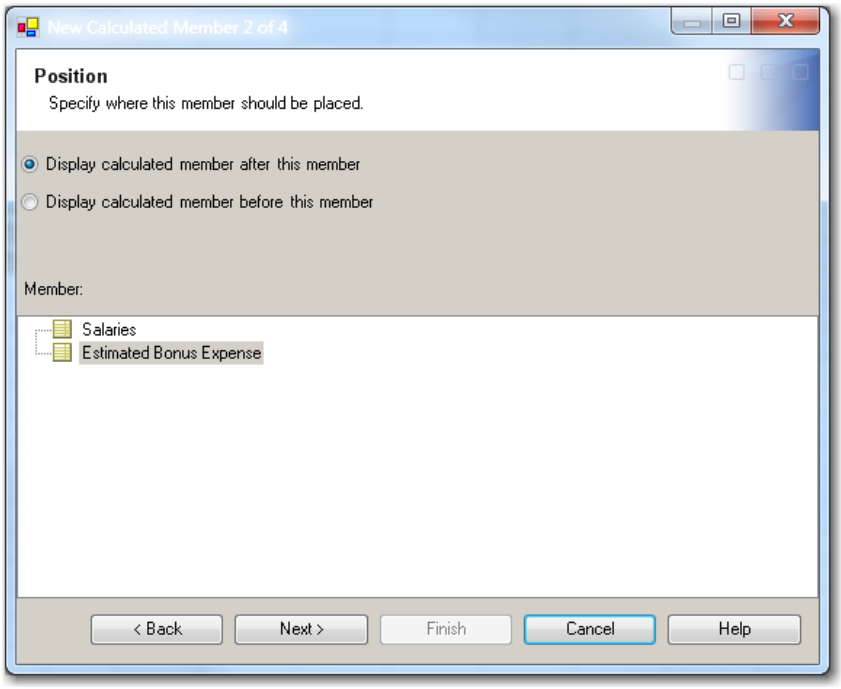
The following window appears:



Select **New**, highlight the desired dimension, and click **Next**.

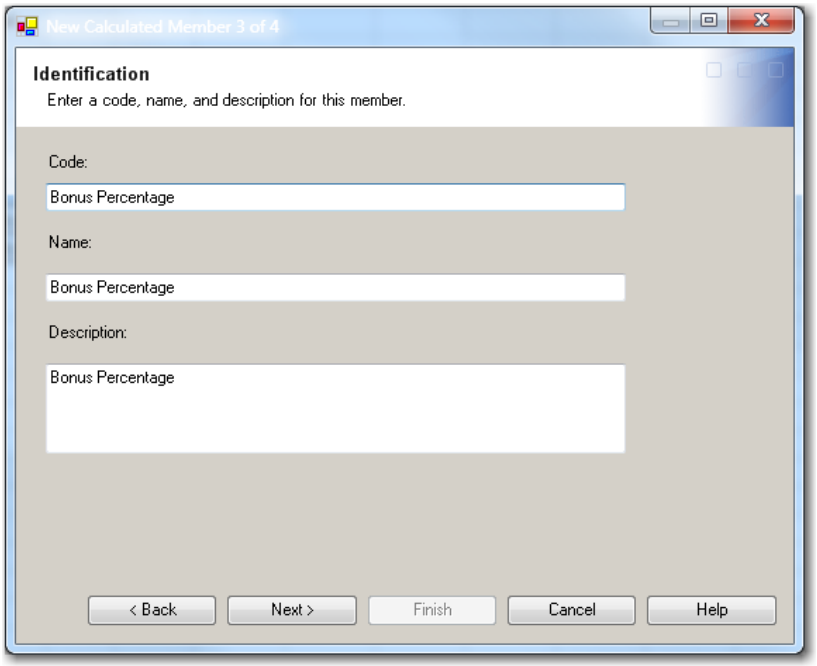


The Position page displays the hierarchical view of the current members selected for display in the table. To determine the placement of the calculated member, highlight an available member and choose whether to display the calculated member before or after the selected member.



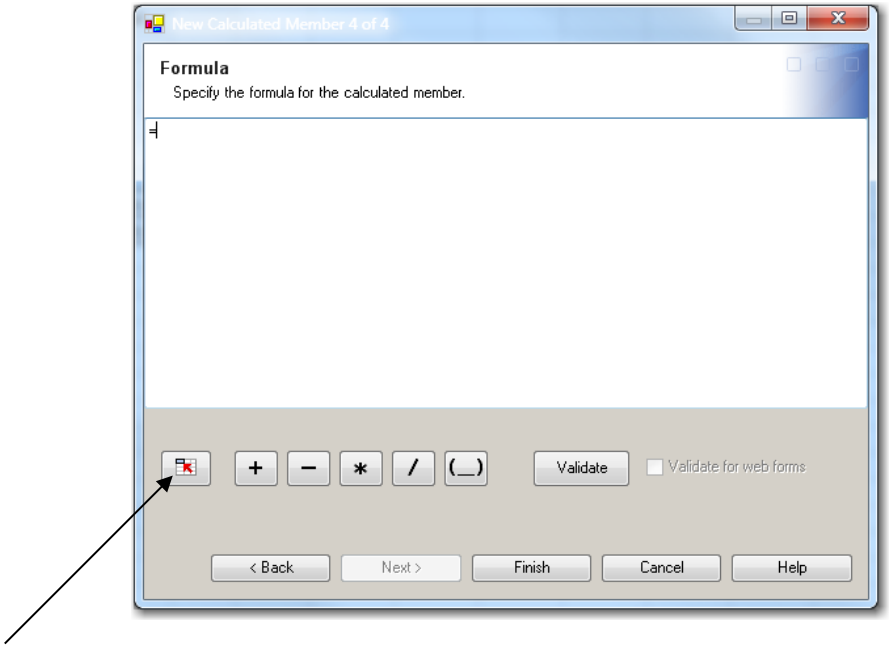
2 Click **Next** to advance.

3 Enter the code, name, and description of the calculated member that you are creating. Click **Next**.

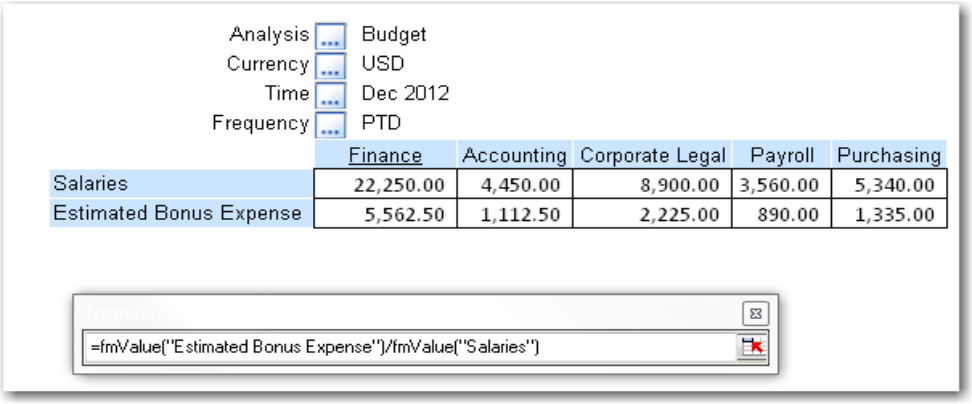


4 Type the formula expression directly in the Formula page or use the icons.

Using the Formula Cell Selector (the icon in the lower left) is an easy way to select dimension members and cell references.



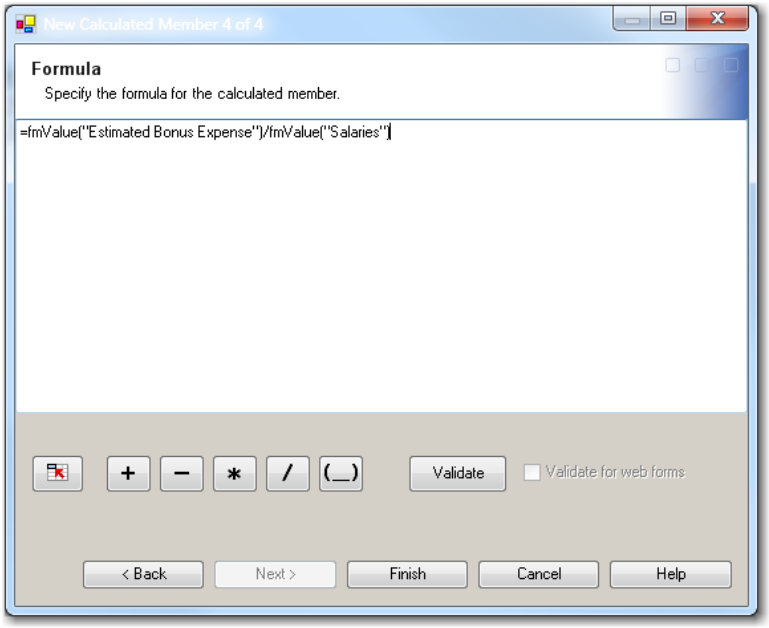
Similar to formula definition in Excel, the Formula Cell Selector is updated with the cell references and dimension members based on cursor placement and selection.



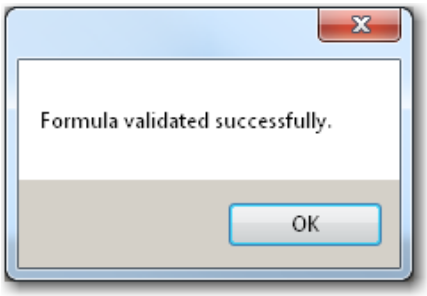
	Finance	Accounting	Corporate Legal	Payroll	Purchasing
Salaries	22,250.00	4,450.00	8,900.00	3,560.00	5,340.00
Estimated Bonus Expense	5,562.50	1,112.50	2,225.00	890.00	1,335.00

5 Click the Formula Cell Selector icon.

You are returned to the Formula window.

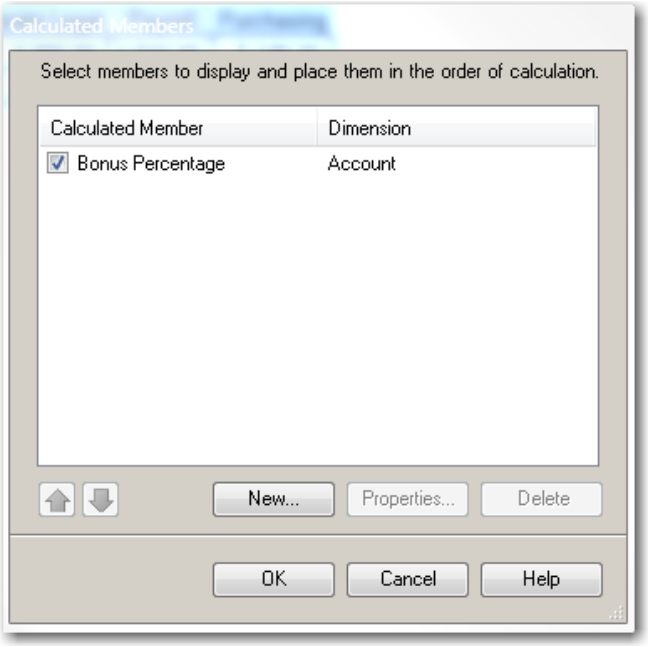


6 After you complete the formula expression, select **Validate** to ensure its accuracy.



7 Click **OK** and **Finish**.

In the following window, you can create a new Excel-based calculated member, view properties of existing members, and delete or disable existing members.



8 Click **OK** to finish creating a new member.

The results are displayed in the following table:

Analysis Budget
Currency USD
Time Dec 2012
Frequency PTD

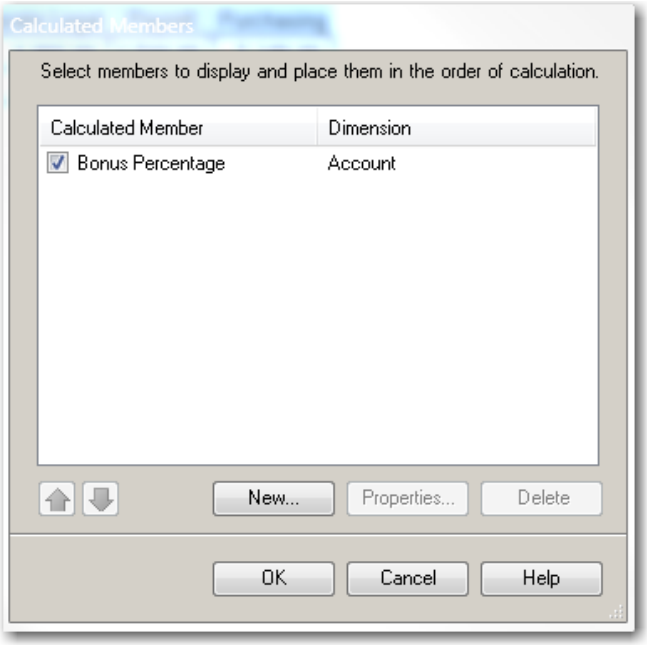
	Finance	Accounting	Corporate Legal	Payroll	Purchasing
Salaries	22,250.00	4,450.00	8,900.00	3,560.00	5,340.00
Estimated Bonus Expense	5,562.50	1,112.50	2,225.00	890.00	1,335.00
Bonus Percentage	0.25	0.25	0.25	0.25	0.25

Removing an Excel-Based Calculated Member

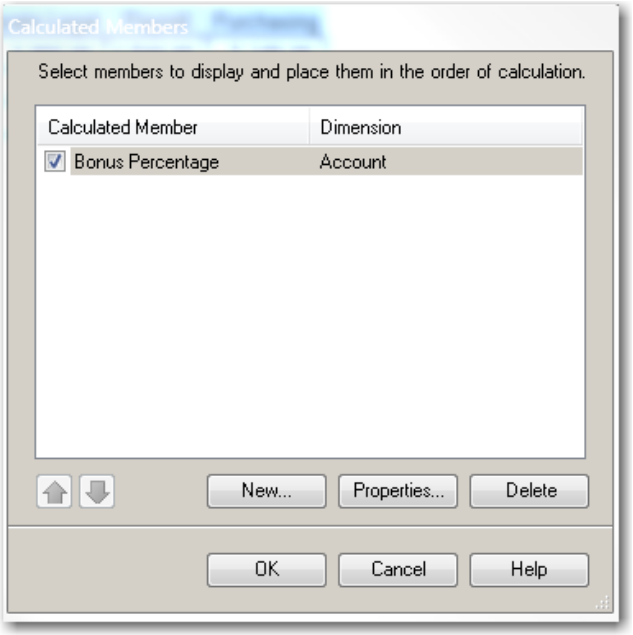
You can remove Excel-based calculated members either by disabling them or deleting them. The following steps illustrate how to remove an existing Excel-based calculated member:

1. With the cursor in any cell of an active table, select **Members ▶ Calculated Members** from the **SAS Financial Management** menu.

   The following window appears:

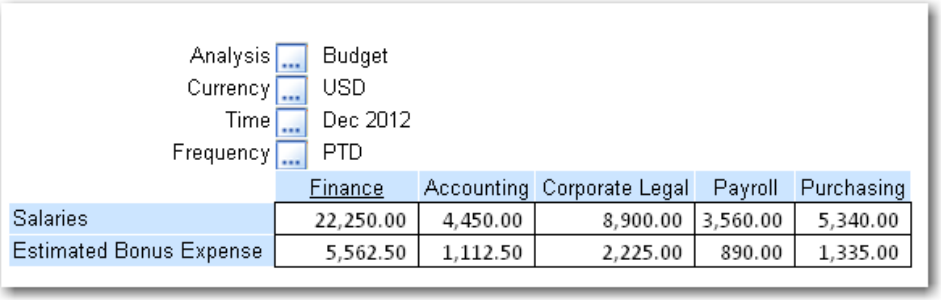


   Highlight the member that you want to remove.



   Either deselect the highlighted calculated member or click **`Delete`**.

When you click **OK**, either action (deselect or click the button) removes the member from the display. Deselecting a calculated member saves it for use at a later time.



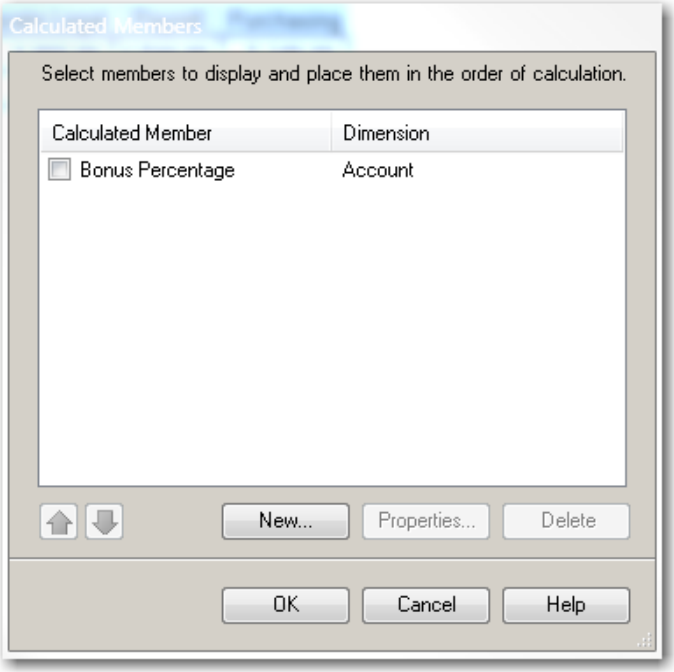
	Finance	Accounting	Corporate Legal	Payroll	Purchasing
Salaries	22,250.00	4,450.00	8,900.00	3,560.00	5,340.00
Estimated Bonus Expense	5,562.50	1,112.50	2,225.00	890.00	1,335.00

Editing an Excel-Based Calculated Member

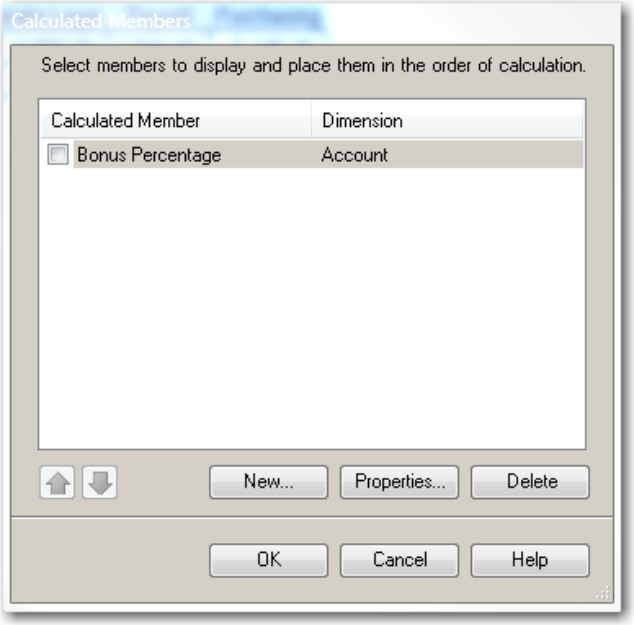
The following steps illustrate how to edit an existing Excel-based Calculated Member:

1. With the cursor in any cell of an active table, select **Members ▶ Calculated Members** from the **SAS Financial Management** menu.

   The following window appears:

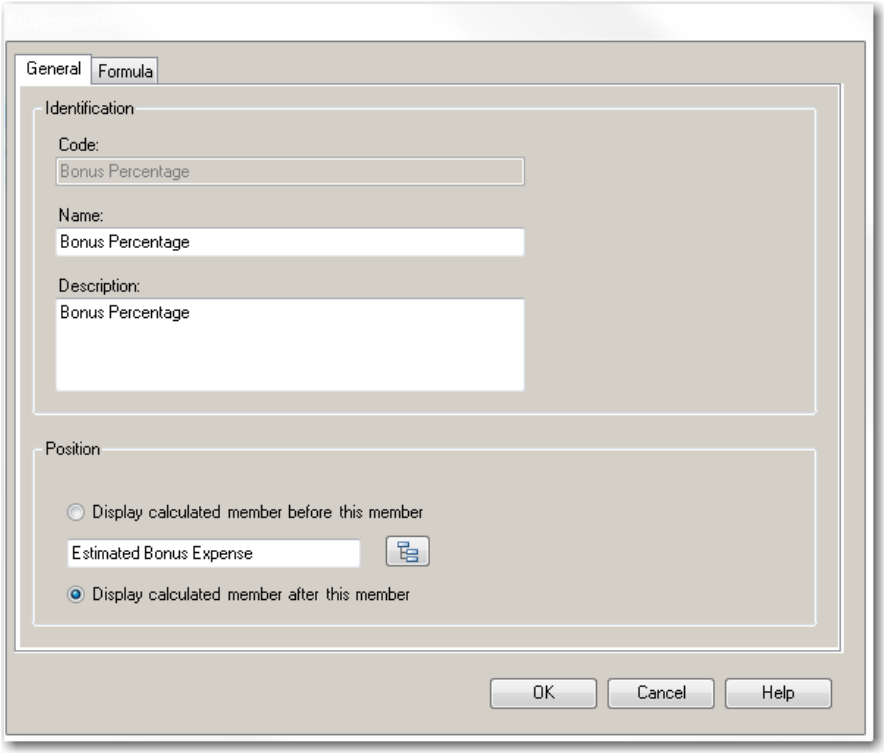


Highlight the member that you want to edit.

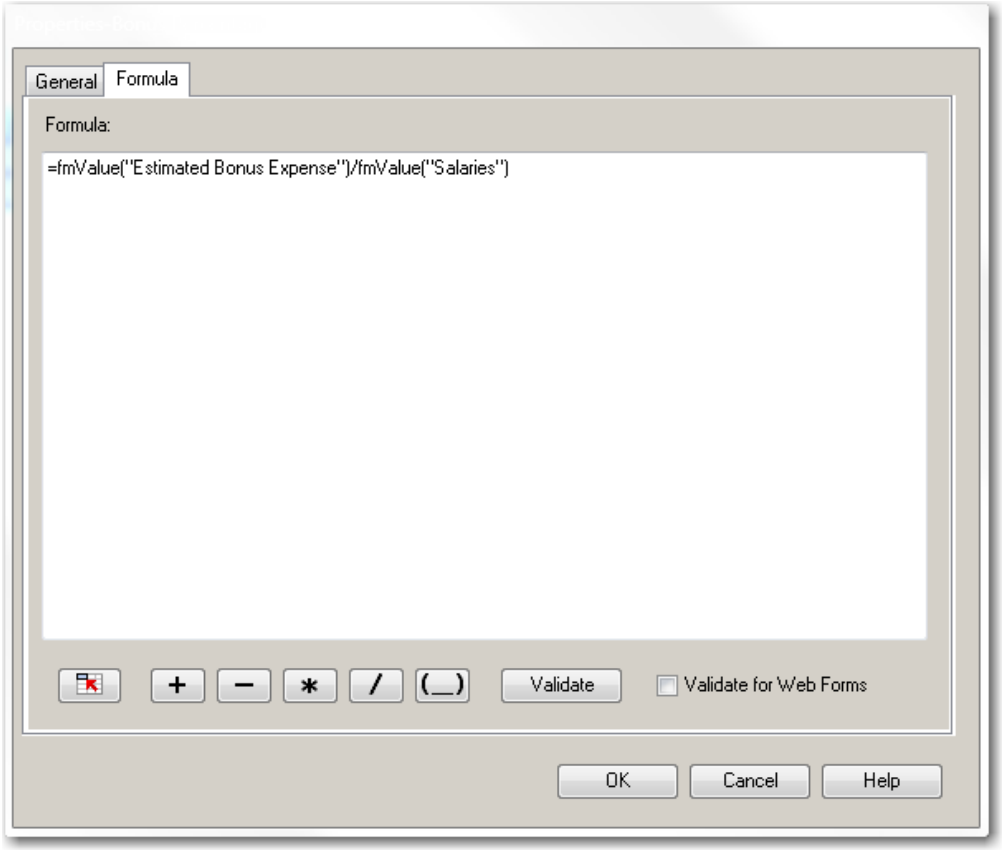


Click **Properties** to open the calculated member's Properties window.

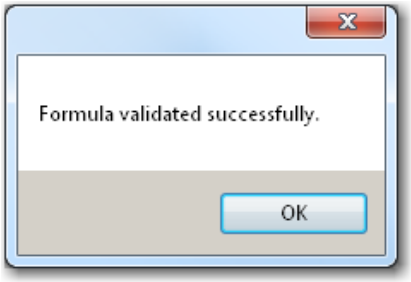
The default tab displayed is the **General** tab. This is where you modify the name, description, and position.



On the **Formula** tab, modify the formula expression and then click **Validate**.

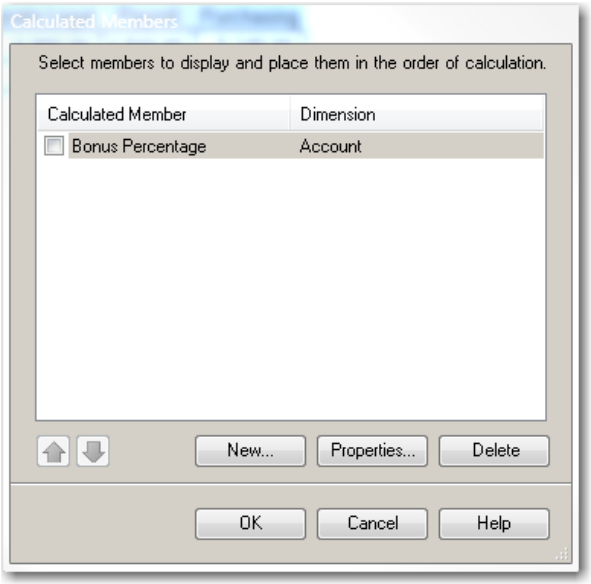


If the formula is valid, click **OK.**



Click **OK** and **Finish**.

Click **OK** to finish editing the calculated member.



Viewing Formula Information for Excel-Based Calculated Members

To view formula information for Excel-based calculated members, select **Members ▶ Calculated Members** from the **SAS Financial Management** menu and follow steps similar to the steps for editing an Excel-based calculated member.

Pivoting an Excel-Based Calculated Member

Not only do Excel-based calculated members look like server-side members, they can also be pivoted and function as table slicers. As illustrated in the first section of this chapter, the calculated member Bonus Percentage was originally displayed in the table rows, as shown here:

Analysis Budget
Currency USD
Time Dec 2012
Frequency PTD

	Finance	Accounting	Corporate Legal	Payroll	Purchasing
Salaries	22,250.00	4,450.00	8,900.00	3,560.00	5,340.00
Estimated Bonus Expense	5,562.50	1,112.50	2,225.00	890.00	1,335.00
Bonus Percentage	0.25	0.25	0.25	0.25	0.25

You can modify the table using **Members ▶ Pivot** on the **SAS Financial Management** menu. The following updated table illustrates the calculated member on the columns:

Analysis Budget
Currency USD
Time Dec 2012
Frequency PTD

	Salaries	Estimated Bonus Expense	Bonus Percentage
Finance	22,250.00	5,562.50	0.25
Accounting	4,450.00	1,112.50	0.25
Corporate Legal	8,900.00	2,225.00	0.25
Payroll	3,560.00	890.00	0.25
Purchasing	5,340.00	1,335.00	0.25

You can modify the table to display the Excel-based calculated member as a slicer:

Analysis Budget
Currency USD
Frequency PTD
Account Bonus Percentage

	Dec 2012
Finance	0.25
Accounting	0.25
Corporate Legal	0.25
Payroll	0.25
Purchasing	0.25

You can nest Excel-based calculated members. In the following table, the Bonus Percentage calculated member is nested inside the Time Dimension on the columns:

Analysis Budget
Currency USD
Frequency PTD

	Dec 2012		
	Salaries	Estimated Bonus Expense	Bonus Percentage
Finance	22,250.00	5,562.50	0.25
Accounting	4,450.00	1,112.50	0.25
Corporate Legal	8,900.00	2,225.00	0.25
Payroll	3,560.00	890.00	0.25
Purchasing	5,340.00	1,335.00	0.25

Resolving Conflicts between Excel-Based Calculated Members

Introduction

Excel-based calculated members can conflict only with other Excel-based calculated members. All server-side formula types (driver, modeling, and reporting) run before Excel-based calculated members. Therefore, in the event of a conflict involving two or more Excel-based calculated members, the members can be ordered in a manner similar to server-side calculated members. Consider the following scenario; there are two Excel-based calculated members:

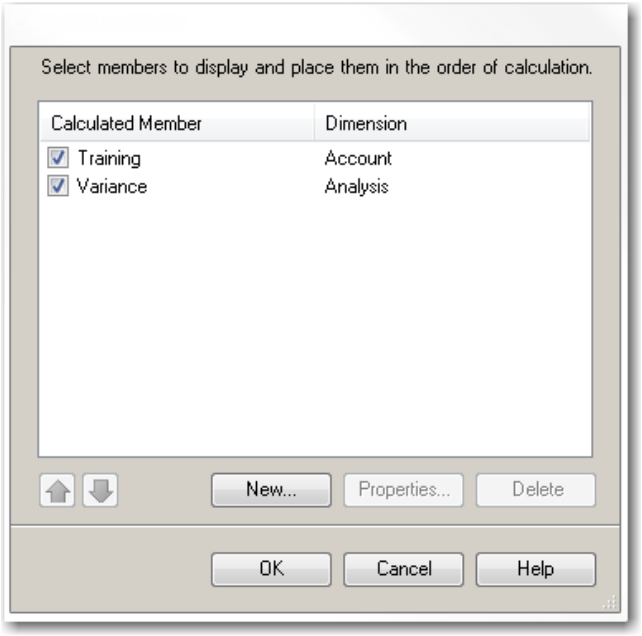
Excel-Based Calculated Member Expressions

Account dimension: Training = 1,000.00

Analysis dimension: Variance = fmValue("ACTUAL")-fmValue("BUDGET")

Results

If the Training-calculated member is ranked higher than the Variance-calculated member as shown here, then the Training formula expression is executed and the Variance formula expression is ignored.

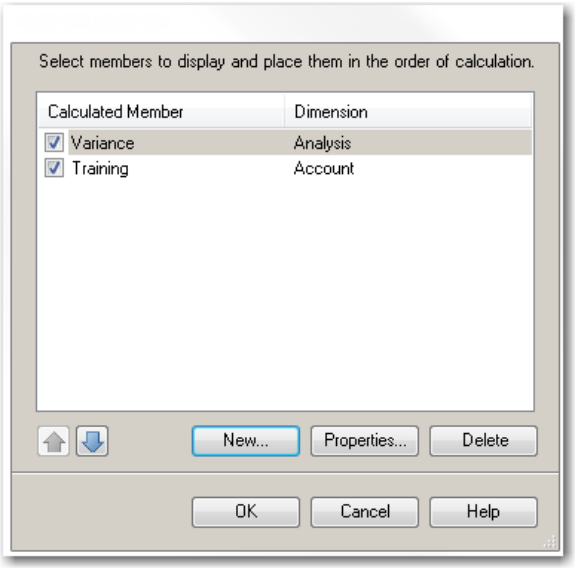


With this ranking, the results are as follows:

Organization Finance
Currency USD
Time Jan 2013
Frequency PTD

	Actual	Budget	Variance
Ending Headcount	4.00	5.00	(1.00)
Training	1,000.00	1,000.00	1,000.00

If the Variance-calculated member is ranked higher than the Training-calculated member, then the Variance formula expression is executed and the Training formula expression is ignored.



With this ranking, the results are as follows:

Organization Finance
Currency USD
Time Jan 2013
Frequency PTD

	Actual	Budget	Variance
Ending Headcount	4.00	5.00	(1.00)
Training	1,000.00	1,000.00	0.00

Examples of Excel-Based Calculated Member Formulas

Introduction

The following sections provide examples for the various functions supported in Excel-based calculated member formulas.

fmValue Function

Use

The fmValue function is used to retrieve crossing values based on the members referenced in the formula. This function replaces the Member function in SAS Financial Management 4.4. Typically, this calculated member is placed on the same dimension as the member(s) that it references in its formula.

Syntax

fmValue(“<member code name>”)

Example

Variance =

fmValue(“ACTUAL”)-fmValue(“BUDGET”)

Product ... Product Total
Account ... Gross Profit
Currency ... USD
Frequency ... PA
Time ... Jan 2012

	Actual	Budget	Variance
Worldwide Operations	1,205,709.00	1,115,000.00	90,709.00
U.S. Operations	420,397.00	415,000.00	5,397.00
European Operations	673,240.00	590,000.00	83,240.00
Canada Operations	79,389.00	75,000.00	4,389.00
Mexico Operations	32,683.00	35,000.00	(2,317.00)

fmCode Function

Use

The fmCode function is used to retrieve the dimension member code based on the dimension referenced in the formula. When used independently of other calculated member functions, this calculated member should be placed in a dimension other than the dimension it is referencing in its formula. The following example illustrates this.

Syntax

fmCode(“<dimension code name>”)

Example

Org Code =

fmCode("ORGANIZATION")

Product	Product Total
Account	Gross Profit
Currency	USD
Frequency	PA
Time	Jan 2012

	Org Code	Actual	Budget	Variance
Worldwide Operations	Worldwide Operations	1,205,709.00	1,115,000.00	90,709.00
U.S. Operations	U.S. Operations	420,397.00	415,000.00	5,397.00
European Operations	European Operations	673,240.00	590,000.00	83,240.00
Canada Operations	Canada Operations	79,389.00	75,000.00	4,389.00
Mexico Operations	Mexico Operations	32,683.00	35,000.00	(2,317.00)

The use of the fmCode function is more typically found in conjunction with other calculated member functions. This use is illustrated in more complex examples later in this chapter.

fmProperty Function

Use

The fmProperty function is used to retrieve a dimension member's standard or custom property values based on the type of property requested.

Following is a list of the Dimensions and standard property types that can be requested. Note that property types are case sensitive; write them as shown below in the context of a formula expression:

ACCOUNT

- AccountBehavior
- AccountType
- BalanceType
- ExchangeRateType
- Intercompany

ORGANIZATION

- Functional Currency
- Reporting Entity

TIME

Level

In addition to standard properties, the fmProperty function also retrieves custom properties created by the user.

When used independently of other calculated member functions, place this calculated member on a dimension other than the dimension that it references in its formula. The following example illustrates this.

Syntax

fmProperty(“<dimension code name>”,”<propertytype>”)

Example

Functional Currency =
fmProperty("ORGANIZATION","FunctionalCurrency")

Product Product Total
Account Gross Profit
Currency USD
Frequency PA
Time Jan 2012

	Functional Currency	Actual	Budget	Variance
Worldwide Operations	USD	1,205,709.00	1,115,000.00	90,709.00
U.S. Operations	USD	420,397.00	415,000.00	5,397.00
Sales	USD	420,397.00	415,000.00	5,397.00
European Operations	EUR	673,240.00	590,000.00	83,240.00
European Sales	EUR	673,240.00	590,000.00	83,240.00
Canada Operations	CAD	79,389.00	75,000.00	4,389.00
Canada Sales	CAD	79,389.00	75,000.00	4,389.00
Mexico Operations	MXN	32,683.00	35,000.00	(2,317.00)
Mexico Sales	MXN	32,683.00	35,000.00	(2,317.00)

The fmProperty function is useful when it is paired with other calculated member functions and IF statements. See “Scenario 2” (page 80) in “Excel-Based Calculated Members and Display Styles” for an example of a more complex calculated member formula using IF and fmProperty.

To illustrate the use of Custom Properties, the following table displays the custom property values assigned to various Customer Dimension members.

Example

Manager =

fmProperty(“Customer”,”Manager”)

Organization Worldwide Operations
Product Product Total
Analysis Actual
Currency USD
Frequency PTD
Time Feb 2012

	Manager	Sales
Customer Total	Robinson	4,218,600.00
Partner	Wilson	931,307.00
HAL	Wilson	95,324.00
Accents	Wilson	459,385.00
Grand Hampton	Wilson	376,598.00
Direct	Montgomery	1,300,916.00
Buy Best	Montgomery	581,932.00
SW Mart	Montgomery	678,119.00
Radio City	Montgomery	40,865.00
Resellers	Young	857,517.00
Jones Distributing	Young	144,862.00
Westco	Young	712,655.00
Government	McKnight	1,128,860.00
DOD	McKnight	492,218.00
Civilian	McKnight	288,674.00
State & Local	McKnight	347,968.00

fmRate Function

Use

The fmRate function resolves to a CDARate function. This function is used to retrieve the driver rate(s) based on the rate type requested. Place this calculated member function on a dimension other than the dimension in which the rate set is defined on. To illustrate, assume that an estimated tax rate table was defined based on the IntOrg dimension. See the following example.

Driver Rate Table in SAS Financial Management Studio

Driver rate type: Tax Rate

Rate	Internal Organization
0.330000	Accounting
0.330000	Administration
0.410000	Canada Operations
0.410000	Canada Sales
0.330000	Central
0.330000	Corporate Comm
0.330000	Corporate Legal
0.330000	Eastern
0.430000	European Operations
0.430000	European Sales
0.330000	Facilities
0.330000	Finance
0.410000	HR Canada
0.410000	HR Europe
0.410000	HR Mexico
0.330000	Human Resources
0.330000	Information Systems
0.410000	IT Support - Canada
0.430000	IT Support - Europe

In this rate table, each rate is defined by the specified Internal Organization member. If a rate is assigned without an Internal Organization member, it is assumed that any Internal Organization members that are not explicitly stated in the table uses this generic rate.

Syntax

fmRate(“<rate type code”)

Example

Est Tax Rates =
fmRate(“Tax Rate”)

Product ... Product Total
Customer ... Customer Total
Analysis ... Budget
Currency ... USD
Frequency ... PA
Time ... Jan 2012

	Net Income	Income Tax	Est Tax Rates	Income before Taxes
Worldwide Operations	683,450.00	431,550.00	0.33	1,115,000.00
U.S. Operations	278,050.00	136,950.00	0.33	415,000.00
Sales	278,050.00	136,950.00	0.33	415,000.00
European Operations	336,300.00	253,700.00	0.43	590,000.00
European Sales	336,300.00	253,700.00	0.43	590,000.00
Canada Operations	44,250.00	30,750.00	0.41	75,000.00
Canada Sales	44,250.00	30,750.00	0.41	75,000.00
Mexico Operations	24,850.00	10,150.00	0.29	35,000.00
Mexico Sales	24,850.00	10,150.00	0.29	35,000.00

fmXRate Function

Use

The fmXRate function is used to retrieve simple exchange rates for a given time period based on the exchange rate type. Following are a list of simple rates and the correct syntax as they should be used in the formula expression. Note that exchange rate types are case sensitive; write them as shown below in the context of a formula expression:

- PeriodAverage
- PeriodClose
- PeriodOpen
- Custom1
- Custom2

Since exchange rates are defined by time periods, do not place this calculated member function in the Time dimension. The result of this formula renders to a CDAXRate function in the table. The following example displays how this formula provides users more insight to the exchange rates stored in SAS Financial Management Studio.

Syntax

fmXRate("<exchange rate type code>","from currency code", “to currency code")

Example

Period Avg Fx Rate – 1 USD per CAD =

fmXrate("PeriodAverage","USD","CAD")

Example of Resulting CDA Formula

CDAXRate("Model","PeriodAverage","USD","CAD","BUDGET","JAN2013")

Organization	Canada Sales
Product	Product Total
Customer	Customer Total
Analysis	Budget
Currency	USD
Frequency	PTD

	Jan 2013	Feb 2013	Mar 2013
Period Avg FxRate - 1 USD per CAD	1.012648	1.011900	1.010693
Net Income	1,048.74	1,370.19	1,809.65

fmCXRate Function

Use

The fmCXRate function is used to retrieve complex exchange rates such as historic and derived rates. Since exchange rates are defined by time periods, do not place this calculated member function in the Time dimension.

The function should include:

- reference to the specific exchange rate type
- to-and-from currencies
- any dimension code(s) or member code(s) to reference from any additional dimension(s) in which the member is defined

All other dimension code/member code combinations are retrieved from the slicers and either row or column members. The following example displays how this formula provides more insight to the exchange rates stored in SAS Financial Management Studio.

Syntax

fmCXRate("<exchange rate type code>","<from currency code>","<to currency code>", "<Dimension Code>","<Member Code>")

Example

Hist Rate – Common Stock - 1 USD per CAD =
fmCXRate("Historic","USD","CAD","ACCOUNT","E12000")

Example of Resulting CDA Formula

CDACXRate("Model","RATE_TYPE","Historic","USD","CAD","ACTUAL", "JAN2012", "ACCOUNT", "Common Stock", "ORGANIZATION", "Admin Canada", "CUSTOM1 (Product)", "Product Total.vc", "CUSTOM2 (Customer)", "Customer Total.vc", "FREQUENCY", "PTD", "ORGANIZATION_TRADER", "ALL", "SOURCE", "Total")

Organization Admin Canada
Analysis Actual
Currency CAD
Frequency PTD

	Jan 2012	Feb 2012	Mar 2012
Hist Rate - Common Stock - 1 USD per CAD	1.012146	1.037344	1.026483
Common Stock	(100,000.00)	(120,000.00)	(125,000.00)

Excel-based calculated member functions can also be combined with other Excel-based calculated member functions in a single formula. This enhances the member functions' usefulness as well as providing more dynamic formulas and flexibility. The following examples illustrate more complex formulas that can be created through the use of multiple functions within a single formula.

The following example combines the use of the fmValue function and fmProperty function. This formula is useful when account balances are not displayed with the default setting.

Example

Variance =

=IF(fmProperty("ACCOUNT","BalanceType")="CREDIT",fmValue("ACTUAL")-fmValue("BUDGET"),fmValue("BUDGET")-fmValue("ACTUAL"))

Organization Sales
Product Product Total
Customer Customer Total
Currency USD
Frequency PTD
Time Jan 2012

	Actual	Budget	Variance
Net Income	69,411.33	68,507.50	903.83
Income Tax	34,187.67	33,742.50	(445.17)
Income before Taxes	103,599.00	102,250.00	1,349.00
Operating Expense	118,121.00	92,750.00	(25,371.00)
Gross Profit	221,720.00	195,000.00	26,720.00
Sales	420,397.00	415,000.00	5,397.00
Cost of Sales	198,677.00	220,000.00	21,323.00
Balance Sheet	804.00	1,005.00	201.00
Assets	141,673.33	139,512.50	(2,160.83)
Liabilities	46,458.00	45,000.00	1,458.00
Stockholder's Equity	94,411.33	93,507.50	903.83
Ending Headcount	4.00	5.00	1.00

Note that this formula derives its results based on the display state as it appears in Excel. Any changes in display styles such as positive/negative display of debit and credit accounts might impact the results of this formula. More on this topic is discussed in "Excel-Based Calculated Members and Display Styles" (page 80).

The next two formulas are useful to display simple exchange rate values based on the functional currency for the organizational members displayed.

Examples

Avg Fx Rate per 1 USD =

=fmXrate("PeriodAverage","USD",fmProperty("ORGANIZATION", "FunctionalCurrency"))

End of Period Fx Rate per 1 USD =

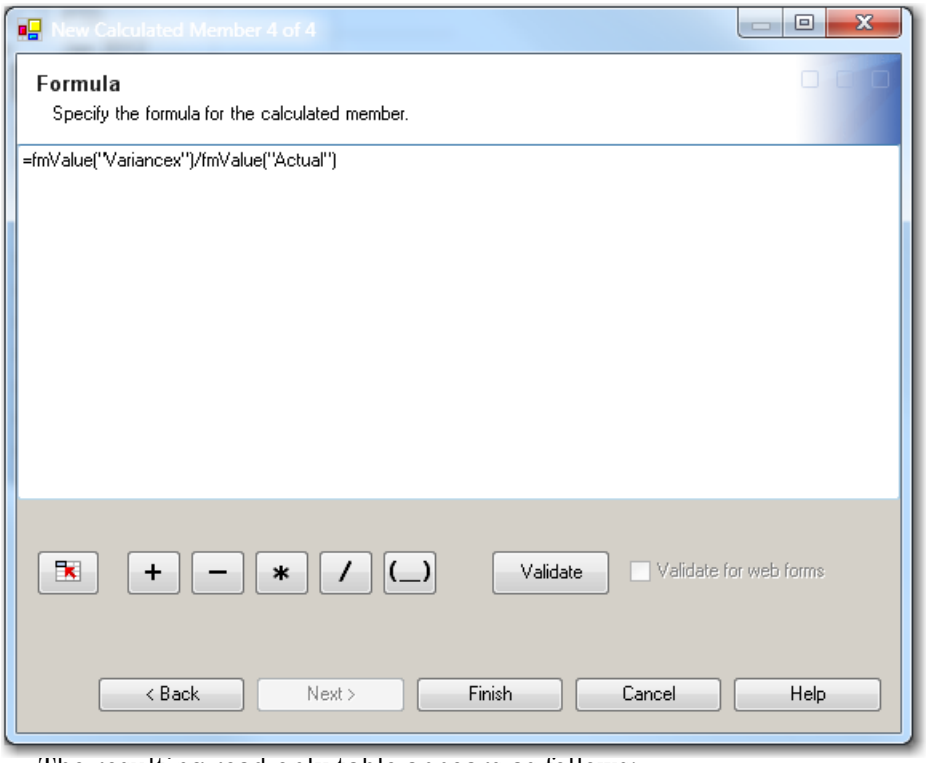
=fmXrate("PeriodClose","USD",fmProperty("ORGANIZATION","FunctionalCurrency"))

Product: Product Total (vc)
Customer: Customer Total (vc)
Analysis: Actual
Currency: USD
Frequency: PTD
Time: Jan 2012

	Worldwide Operations	U.S. Operations	European Operations	Canada Operations	Mexico Operations
Avg Fx Rate per 1 USD	1.000000	1.000000	0.779855	1.013727	13.325658
End of Period Fx Rate per 1 USD	1.000000	1.000000	0.781470	1.017942	13.329601
Net Income	244,064.00	57,029.00	74,963.00	79,389.00	32,683.00
Balance Sheet	(0.01)	0.00	(0.00)	(0.00)	(0.00)
Assets	928,104.80	278,504.80	348,251.00	211,486.00	89,863.00
Liabilities	358,354.69	116,475.80	156,287.00	70,975.89	14,616.00
Stockholder's Equity	569,750.12	162,029.00	191,964.00	140,510.11	75,247.00

Reference to Another Excel-Based Calculated Member in the Same Dimension

Excel-based calculated members can reference other Excel-based calculated members in the same dimension. To illustrate, consider the following example where Variance Percentage is an Excel-based calculated member in the Analysis dimension referencing Variance, also an Excel-based calculated member:

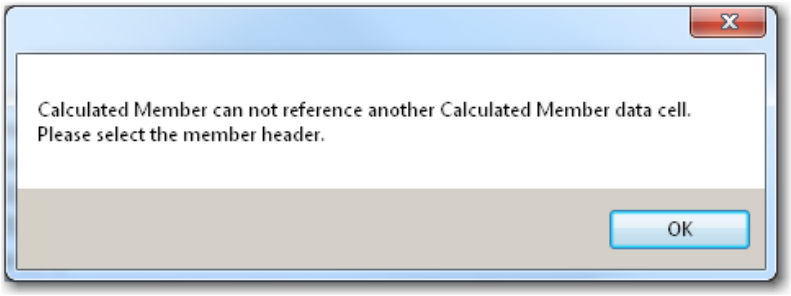


The resulting read-only table appears as follows:

Organization: Sales
Product: Product Total
Customer: Customer Total
Currency: USD
Frequency: PTD
Time: Jan 2012

	Actual	Budget	Variance	Variance Percentage
Net Income	69,411.33	68,507.50	903.83	1.3%
Income Tax	34,187.67	33,742.50	(445.17)	-1.3%
Income before Taxes	103,599.00	102,250.00	1,349.00	1.3%
Operating Expense	118,121.00	92,750.00	(25,371.00)	-21.5%
Gross Profit	221,720.00	195,000.00	26,720.00	12.1%
Sales	420,397.00	415,000.00	5,397.00	1.3%
Cost of Sales	198,677.00	220,000.00	21,323.00	10.7%
Balance Sheet	804.00	1,005.00	201.00	25.0%
Assets	141,673.33	139,512.50	(2,160.83)	-1.5%
Liabilities	46,458.00	45,000.00	1,458.00	3.1%
Stockholder's Equity	94,411.33	93,507.50	903.83	1.0%
Ending Headcount	4.00	5.00	1.00	25.0%

Although Excel-based calculated members can refer to other Excel-based calculated members, they cannot refer to a crossing with a value that is generated by an Excel-based calculated member. A calculated member that refers to such a cell triggers the following message:



Note that the cell is rendered as an absolute reference. Based on the formula expression, Sublease Income is computed as –35% of the Account member Rent. The resulting read-only table and related cell reference are displayed as follows:

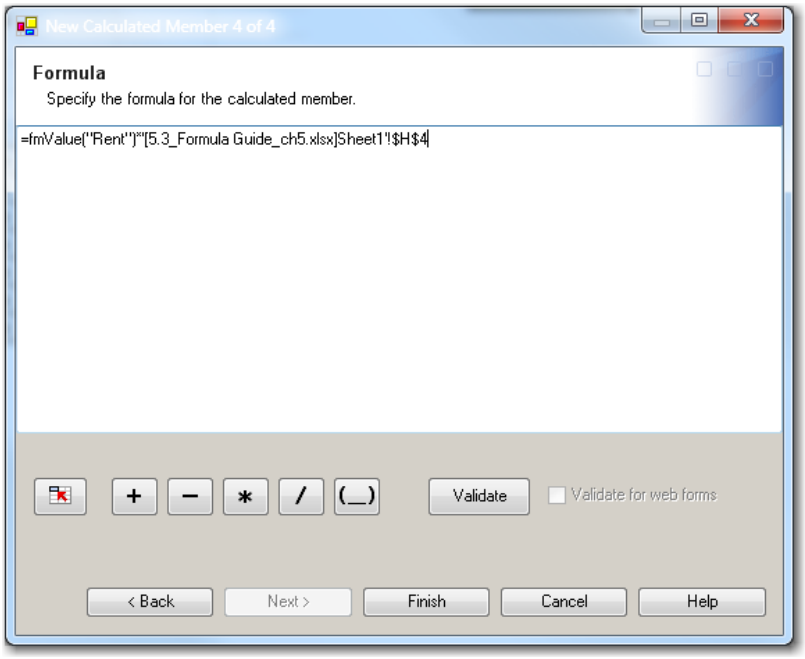
Organization	Facilities
Product	Product Total
Customer	Customer Total
Analysis	Budget
Currency	USD
Frequency	PTD

-0.35

	Jan 2013	Feb 2013	Mar 2013	Q1 2013
Rent	25,000.00	25,000.00	25,000.00	75,000.00
Sublease Income	(8,750.00)	(8,750.00)	(8,750.00)	(26,250.00)
Electric	5,000.00	5,000.00	5,000.00	15,000.00
Water	4,500.00	4,500.00	4,500.00	13,500.00
Repairs & Maintenance	2,500.00	2,500.00	2,500.00	7,500.00
Telecom	1,500.00	1,500.00	1,500.00	4,500.00
I/S Expenses	750.00	750.00	750.00	2,250.00
Other Facilities Expense	500.00	500.00	500.00	1,500.00

Absolute References to Cells in the Same Workbook

Except for the resulting value of an Excel-based calculated member, Excel-based calculated members can refer to any cell in the same workbook, either within or outside of a table. In the following example, Sublease Income is modified to refer to a percentage value outside the table.



While an Excel-based calculated member formula expression enables you to refer to a cell in another workbook, the formula runs only when that other workbook is open. For this reason, it is not recommended to create formulas that refer to cells in other workbooks.

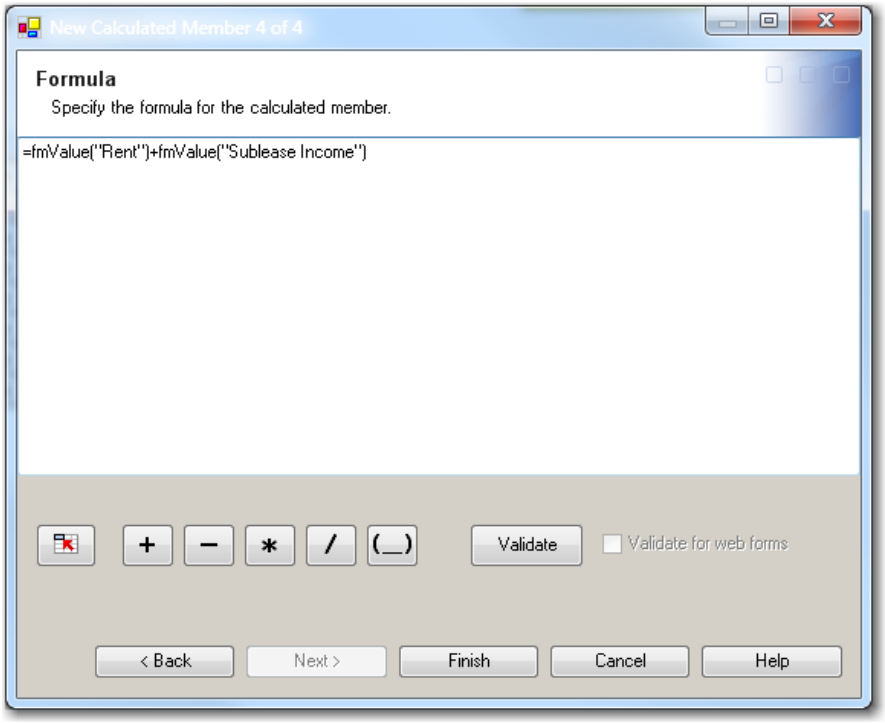
Using Any Excel Function or Valid Excel Expression

Introduction

Excel-based calculated members support the use of all Excel functions with formula expressions. To illustrate, consider the following expression that sums both server-side and Excel-based calculated member values:

Example

Subtotal - Net Rental Expense =
SUM(fmValue("R32000"),fmValue("Est. Sublease Income"))



The following table displays the results for the calculated member Subtotal - Net Rental Expense:

Organization	Facilities
Product	Product Total
Customer	Customer Total
Analysis	Budget
Currency	USD
Frequency	PTD

	Jan 2013	Feb 2013	Mar 2013	Q1 2013
Subtotal - Net Rental Expense	16,250.00	16,250.00	16,250.00	48,750.00
Rent	25,000.00	25,000.00	25,000.00	75,000.00
Sublease Income	(8,750.00)	(8,750.00)	(8,750.00)	(26,250.00)
Electric	5,000.00	5,000.00	5,000.00	15,000.00
Water	4,500.00	4,500.00	4,500.00	13,500.00
Repairs & Maintenance	2,500.00	2,500.00	2,500.00	7,500.00
Telecom	1,500.00	1,500.00	1,500.00	4,500.00
I/S Expenses	750.00	750.00	750.00	2,250.00
Other Facilities Expense	500.00	500.00	500.00	1,500.00

Excel-Based Calculated Members and Display Styles

Introduction

Since fmValue resolves to a cell reference, changes to default styles in Excel such as the display of debit and credit values in a table can affect the resulting Excel-based calculated member result. The following examples illustrate the behavior in three scenarios:

Scenario 1: Excel Styles Set to Display Default for Debit and Credit Balances

Properties defined on the Model in SAS Financial Management Studio are set to display debit accounts with a debit balance set to positive and credit accounts with credit balances as negative numbers.

On the table, Table Properties are set to display debit accounts with a debit balance as Default and credit accounts with credit balances as Default.

Variance = fmValue("BUDGET")-fmValue("ACTUAL")

Organization	Sales
Product	Product Total
Customer	Customer Total
Currency	USD
Time	Jan 2012
Frequency	PTD

	Actual	Budget	Variance
Net Income	(69,411.33)	(68,507.50)	903.83
Income Tax	34,187.67	33,742.50	(445.17)
Income before Taxes	(103,599.00)	(102,250.00)	1,349.00
Operating Expense	118,121.00	92,750.00	(25,371.00)
Gross Profit	(221,720.00)	(195,000.00)	26,720.00
Sales	(420,397.00)	(415,000.00)	5,397.00
Cost of Sales	198,677.00	220,000.00	21,323.00

With the members referenced in the Excel-based calculated member on the table in this example, the formula expression for a single crossing resolves as follows:

=D11-C11

Scenario 2: Excel Styles Set to Display Credit Balances as Positive

Properties defined on the Model in SAS Financial Management Studio are set to display debit accounts with a debit balance set to positive and credit accounts with credit balances as negative numbers.

On the table, Table Properties are set to display debit accounts with a debit balance as Default and credit accounts with credit balances as *Positive*.

In order to return the correct results, the formula is modified as follows:

Variance =
IF(fmProperty("ACCOUNT","BalanceType")="CREDIT",fmValue("ACTUAL")-fmValue("BUDGET"),fmValue("BUDGET")-fmValue("ACTUAL"))

Organization Sales
Product Product Total
Customer Customer Total
Currency USD
Time Jan 2012
Frequency PTD

	Actual	Budget	Variance
Net Income	69,411.33	68,507.50	903.83
Income Tax	34,187.67	33,742.50	(445.17)
Income before Taxes	103,599.00	102,250.00	1,349.00
Operating Expense	118,121.00	92,750.00	(25,371.00)
Gross Profit	221,720.00	195,000.00	26,720.00
Sales	420,397.00	415,000.00	5,397.00
Cost of Sales	198,677.00	220,000.00	21,323.00

With the members referenced in the Excel-based calculated member on the table, the formula expression for a single crossing resolves as follows in this example:

=IF("Credit"="CREDIT",C11-D11,D11-C11)

Scenario 3: Excel Styles Set to Display Credit Balances as Positive and Referenced Members Not Displayed on the Table

Properties defined on the Model in SAS Financial Management Studio are set to display debit accounts with a debit balance set to positive and credit accounts with credit balances as negative numbers.

On the table, Table Properties are set to display debit accounts with a debit balance as Default and credit accounts with credit balances as *Positive*.

The Actual and Budget Analysis members are not displayed on the Analysis Dimension.

Organization Sales
Product Product Total
Customer Customer Total
Currency USD
Time Jan 2012
Frequency PTD

	Forecast	Variance
Net Income	0.00	903.83
Income Tax	0.00	(445.17)
Income before Taxes	0.00	1,349.00
Operating Expense	0.00	(25,371.00)
Gross Profit	0.00	26,720.00
Sales	0.00	5,397.00
Cost of Sales	0.00	21,323.00

Even when the members referenced in the Excel-based calculated member are not on the table, FM Table Property styles are still honored and the Variance is calculated correctly. In order to do this, the resulting formula expression for a given crossing resolves to the following:

=IF("Credit"="CREDIT",(CDAGet("Formula Guide Model", "Account", "Net Inc", "Analysis", "Actual", "Organization", "Sales", "Product", "Product Total", "Customer", "Customer Total", "CURRENCY", "USD", "Time", "Jan 2012", "FREQUENCY", "PTD") * TableCreditNeg("NewTable0") / TableScale("NewTable0"))-(CDAGet("Formula Guide Model", "Account", "Net Inc", "Analysis", "Budget", "Organization", "Sales", "Product", "Product Total", "Customer", "Customer Total", "CURRENCY", "USD", "Time", "Jan 2012", "FREQUENCY", "PTD") * TableCreditNeg("NewTable0") / TableScale("NewTable0")),(CDAGet("Formula Guide Model", "Account", "Net Inc", "Analysis", "Budget", "Organization", "Sales", "Product", "Product Total", "Customer", "Customer Total", "CURRENCY", "USD", "Time", "Jan 2012", "FREQUENCY", "PTD") * TableCreditNeg("NewTable0") / TableScale("NewTable0"))-(CDAGet("Formula Guide Model", "Account", "Net Inc", "Analysis", "Actual", "Organization", "Sales", "Product", "Product Total", "Customer", "Customer Total", "CURRENCY", "USD", "Time", "Jan 2012", "FREQUENCY", "PTD") * TableCreditNeg("NewTable0") / TableScale("NewTable0")))

The additional information in the formula expression occurs when the member(s) referenced in the formula are removed from the table. This additional detail provides the information to support associated styles such as scaling, debit, and credit settings selected in Excel Properties.

Formulas That Run Out of Bounds

In some scenarios, Excel-based calculated members can be written such that the member (or members) referenced in the expression are “out of bounds”; that is, the member is either not on the table or the member is not in the cycle or model. These scenarios most typically occur when an expression uses a time offset.

When a member is in the cycle and model but not on the table, an Excel-based calculated member resolves to a CDAGet formula and the requested value is retrieved. When the member is in the cycle, but not in the model or the member is not in the cycle or model, the requested result is a zero.

Consider the following Excel-based calculated member:

Beginning Bal - Equipment =
fmValue("Ending Balance - Equipment”","TIME",-1)

Organization	Worldwide Operations
Product	Product Total
Customer	Customer Total
Analysis	Actual
Currency	USD
Frequency	PTD

	Jan 2012	Feb 2012	Mar 2012
Ending Balance - Equipment	57,832.00	57,832.00	57,832.00
Additions - Equipment	55,276.00	55,276.00	55,276.00
Disposals - Equipment	(5,692.00)	(5,692.00)	(5,692.00)
Transfers & WO - Equipment	8,248.00	8,248.00	8,248.00
Beginning Balance - Equipment	0.00	57,832.00	57,832.00

In the example above, the resulting value for Beginning Balance - Equipment for Jan 2012 is 0.00 since Dec 2011 is not in the cycle or model.

How Calculated Members are Displayed in Excel?

Excel-based calculated members are displayed in the following scenarios:

Scenario 1. Time Offsets

Example =fmValue("Actual","Time",-1)

Excel The resulting value is displayed.

Scenario 2. Referencing a Member or Hidden Member Not on the Table

Example =fmValue("Actual") where actual is not displayed on the table

Excel The resulting value is displayed.

Scenario 3. Placing an Excel-Based Calculated Member Before or After a Member or Hidden Member Not on the Table

Example =fmValue("Variance to Last Year") which is placed after the Analysis member Last Year and then Last Year is not displayed on the table

Excel The resulting value is displayed.

Excel-Based Calculated Members and Supplemental Tables

Based on the design of Supplemental tables, Excel-based calculated members can be placed only on the dimension displayed on the column. Also, these Excel-based calculated members can reference members only on the columns or slicers. This design limits the recommended selection of Excel-based calculated members to fmValue, fmCode and fmProperty.

Since fmRate, fmXRate, and fmCXRate render to CDA expressions, Supplemental tables do not render results for these functions due to the fact that CDA formulas are not supported for Supplemental tables.

CHAPTER 6

Advanced Formula Concepts

Fixed Members ... *85*
Explicit Members and Implicit (Fixed) Members ... *85*
Explicit versus Implicit: Who Wins? ... *90*
Formula Scope ... *92*
Formula Scope and Member Selection Rules ... *99*
Defining Multiple Formulas on One Member ... *103*
Introduction ... *103*
Uses of the SUBSTR Function ... *103*
Member-Based Uses of the IF and NESTIF Functions ... *103*
Value-Based Uses of the IF and NESTIF Functions ... *103*
Ranking Multiple Formulas ... *105*
Introduction ... *105*
Formula 1 ... *105*
Formula 2 ... *105*

Fixed Members

Explicit Members and Implicit (Fixed) Members

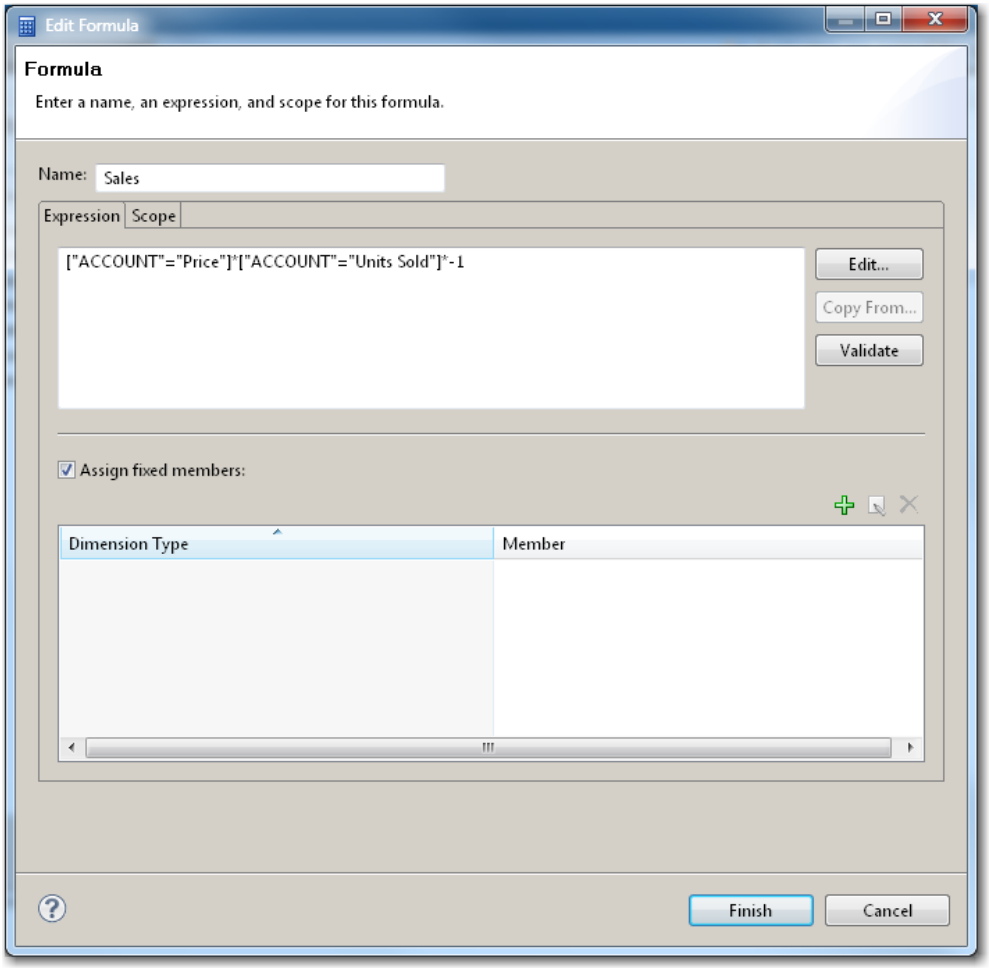
All SAS Financial Management formulas can contain two types of members: *explicit members* and *implicit members*.

- Explicit members are members that are clearly defined within the formula expression.
- Implicit members are members that are implied, but not directly expressed, in the formula expression.

Fixed members are considered to be implicit members that can be used as an optional means of limiting where a formula expression reads from. You can explicitly state which members a formula should read from in the formula expression. However, fixed members offer an easy alternative to assigning members that apply to each formula input in the formula expression.

You can select only one member *per dimension* as a fixed member.

1 Use the Assign fixed check box in the Formula wizard to select fixed members on an expression-by-expression basis.



2 Click the green plus icon to select a fixed member.

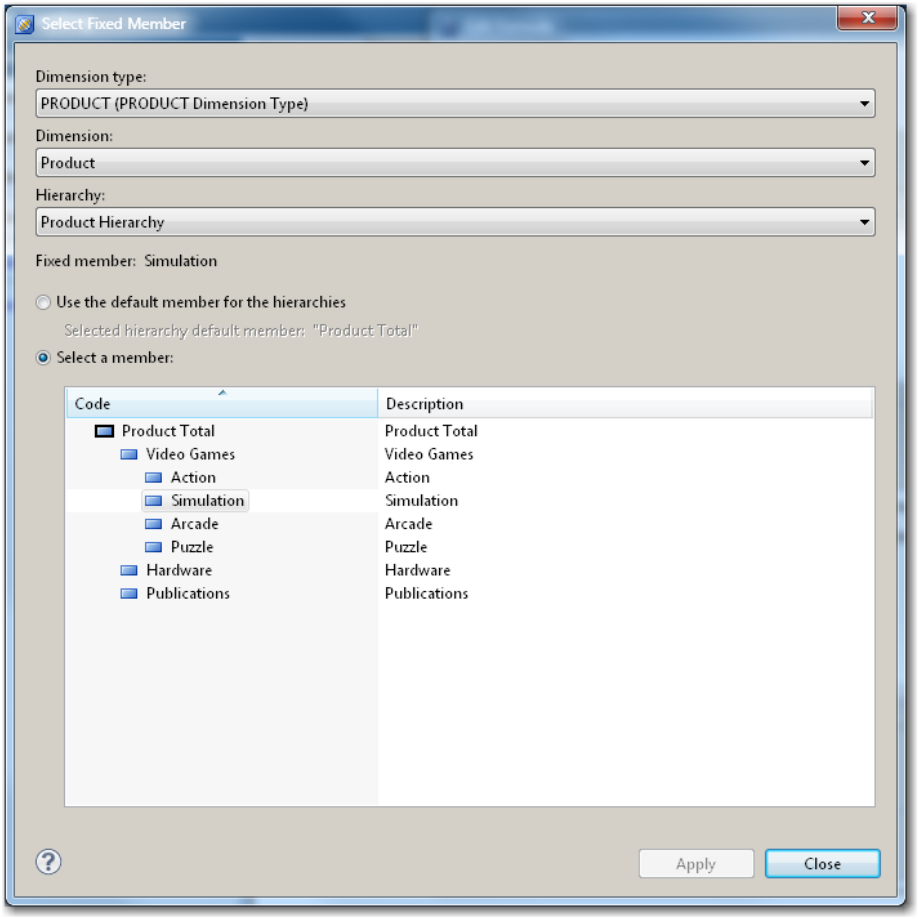
3 Select the desired dimension type, dimension, and hierarchy.

Members are displayed in the context of the current view of the selected dimension and hierarchy.

4 Select the default member of a given hierarchy or a different member.

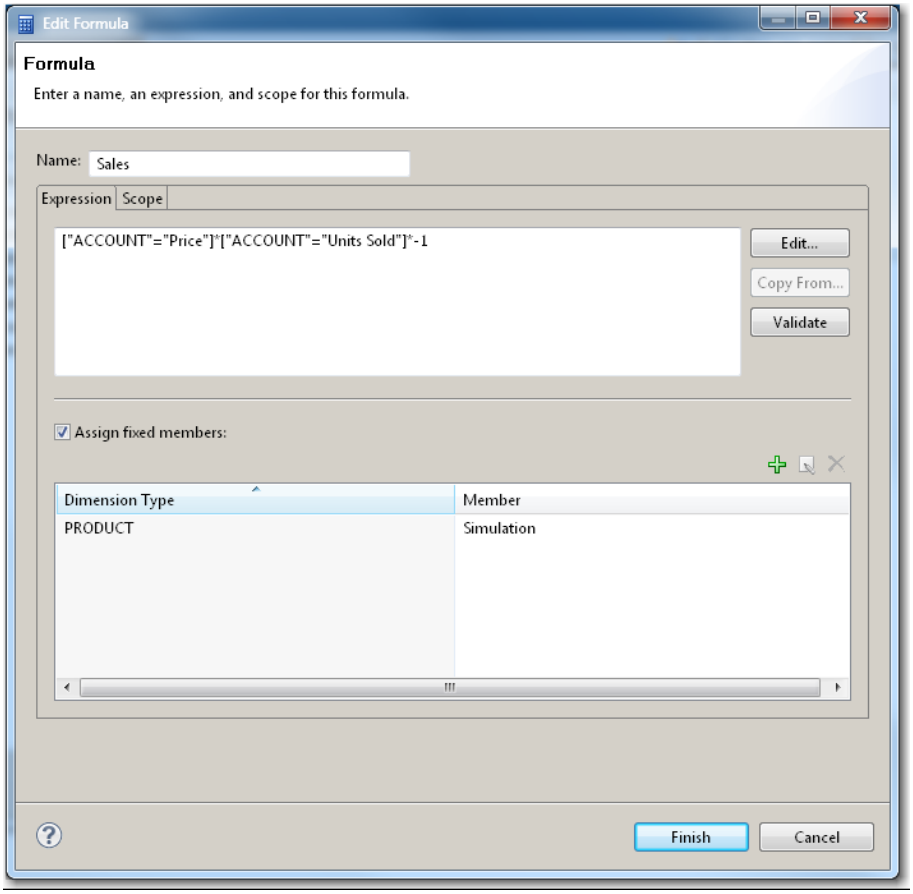
Either option allows for only one default member per dimension type.

5 Highlight the desired member and click **Apply**.



6 Click **Close** to return to the **Formula** tab.

7 View the formula expression and selected Fixed members.



The results of the fixed member selection for a given dimension are displayed. You can continue to select additional fixed members for different dimensions, modify current selections, and/or remove current selections.

For existing calculated members, fixed member selection is available through **Member Properties ▶ Formulas ▶ Expression**.

To illustrate how fixed members work, consider the following modeling formula example:

Formula

Sales = ["ACCOUNT"="Price"]*["ACCOUNT" ="Units Sold"]*-1

Fixed Members

Product dimension: Simulation

Adding Product member Simulation as a fixed member has the same effect as if the formula expression were written as follows:

Sales = ["ACCOUNT"="Price"]["PRODUCT"="Simulation"]*["ACCOUNT" ="Units Sold"]["PRODUCT"="Simulation"]*-1

Facts

The following data entry table displays the facts that were entered for Price and Units Sold for the leaf members under Product member Video Games.

Organization Eastern
Customer Radio City
Analysis Budget
Currency USD
Time Jan 2013
Frequency PTD
Source Form Data
TRADER External

	Video Games	Video Games (vc)	Action	Simulation	Arcade	Puzzle
Price		0.00	15.00	20.00	25.00	30.00
Units Sold	700.00	0.00	100.00	150.00	200.00	250.00

Results

In the following table, results for Sales for all leaf members under Product member Video Games are (3,000.00). This is computed by multiplying the Price for Simulation by the Units Sold of Simulation: (20 x 150)* -1 = (3,000.00).

Organization Eastern
Customer Radio City
Analysis Budget
Currency USD
Frequency PTD
Time Jan 2013
Source Form Data
TRADER External

	Video Games	Video Games (vc)	Action	Simulation	Arcade	Puzzle
Price		0.00	15.00	20.00	25.00	30.00
Units Sold	700.00	0.00	100.00	150.00	200.00	250.00
Sales	(15,000.00)	(3,000.00)	(3,000.00)	(3,000.00)	(3,000.00)	(3,000.00)

Since the Sales account is a modeling formula, the Sales account result for product Video Games is (15,000.00). This is the sum of all the subordinate leaf members. Note that the formula runs at Video Games.vc as well. This is the intended behavior and should be considered when using fixed members.

Explicit versus Implicit: Who Wins?

The previous section provides an example of a formula expression with an implicit reference to Product member Simulation. Alternatively, you can write the formula expression with explicit references, rendering the same results. The modified formula with explicit references is as follows:

Sales = ["PRODUCT"="Simulation"]["ACCOUNT"="Price"]*["PRODUCT"="Simulation"] ["ACCOUNT"="Units Sold"]* -1

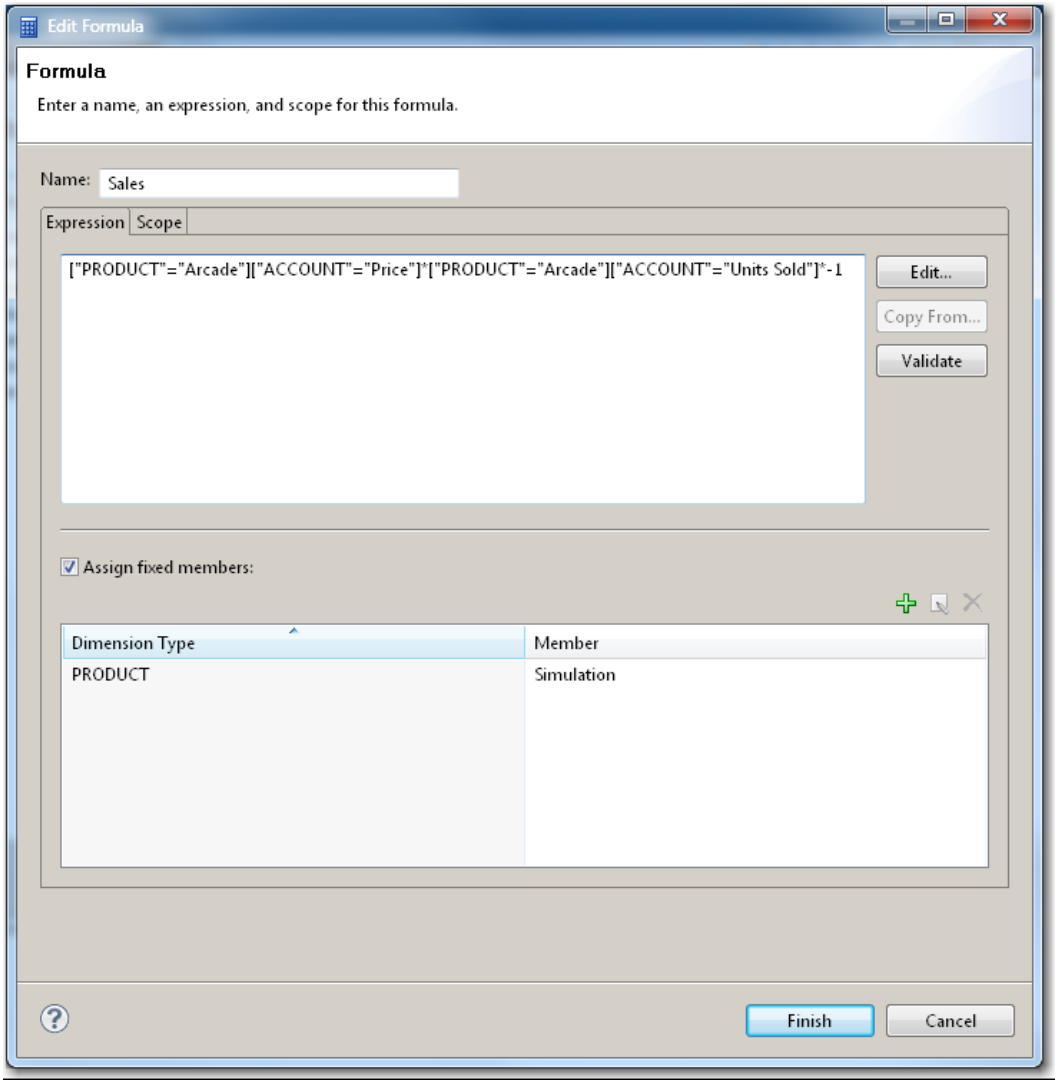
In some scenarios, you might use a fixed member that conflicts with explicit references in a formula expression. In such cases, the explicitly stated member wins, and the implicitly stated member is ignored. In the following example, the formula expression explicitly states Product member Arcade, while the Fixed member selection implicitly states Product member Simulation.

Formula

Sales = ["PRODUCT"="Arcade"]["ACCOUNT"="Price"]*["PRODUCT"="Arcade"] ["ACCOUNT"="Units Sold"]* -1

Fixed Members

Product dimension: Simulation



Facts

The following data-entry table displays the facts that were entered for Price and Units Sold for the leaf members under Product member Video Games.

Organization Eastern
Customer Radio City
Analysis Budget
Currency USD
Time Jan 2013
Frequency PTD
Source Form Data
TRADER External

	Video Games	Video Games (vc)	Action	Simulation	Arcade	Puzzle
Price		0.00	15.00	20.00	25.00	30.00
Units Sold	700.00	0.00	100.00	150.00	200.00	250.00

Results

Organization Eastern
Customer Radio City
Analysis Budget
Currency USD
Frequency PTD
Time Jan 2013
Source Form Data
TRADER External

	Video Games	Video Games (vc)	Action	Simulation	Arcade	Puzzle
Price		0.00	15.00	20.00	25.00	30.00
Units Sold	700.00	0.00	100.00	150.00	200.00	250.00
Sales	(25,000.00)	(5,000.00)	(5,000.00)	(5,000.00)	(5,000.00)	(5,000.00)

In the preceding table, results for Sales for all leaf members under Video Games are (5,000.00). This is computed by multiplying the Price for Arcade by the Units Sold of Arcade: (25 * 200.00)* -1 = (5,000.00).

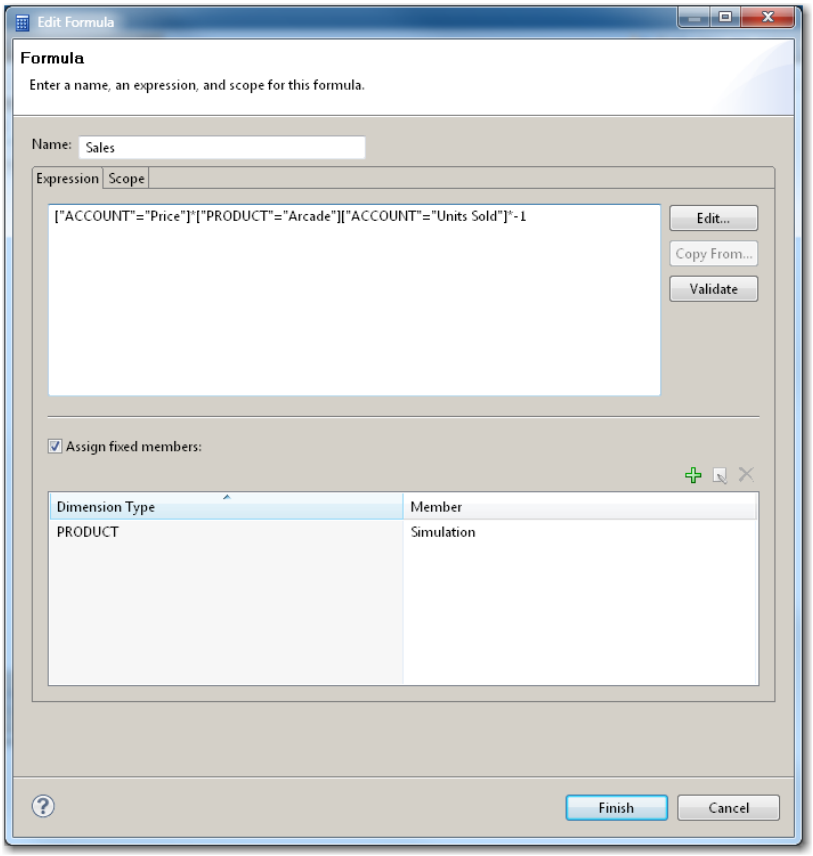
In the following scenario, the formula expression is modified slightly so that the product member Arcade is explicitly stated only with the Units Sold account. The fixed member reference and the facts remain unchanged.

Formula

Sales = ["ACCOUNT"="Price"]*["PRODUCT"="Arcade"] ["ACCOUNT"="Units Sold"]* -1

Fixed Members

Product dimension: –Simulation



Results

Organization	Eastern
Customer	Radio City
Analysis	Budget
Currency	USD
Frequency	PTD
Time	Jan 2013
Source	Form Data
TRADER	External

	Video Games	Video Games (vc)	Action	Simulation	Arcade	Puzzle
Price		0.00	15.00	20.00	25.00	30.00
Units Sold	700.00	0.00	100.00	150.00	200.00	250.00
Sales	(20,000.00)	(4,000.00)	(4,000.00)	(4,000.00)	(4,000.00)	(4,000.00)

In the preceding table, results for Sales for all leaf members under Video Games are (4,000.00). This is computed by multiplying the Price for Simulation by the Units Sold for Arcade: (20 * 200)* -1 = (4,000.00).

Formula Scope

Formula scope is an optional means of restricting where a formula runs. It can be applied on an expression-by-expression basis and is available for all server-side formulas (reporting, driver, and modeling formulas). From a performance standpoint,

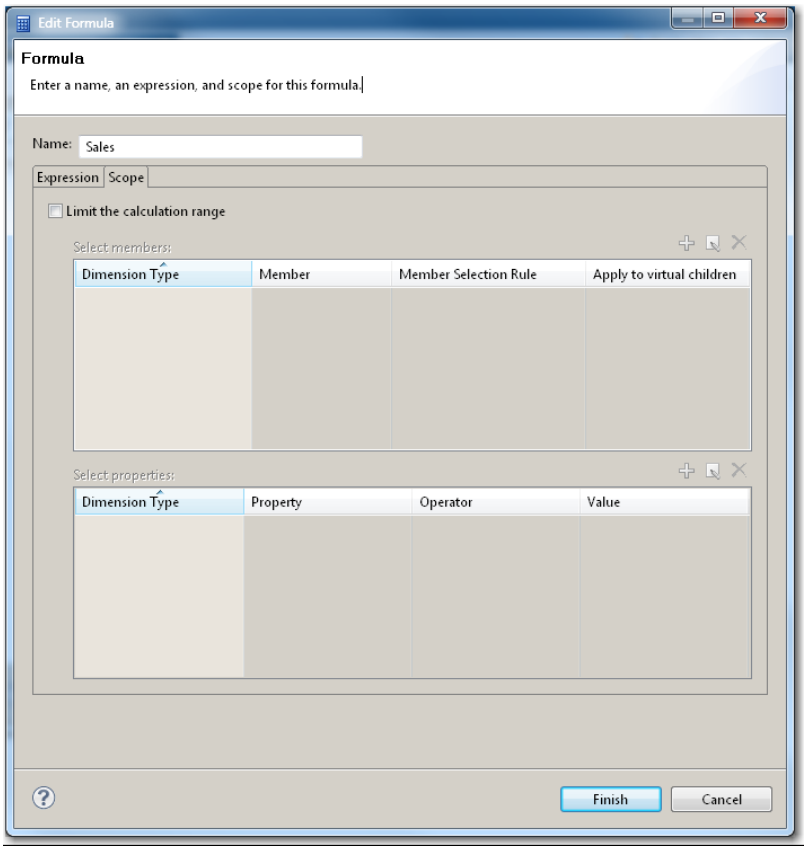
formula scope is most effective when it is used with modeling formulas to limit the number of crossings where a formula runs.

There are two different means of scoping formulas:

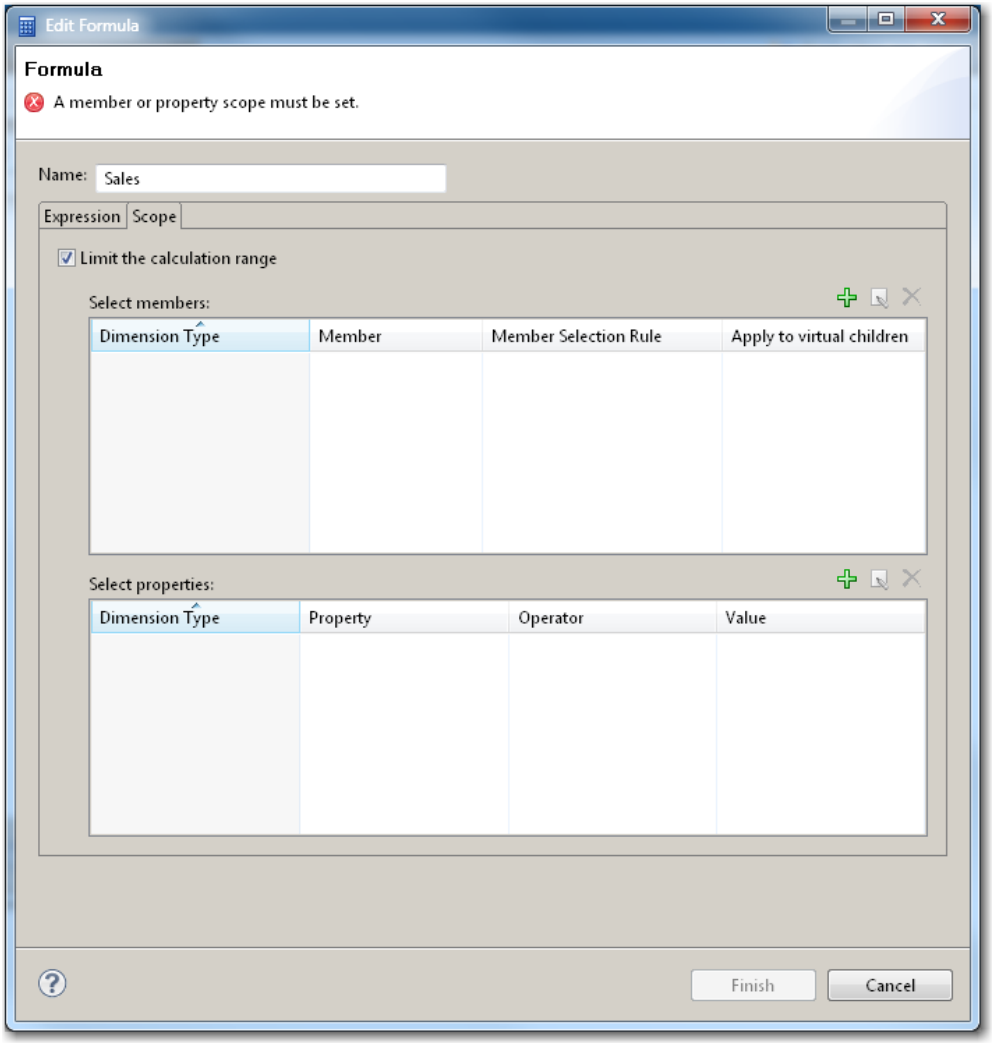
- Member scoping
- Property scoping

Member scoping is based on the assignment of member selection rules on selected members. Property scoping is based on the assignment of operators and available property values for the selected property type. Member and property scoping can be used independently of one another or in combination. Applying both member and property selection rules results in the intersection of all crossings based on the member and property selection rules applied. The following sections illustrate the use and related examples of formula scoping.

In defining formula scope, you can select multiple members per dimension. You define formula scope on an expression-by-expression basis in the Formula wizard, as shown here:

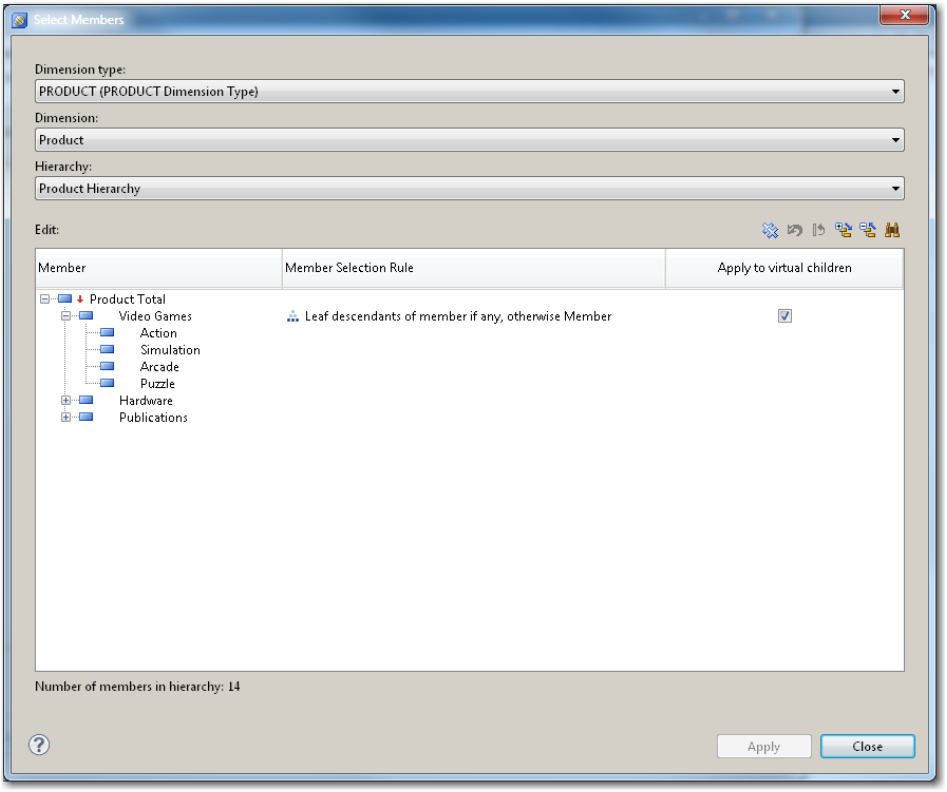


1 Select the **Limit the calculation range** check box.



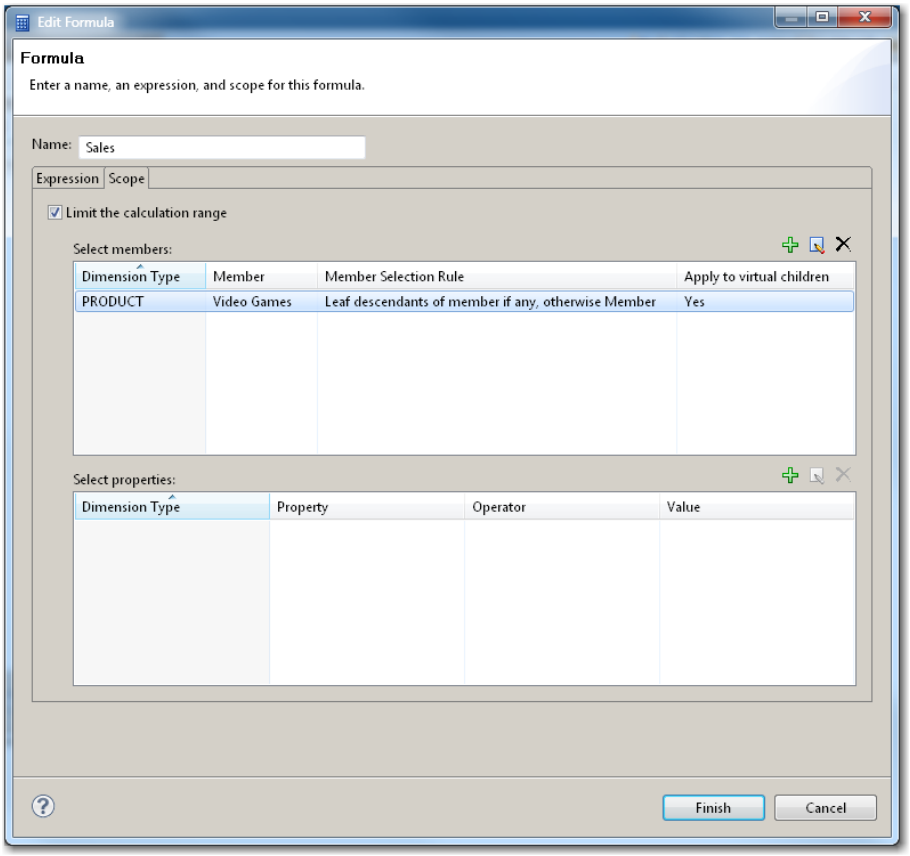
2 Click the green plus icons to select members or properties for scoping.

For member scoping, select the desired dimension type, dimension, and hierarchy. Members are displayed in the context of the current view of the selected dimension and hierarchy. You can select from the available hierarchies and choose multiple members in each dimension. Highlight the desired member(s) and choose the Member Selection Rule associated with the member as well as whether to apply to the virtual children.



3 Click **Apply** and **Close** when member scoping is complete.

The results of specifying member scope for the formula are displayed on the **Scope** tab. You can specify additional members or select properties to scope by.

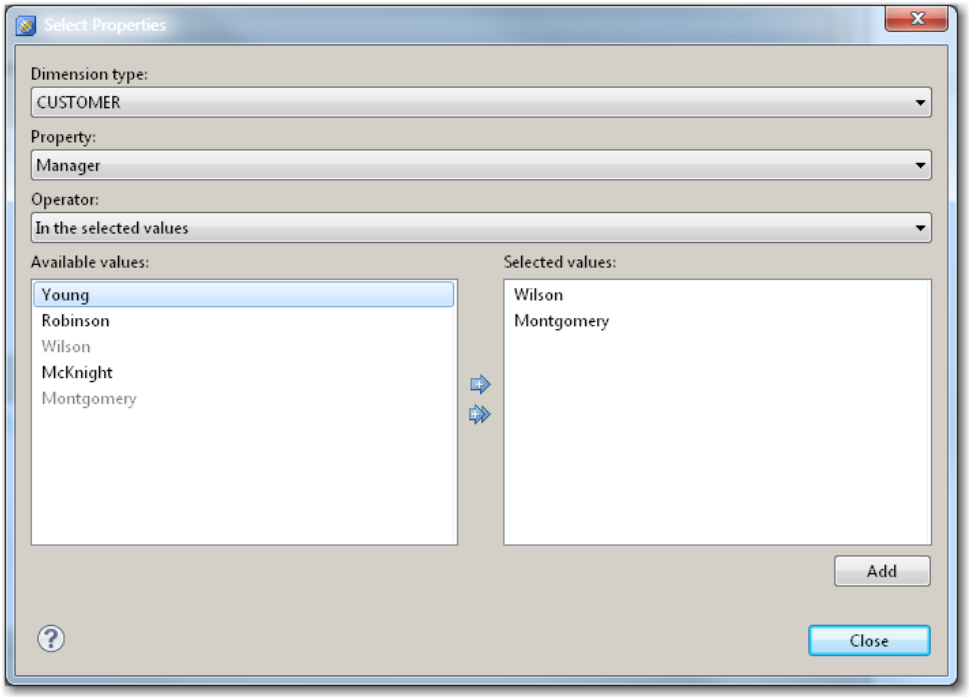


4 To select properties, click the green plus icon.

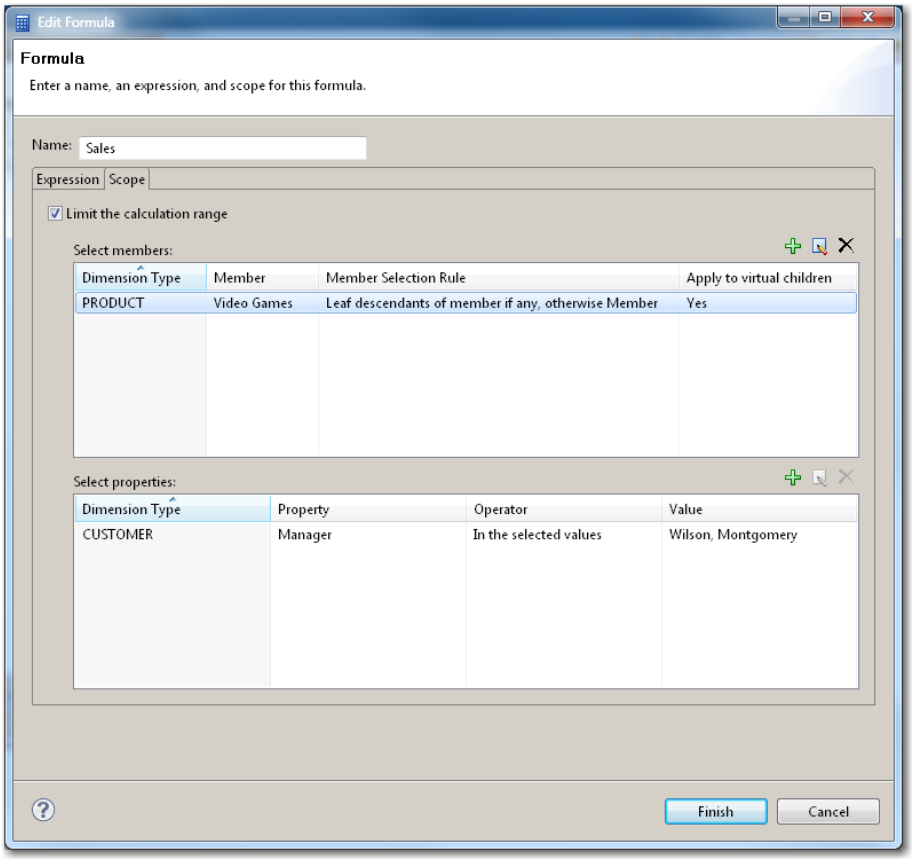
5 Select the desired dimension type, property, and operator.

Property values are displayed in the context of the current view of the selected.

You can select multiple property types and values in each dimension.

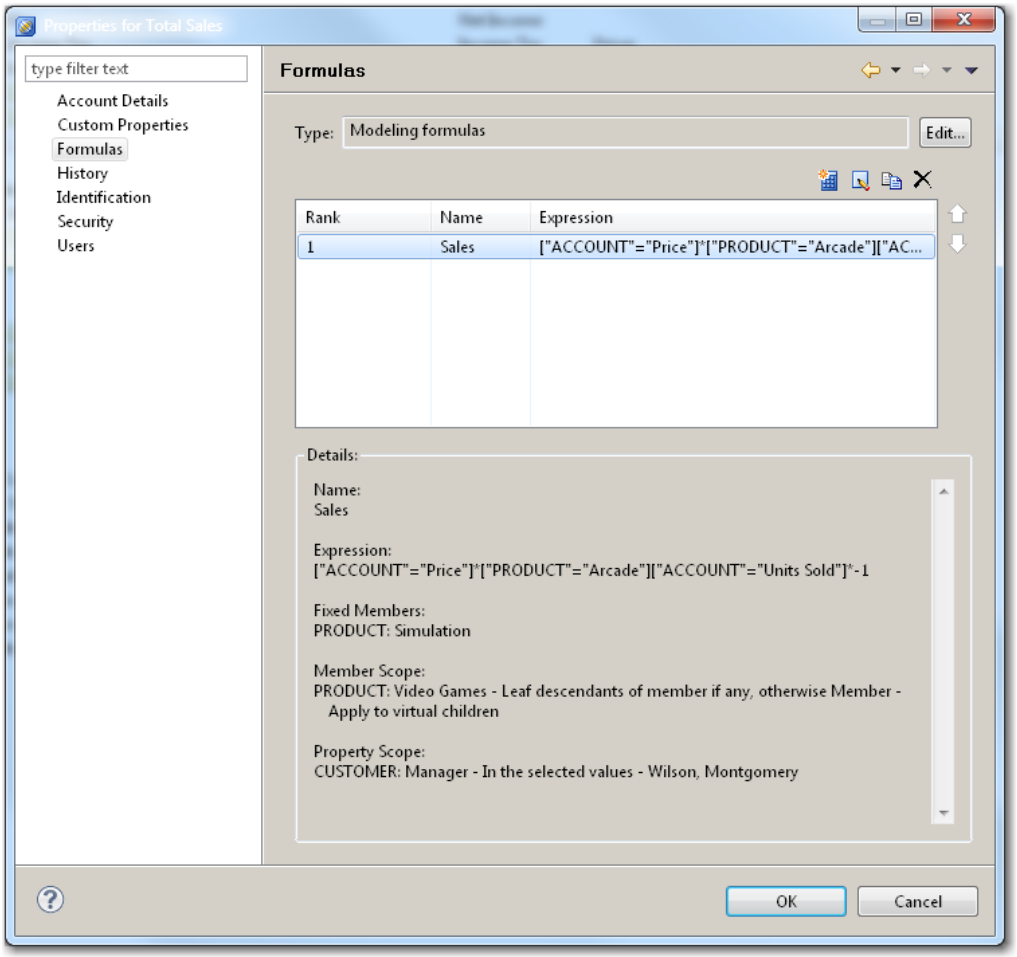


6 Click **Add** and **Close** to see the summary of member and property scoping on the **Scope** tab.



7 Click **Finish** when the formula scoping is complete.

A summary window shows the expression text, fixed members, member, and property scope.



In the following data entry table, you can see the crossings for the Sales accounts that are calculated based on their gray color.

Organization	Sales (vc)
Product	Product Total (vc)
Time	Jan 2013
Analysis	Budget
Currency	USD
Frequency	PTD
Source	Form Data
TRADER	External

		Price	Units Sold	Sales
Customer Total	Robinson		1,360.00	1,868.00
Customer Total (vc)	Robinson	1.00	10.00	0.00
Partner	Wilson		140.00	0.00
Partner (vc)	Wilson	1.10	20.00	0.00
HAL	Wilson	1.20	30.00	0.00
Accents	Wilson	1.30	40.00	0.00
Grand Hampton	Wilson	1.40	50.00	0.00
Direct	Montgomery		300.00	500.00
Direct (vc)	Montgomery	1.50	60.00	90.00
Buy Best	Montgomery	1.60	70.00	112.00
SW Mart	Montgomery	1.70	80.00	136.00
Radio City	Montgomery	1.80	90.00	162.00
Resellers	Young		330.00	0.00
Resellers (vc)	Young	1.90	100.00	0.00
Jones Distributing	Young	2.00	110.00	0.00
Westco	Young	2.10	120.00	0.00
Government	McKnight		580.00	1,368.00
Government (vc)	McKnight	2.20	130.00	286.00
DOD	McKnight	2.30	140.00	322.00
Civilian	McKnight	2.40	150.00	360.00
State & Local	McKnight	2.50	160.00	400.00

Note: Since the calculated member is scope for Manager Property Type values McKnight and Montgomery, the formula execution also applies to the virtual children as well. The values at Direct and Government are due to the aggregation of the driver formula results since driver and modeling formulas run only at leaf-level crossings. Customer members associated with property type values other than McKnight and Montgomery are rendered as yellow, writeable cells at leaf levels for the Sales account.

Formula Scope and Member Selection Rules

As shown in previous examples, the Formula Editor window provides member selection in the context of a hierarchy, displaying the current as-of date view for the selected hierarchy.

When you select a member for formula scope, a member selection rule is assigned. Member selection rules are based on the formula type of the calculated member and the selected dimension type.

For driver and modeling formulas, there are three member selection rules:

- Member available only for the Analysis dimension
- Leaf descendants of Member if any, otherwise Member
- Exclude

For reporting formulas, there are ten member selection rules:

- Member
- Member and its children
- Member and its descendants
- Member and its leaf descendants
- Members and its non-leaf descendants
- Children of member if any, otherwise Member
- Descendants of member if any, otherwise Member
- Leaf descendants of member if any, otherwise Member
- Non-leaf descendants of member if any, otherwise Member
- Exclude

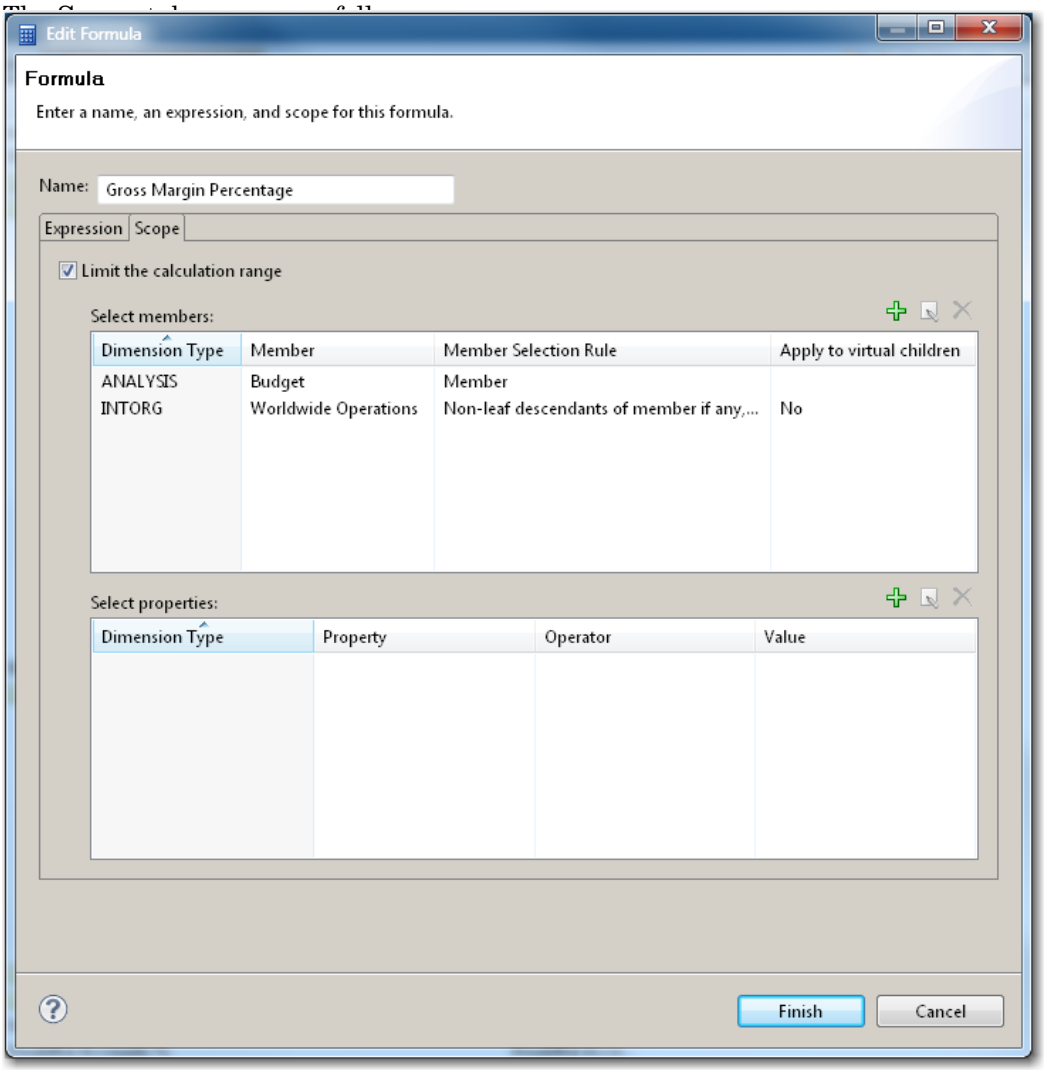
In addition to these member selection rules, the **Apply to virtual children** check box applies the rule selected to the virtual child.

The following scenario illustrates how member selection rules are rendered.

Reporting formula – Gross Profit Percentage

Formula Scope

["ANALYSIS"="Budget"]
["INTORG"= "Worldwide Operations"]



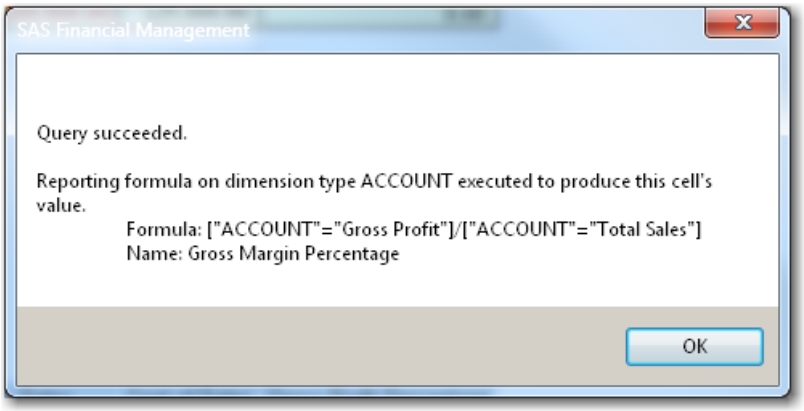
Because the Analysis dimension has only flat hierarchies, the only selection rule available for an Analysis member is **Member**. Because the Worldwide Operations IntOrg member is a roll-up member and the formula is a reporting formula, you can select one of ten member selection rules. For this example, Worldwide Operations is assigned **Non-leaf descendants of member if any, otherwise Member**. You can choose to apply the virtual child to this rule.

The table below displays only the rows that have data in them. Note that the Gross Profit Percentage account member does not return results for Worldwide Operations, Central, Eastern, Western, European Sales, Canada Sales and Mexico Sales. The formula only ran for U.S. Operations, Sales, European Operations, Canada Operations, and Mexico Operations since there was data available and these members all qualified as Non-leaf descendants of Worldwide Operations.

Product: Product Total
Customer: Customer Total
Analysis: Budget
Currency: USD
Frequency: PTD
Time: Jan 2013

	Gross Profit	Sales	Cost of Sales	Gross Profit Percentage
Worldwide Operations	(5,919,000.00)	(7,300,000.00)	3,206,000.00	0.00
U.S. Operations	(4,375,250.00)	(4,925,000.00)	1,781,000.00	0.89
Sales	(4,375,250.00)	(4,925,000.00)	1,781,000.00	0.89
Central	(562,500.00)	(950,000.00)	625,000.00	0.00
Eastern	(3,269,000.00)	(3,100,000.00)	606,000.00	0.00
Western	(543,750.00)	(875,000.00)	550,000.00	0.00
European Operations	(975,000.00)	(1,500,000.00)	900,000.00	0.65
European Sales	(975,000.00)	(1,500,000.00)	900,000.00	0.00
Canada Operations	(443,750.00)	(675,000.00)	400,000.00	0.66
Canada Sales	(443,750.00)	(675,000.00)	400,000.00	0.00
Mexico Operations	(125,000.00)	(200,000.00)	125,000.00	0.63
Mexico Sales	(125,000.00)	(200,000.00)	125,000.00	0.00

Cell Information for U.S. Operations Gross Profit Percentage is as follows:



Defining Multiple Formulas on One Member

Introduction

You can define an unlimited number of formulas on a single calculated member. To improve formula performance, maintenance, and manageability, define multiple formulas on one member as an alternative in the following scenarios:

Scenario 1. Uses of the SUBSTR Function

Example: Bonus Expense = NESTIF

(SUBSTR(CURRENT("INTORG"),2,1)= "2", ["ACCOUNT"="Salaries"]*.25,
(SUBSTR(CURRENT("INTORG"),2,1)= "1", ["ACCOUNT"="Salaries "]*.20,
(SUBSTR(CURRENT("INTORG"),2,1)= "5", ["ACCOUNT"="Salaries "]*.20,
1=1, ["ACCOUNT"="Salaries"]*.15)

Defining multiple formulas on one member is not applicable to value-based uses of the IF and NESTIF functions. The following topic describes this in more detail.

Scenario 2. Member-Based Uses of the IF and NESTIF Functions

Example: Discounts = NESTIF

(["PRODUCT"="Game Controller"],["ACCOUNT"="Sales"]*-0.20, ["PRODUCT"="Joy Stick"],["ACCOUNT"="Sales"]*-0.25, ["PRODUCT"="Flight Stick"],["ACCOUNT"="Sales"]*-0.30)

Scenario 3. Value-Based Uses of the IF and NESTIF Functions

Example: Sales Bonus =
IF(["ACCOUNT"="Sales"]["ANALYSIS"="ACTUAL"]>
["ACCOUNT"="Sales"]["ANALYSIS"="BUDGET"],["ACCOUNT"="Sales"]*-.05,0)

In this example, the formula for the Sales Bonus account tests the numeric values of Actual Sales and Budget Sales.

To define multiple formulas on the same member, use the Formula wizard on the **Formulas** tab of the Member Properties window. To illustrate the creation of multiple formulas on the same member, consider the following formula on the Discount account:

Discounts = NESTIF

(["PRODUCT"=" Game Controller "],["ACCOUNT"="Sales"]*-0.20, ["PRODUCT"="Joy Stick"],["ACCOUNT"="Sales"]*-0.25, ["PRODUCT"="Flight Stick"],["ACCOUNT"="Sales"]*-0.30)

Because this formula contains a member-based use of the NESTIF function, it can be rewritten as three separate formula expressions:

Formula 1

Name: Game Controller
Expression: ["ACCOUNT"="Sales"]*-0.20
Scope: ["PRODUCT"="Game Controller"]

Formula 2

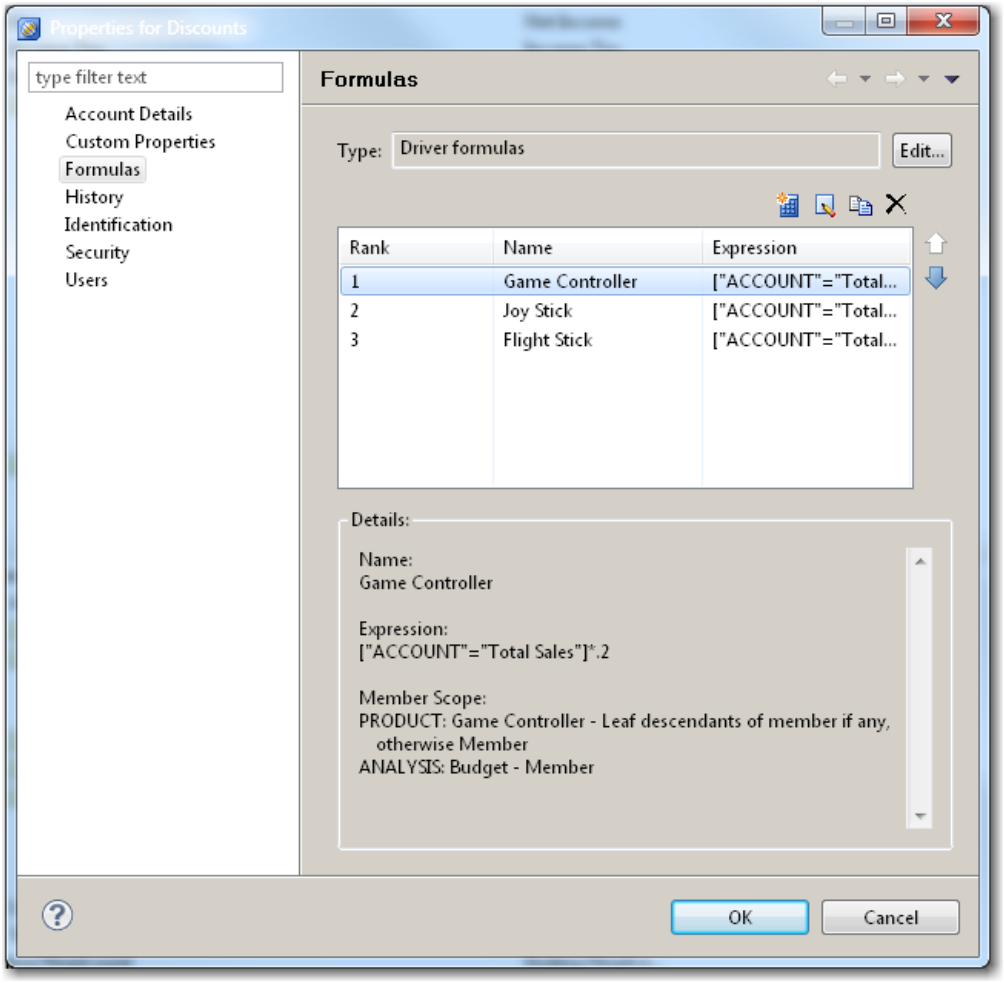
Name: Joy Stick
Expression: ["ACCOUNT"="Sales"]*-0.25
Scope: ["PRODUCT"="Joy Stick"]

Formula 3

Name: Flight Stick
Expression: ["ACCOUNT"="Sales"]*-0.30
Scope: ["PRODUCT"="Flight Stick"]

Follow the steps used to create a new calculated member or to add a formula to an existing member when you define multiple formulas on the same member. For more information about creating formula expressions, see Chapter 4, "Formula Basics for Server-Side Calculated Members." The formula expression name is visible in SAS Financial Management Studio and through **Cell Information** in Excel.

In addition, each formula has its own Fixed member and scope selections. The results of creating a calculated member with multiple formulas are displayed on the **Formulas** tab:



Ranking Multiple Formulas

Introduction

Each formula is assigned a rank when you create it. The initial rank order for a calculated member is based on the order in which the formulas were created. The rank of a formula determines which formula is executed in the event of formula scope overlap.

To illustrate the ranking of formulas, consider the Discounts example in "Defining Multiple Formulas on One Member," modified slightly as follows:

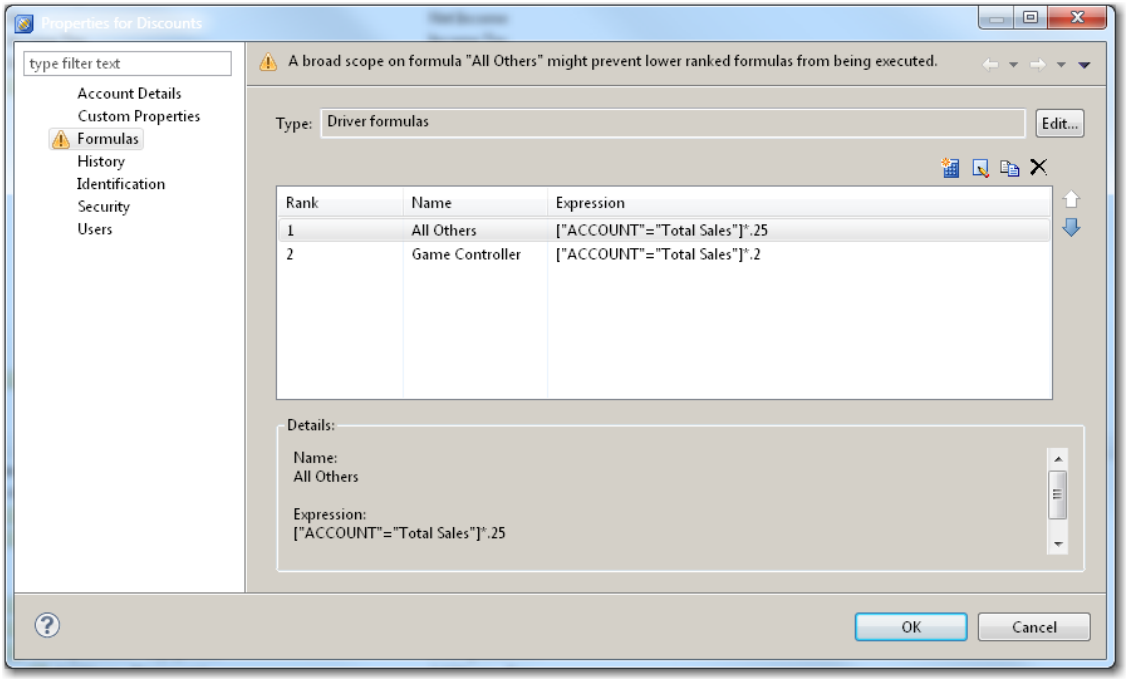
Formula 1

Name: All Others
Expression: ["ACCOUNT"="Sales"]*-0.25
Scope: Not limited

Formula 2

Name: Game Controller
Expression: ["ACCOUNT"="Sales"]*-0.20
Scope: ["PRODUCT"="Game Controller"]

On the **Formulas** tab for the modified Discounts calculated member, note that the formula All Others is ranked first.



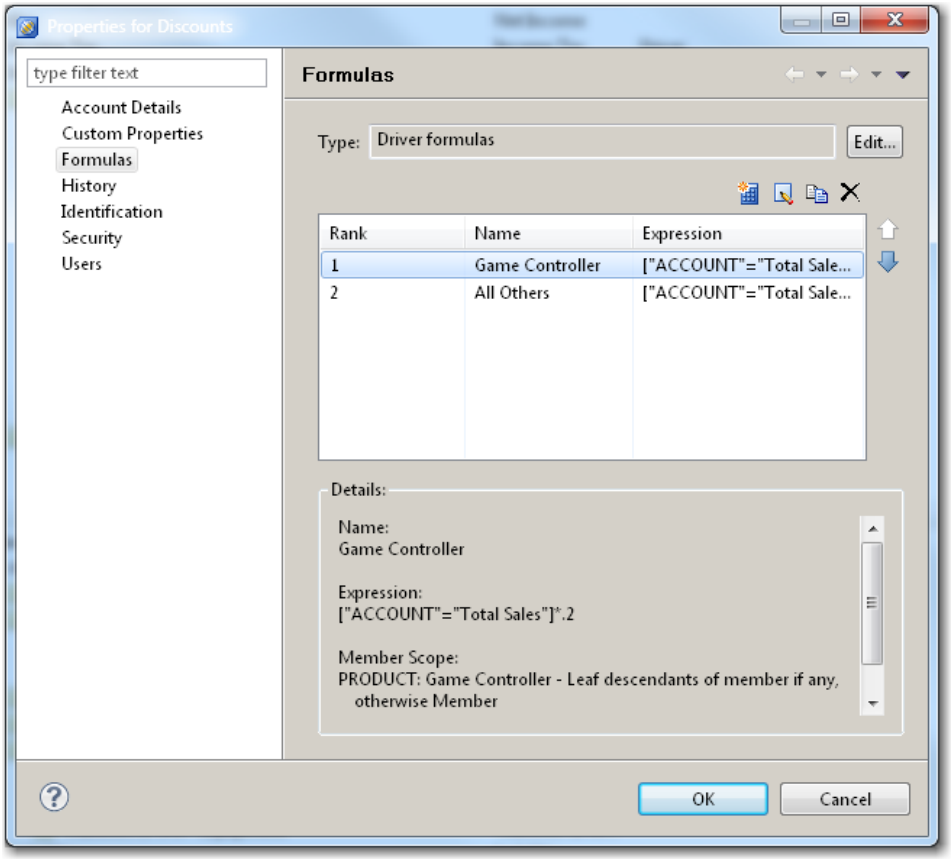
Results

The following table displays the results when the All Others formula is ranked first, rendering the computed Discounts values of 25% for all Product members:

Organization Worldwide Operations
Customer Customer Total
Analysis Budget
Currency USD
Frequency PA
Time Q2 2013

	Hardware	Game Controller	Joy Stick	Flight Stick
Sales	(22,675.00)	(6,750.00)	(11,725.00)	(4,200.00)
Discounts	(5,668.75)	(1,687.50)	(2,931.25)	(1,050.00)
% Discount	0.25	0.25	0.25	0.25

When the formulas are rearranged so that the Game Controller formula is ranked first, the warning message is not displayed.



Results

The following table displays the results when the formula for the Game Controller Product member is ranked first. This results in Discounts of 20% for Game Controller and 25% for all other Product members.

Organization	Worldwide Operations
Customer	Customer Total
Analysis	Budget
Currency	USD
Frequency	PA
Time	Q2 2013

	Hardware	Game Controller	Joy Stick	Flight Stick
Sales	(22,675.00)	(6,750.00)	(11,725.00)	(4,200.00)
Discounts	(5,331.25)	(1,350.00)	(2,931.25)	(1,050.00)
% Discount	0.24	0.20	0.25	0.25

CHAPTER 7

Calculated Members Restrictions

Introduction ... 110
Ignoring Formulas on Rollup Members ... 110
- *Description* ... 110
- *Affected Formula Types* ... 110
- *Example* ... 111
- *Driver Formula* ... 111
- *Results* ... 111
- *Warning Message for SAS Financial Management Studio* ... 111

Time Offsets: Formulas That Run Out of Bounds ... 112
- *Description* ... 112
- *Affected Formula Types* ... 112
- *Example* ... 112
- *Results* ... 112
- *Warning Message* ... 113

Balance Account Calculated Members Scoped to Time ... 113
- *Description* ... 113
- *Affected Formula Types* ... 113
- *Example* ... 114
- *Results* ... 115
- *Warning Messages* ... 115
  - *SAS Financial Management Studio Message* ... 115
  - *Excel Message* ... 115

Referencing a Dimension and/or Member Not in the Model ... 115
- *Description* ... 115
- *Affected Formula Types* ... 116
- *Example* ... 116
- *Results* ... 117
- *Warning Messages* ... 117
  - *SAS Financial Management Studio* ... 117
  - *Excel* ... 117

Referencing Only Constants ... 118
- *Description* ... 118
- *Affected Formula Types* ... 118
- *Example* ... 118
- *Results* ... 119
- *Warning Messages* ... 119
  - *SAS Financial Management Studio* ... 119
  - *Excel* ... 119

Circular References ... 120
- *Description* ... 120
- *Affected Formula Types* ... 120
- *Example* ... 120
- *Results* ... 121
- *Warning Message for Excel* ... 121

Dividing by Zero ... 122
- *Description* ... 122
- *Affected Formula Types* ... 122
- *Example* ... 122
- *Results* ... 123
- *Warning Message for Excel* ... 123
- *Recommendation* ... 124

Introduction

This chapter identifies formula expressions that are ignored, invalid, or not rendered in SAS Financial Management. These three formula expression restrictions can be defined as follows:

- **Ignored:** Calculated members in this category are not executed due to the priority of other computations such as aggregation/rollup logic. Warning messaging is provided on the **Formulas** page for **Model Properties**. Since the formula expression is ignored, there is no cell information.
- **Invalid:** Calculated members in this category are not executed at query time because they are incorrect in the context of the model. In most cases, these expressions pass validation in the **Formula Editor**. However, the expressions have warning messages on the **Formulas** page for **Model Properties**. An invalid driver formula is displayed as a gray, non-writeable cell with a value of 0. Modeling and Reporting formulas render as red cells. Warning messaging is provided through **Cell Information** for Modeling and Driver formulas.
- **Not Rendered:** This type of formula restriction applies to the use of constants in Driver and Modeling formulas. Warning messaging is provided on the **Formulas** page for **Model Properties**. For both formula types, the result is a gray, non-writeable cell with a value of 0. No warning messaging is provided through **Cell Information**.

Ignoring Formulas on Rollup Members

Description

Any formula type with an expression *on a roll-up member* in the dimension it was created in is always ignored. The result is the sum of the leaf values.

Affected Formula Types

- Driver
- Modeling
- Reporting

Example

Consider the following Account hierarchy, where account R40000 (Staff Expenses) is a driver formula.

Staff Expenses
- Salaries
- Benefits
- Travel
- Other Selling Expenses

Gross Profit

Driver Formula

Staff Expenses = ["ACCOUNT" = "Gross Profit"]* -0.2

Results

The following table displays the results. Because account (Staff Expenses is a roll-up member, the formula is ignored, and the leaf values are summed.

Organization	Western
Product	Arcade
Customer	Jones Distributing
Analysis	Actual
Currency	USD
Frequency	PTD

	Jan 2012
Gross Profit	(100,000.00)
Sales	(100,000.00)
Staff Expenses	35,500.00
Salaries	30,000.00
Benefits	3,000.00
Travel	1,500.00
Other Selling Expenses	1,000.00

For Jan 2012, the result is computed as 30,000.00 + 3,000.00 + 1,500.00 + 1,000.00 for a result of 35,500.00, instead of the formula for Staff Expenses executing as (100,000.00) * -0.2 to yield 20,000.00.

Warning Message for SAS Financial Management Studio

The following warning message is displayed in SAS Financial Management Studio on the Details pane for Show members, on the Formulas page for Model Properties:

```
Formula is assigned to a non-leaf member.
```

Time Offsets: Formulas That Run Out of Bounds

Description

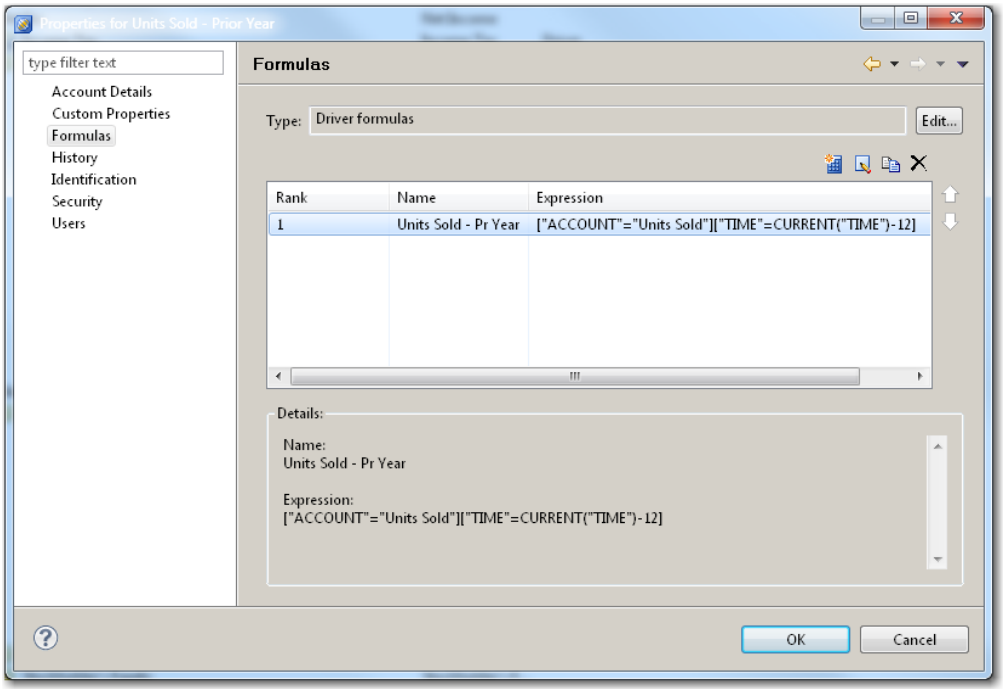
A formula referencing a time offset such as ["TIME"=CURRENT("TIME")-*x*] results in crossings where the formula runs that are "out of bounds." This happens because the formula is seeking inputs from crossings that do not exist. Crossings where formulas run out of bounds are deemed not applicable to the formula and return a value of 0.

Affected Formula Types

- Driver
- Modeling
- Reporting
- Excel-based calculated member

Example

In the following example, Units Sold - Prior Year is a Driver formula defined as follows:



Results

The following table displays the results for Account members Units Sold and Units Sold - Prior Year. In this cycle, Jan 2012 is the first-time member in the Time hierarchy. Based on the formula expression, results for time members from Jan 2012 through Dec

2012 return a 0. This is because their formula inputs (for example, Jan 2011 and Feb 2011) do not exist and are "out of bounds."

Organization Mexico Sales
Analysis Budget
Currency MXN
Frequency PA

	Jan 2012	Feb 2012	Mar 2012	Q1 2012	Jan 2013	Feb 2013	Mar 2013	Q1 2013
Units Sold	50.00	75.00	100.00	225.00	125.00	150.00	175.00	450.00
Units Sold - Prior Year	0.00	0.00	0.00	0.00	50.00	75.00	100.00	225.00

Warning Message

No warning is provided.

Balance Account Calculated Members Scoped to Time

Description

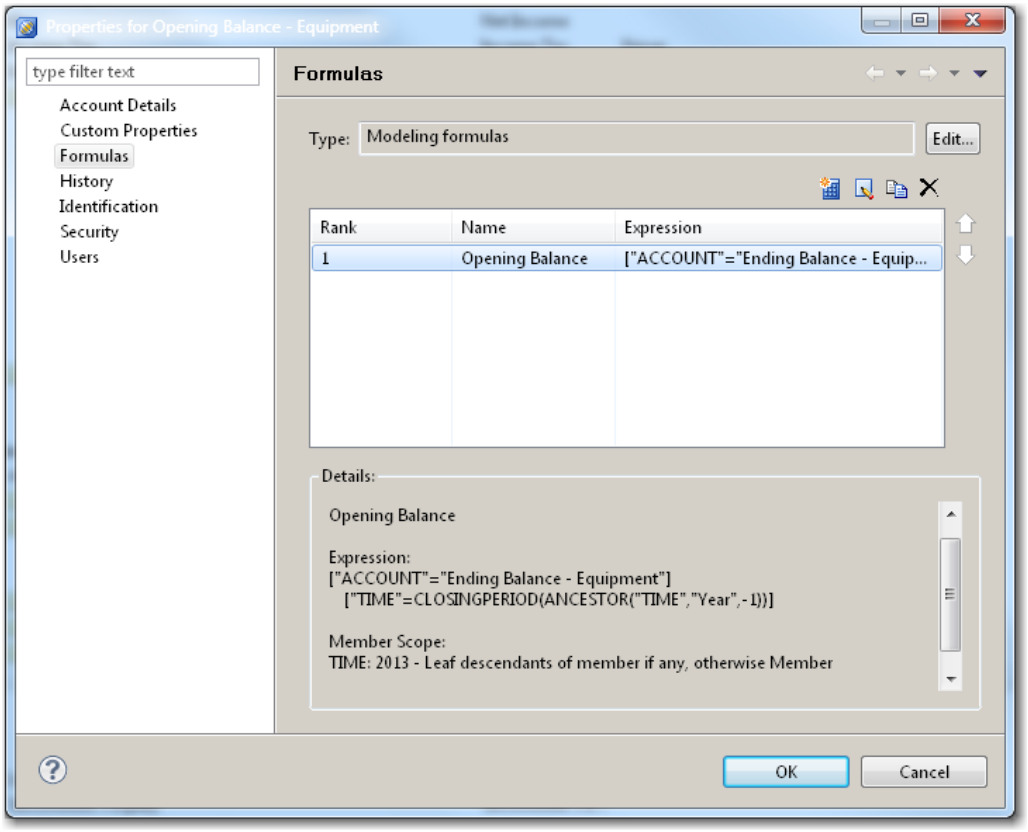
A calculated member on a balance account type scoped on the Time dimension is considered invalid. If the calculated member is assigned as a Driver, the result is 0 for the time periods to which it is scoped and in any future time periods. If the calculated member is assigned as a Modeling formula, the result is a red cell (NaN) for the time periods to which it is scoped and any future time periods.

Affected Formula Types

- Driver
- Modeling

Example

In the following example, Account member Beginning Balance - Equipment is a Modeling formula that takes the Actual value from account member A26000 (Ending Balance, Equipment) for the prior year's ending period as a starting point for budgeting purposes:



Results

The following table displays the results of scoping a Modeling formula on Time for a Balance account type. As displayed on the **Formulas** tab, the Opening Balance - Equipment Account member is scoped to all leaf descendants of the Time member 2013. As a result, any Time periods in the formula scope as well as future time members return red cells (NaNs).

Organization ... European Operations
Analysis ... Budget
Currency ... EUR
Frequency ... PTD

	Dec 2012	Jan 2013	Feb 2013	Mar 2013	Q1 2013
Ending Balance - Equipment	450,000.00				
Opening Balance - Equipment	450,000.00				
Additions - Equipment	0.00	0.00	0.00	0.00	0.00
Disposals - Equipment	0.00	0.00	0.00	0.00	0.00
Transfers & WO - Equipment	0.00	0.00	0.00	0.00	0.00

Warning Messages

SAS Financial Management Studio Message

The following warning message is displayed in SAS Financial Management Studio on the **Details** pane for Show members on the Formulas page for Model Properties**:**

```
A modeling or driver formula on a balance account cannot be scoped for
the time dimension type.
```

Excel Message

For a Modeling formula, the following cell information is provided:

```
Query failed due to a formula error.
```

Referencing a Dimension and/or Member Not in the Model

Description

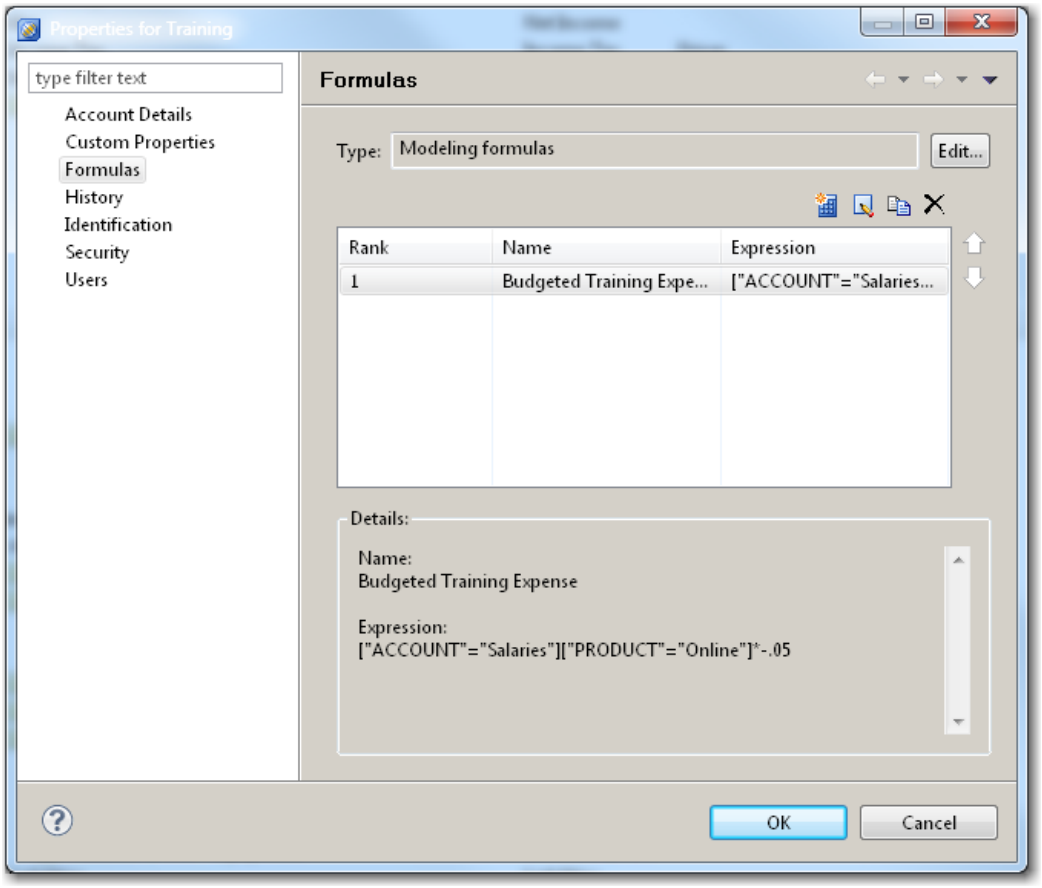
A calculated member that references a dimension and member that is not in the model is invalid. A calculated member that references a member that is not in the hierarchy is also invalid. If assigned as a Driver, the result is 0. If the calculated member is assigned as a Modeling or Reporting formula, the result is a red cell (NaN).

Affected Formula Types

- Driver
- Modeling
- Reporting

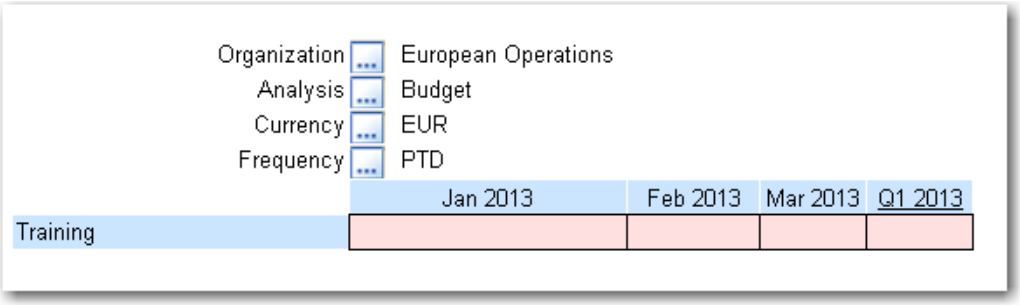
Example

In the following example, Account member Training is a Modeling formula that references a Product member that is no longer in the hierarchy.



Results

The following table displays the results of a Modeling formula referencing a member that is not in the hierarchy.



Warning Messages

SAS Financial Management Studio

The following warning message is displayed in SAS Financial Management Studio on the **Details** pane for Show members on the Formulas page for Model Properties:

```
Formula references a member (Online) that is not in the model.
```

Excel

For a Modeling or Reporting formula, the following cell information is provided:

```
Query failed due to a formula error.
```

Referencing Only Constants

Description

Calculated members referencing only constants in the formula expression are invalid. Examples of formula expressions not supported are displayed below:

X = 1
Y =DRATE("")
Z= DRATE("") +/- 1

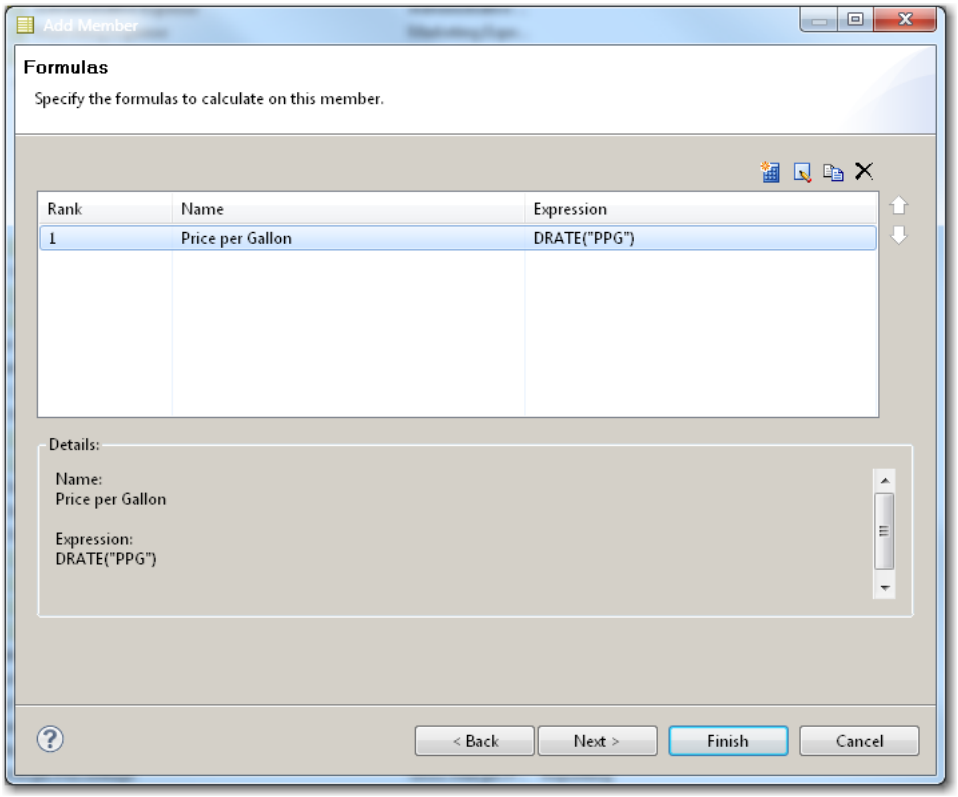
If assigned as either a Driver or Modeling formula, the displayed result is a gray, non-writeable cell with a value of 0.

Affected Formula Types

- Driver
- Modeling

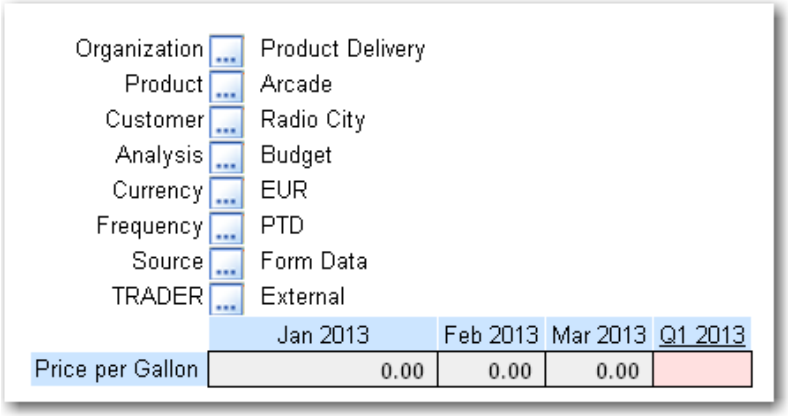
Example

In the following example, Account member Price per Gallon is a Modeling formula that references a constant.



Results

The following table displays the results of a Modeling formula referencing a constant.



Organization	Product Delivery
Product	Arcade
Customer	Radio City
Analysis	Budget
Currency	EUR
Frequency	PTD
Source	Form Data
TRADER	External

	Jan 2013	Feb 2013	Mar 2013	Q1 2013
Price per Gallon	0.00	0.00	0.00	

Warning Messages

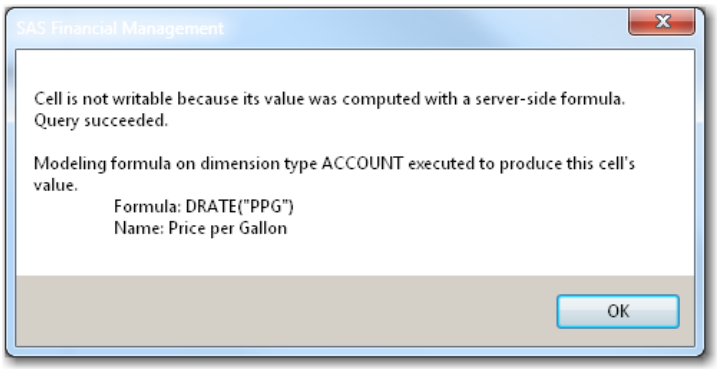
SAS Financial Management Studio

The following warning message is displayed in SAS Financial Management Studio on the **Details** pane for Show members on the Formulas page for Model Properties:

```
Constant expressions are not supported for modeling or driver formulas.
```

Excel

For a Modeling or Driver formula, the following cell information is provided:



Circular References

Description

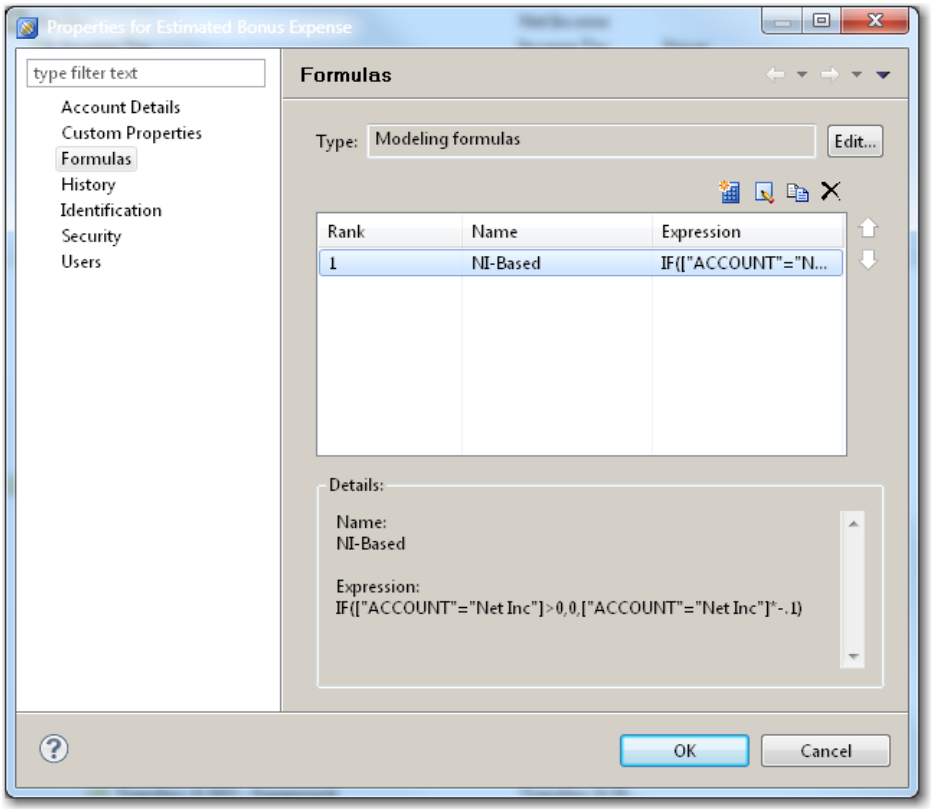
Calculated members with formula expressions containing circular references are invalid.

Affected Formula Types

- Driver
- Modeling
- Reporting

Example

In the following example, Estimated Bonus Expense is calculated based on Net Income. In the existing hierarchy, the Bonus account expression is a circular reference since Estimated Bonus Expense is a descendant of Net Income.



Results

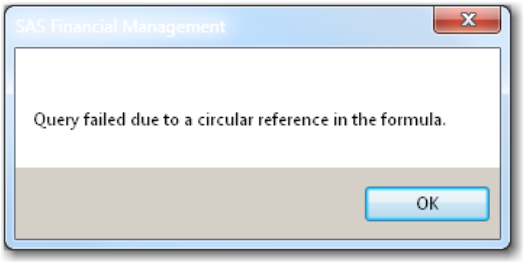
The following table displays the results of the circular reference in the Estimated Bonus Expense account member. The circular reference affects all account members that it rolls up to.

Organization	Canada Operations
Analysis	Budget
Currency	CAD
Frequency	PTD

	Jan 2013
Net Income	
Income Tax	0.00
Income before Taxes	
Operating Expense	
Administrative Expense	0.00
Marketing Expense	0.00
Selling Expense	
Staff Expenses	
Salaries	100,000.00
Estimated Bonus Expense	
Benefits	30,000.00
Travel	10,000.00
Other Selling Expenses	5,000.00
Other Operating Expense	0.00
Gross Profit	0.00

Warning Message for Excel

For a Modeling or Reporting formula, the following cell information is provided:



Dividing by Zero

Description

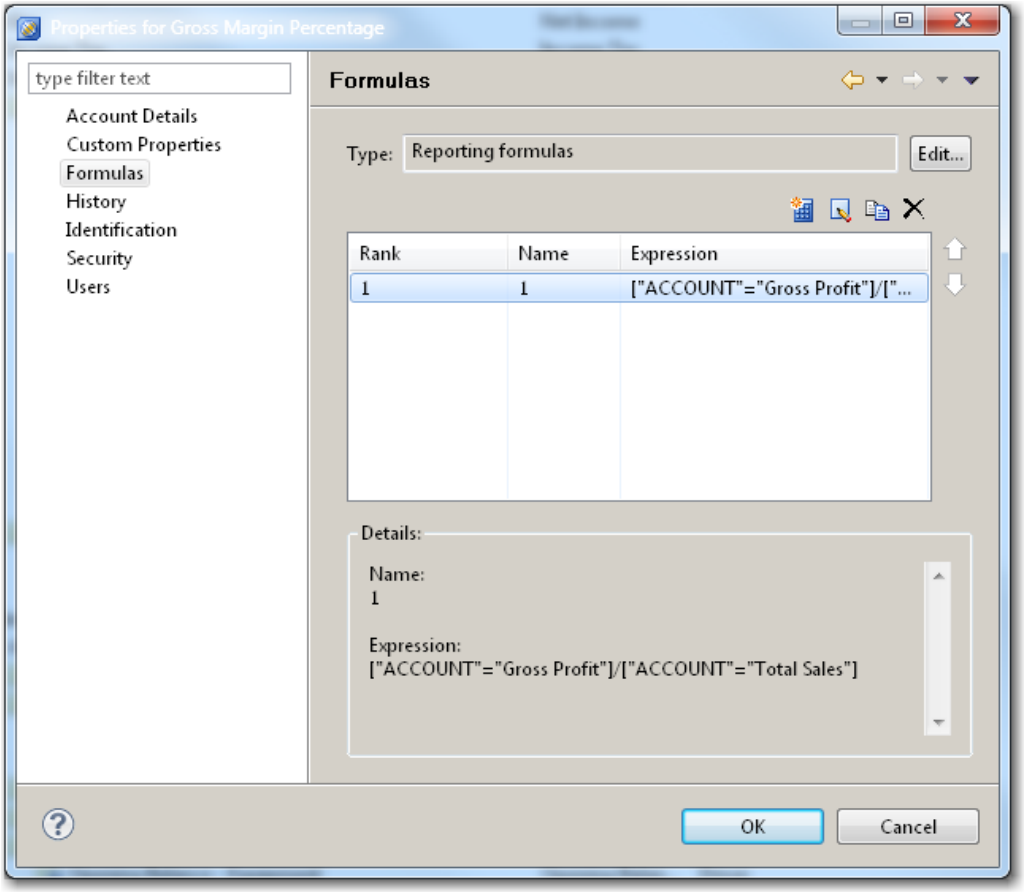
A calculated member where the divisor returns a value of 0 is invalid.

Affected Formula Types

- Driver
- Modeling
- Reporting

Example

In the following example, Gross Margin Percentage is calculated by dividing Gross Profit by Total Sales.



Results

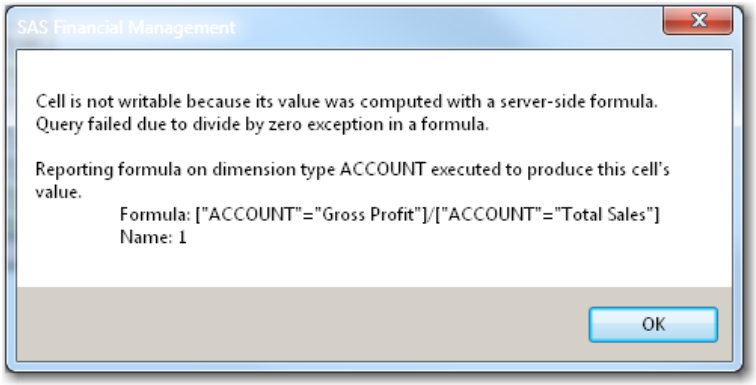
In the following table, Sales results have not been entered yet. Therefore, the formula currently is being divided by 0 pending Sales account data input.

Organization	European Sales
Customer	Accents
Product	Arcade Secrets
Analysis	Budget
Currency	EUR
Frequency	PTD

	Jan 2012
Net Income	82,500.00
Income Tax	0.00
Income before Taxes	82,500.00
Operating Expense	82,500.00
Gross Profit	0.00
Sales	0.00
Cost of Sales	0.00
Gross Margin Percentage	

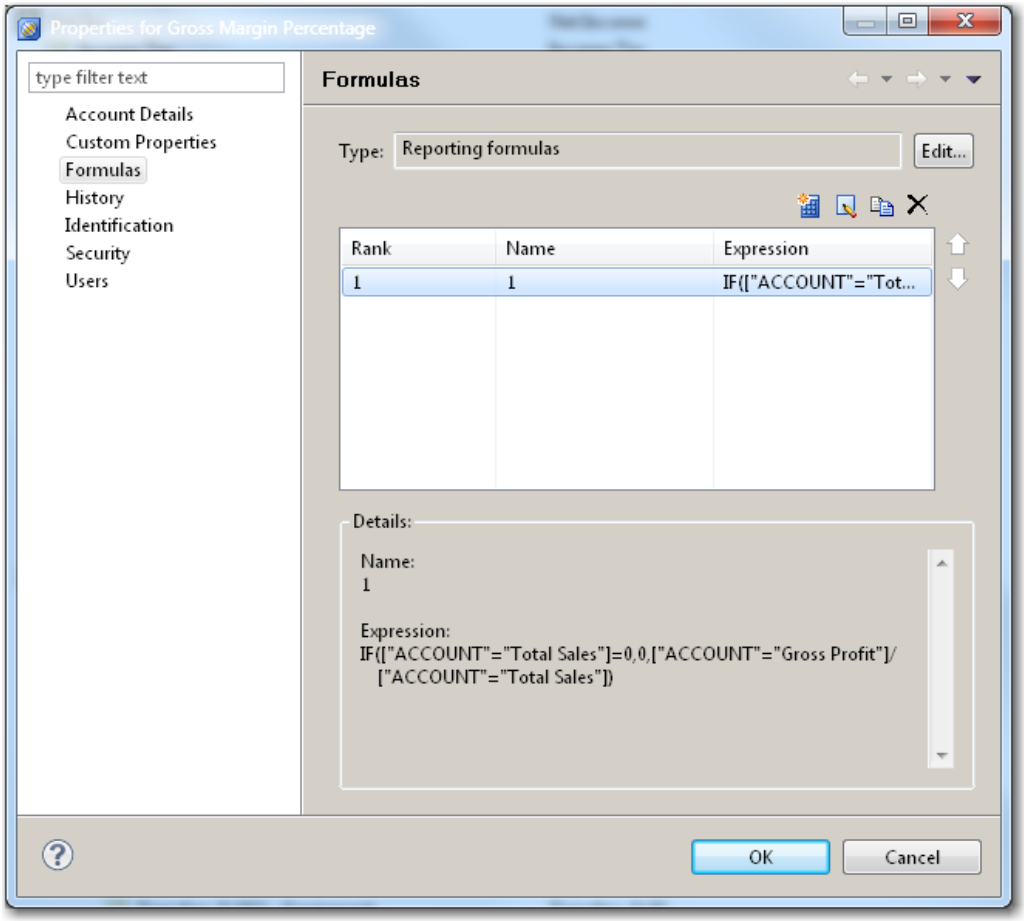
Warning Message for Excel

For a Modeling or Reporting formula, the following cell information is provided:



Recommendation

To prevent this type of invalid calculated member, the formula expression can be re-written in the following manner:



By including an argument in the event that the divisor is 0, the formula expression is no longer considered invalid.

The resulting table is as follows:

Organization	European Sales
Customer	Accents
Product	Arcade Secrets
Analysis	Budget
Currency	EUR
Frequency	PTD

	Jan 2012
Net Income	82,500.00
Income Tax	0.00
Income before Taxes	82,500.00
Operating Expense	82,500.00
Gross Profit	0.00
Sales	0.00
Cost of Sales	0.00
Gross Margin Percentage	0.00

CHAPTER 8

Statistical Account Behavior

Introduction *127*
Statistical Accounts for Data Entry *128*
Introduction *128*
Facts *128*
Hierarchical Roll-Ups *128*
Time Aggregation *129*
Frequency Computation *130*
Data Entry *131*
Calculated Statistical Accounts *131*
Introduction *131*
Formula *131*
Facts *132*
Hierarchical Roll-Ups *132*
Frequency Computation *133*
Restrictions *133*

Introduction

SAS Financial Management has three statistical account types:

- Statistical Flow. This account type resembles the Expense and Revenue account types. It can participate in currency conversion, and it can be a source account to Retained Earnings.
- Statistical Balance. The Statistical Balance account type resembles the Asset and Liability account types. However, it cannot be a source account to a Cumulative Translation Adjustment account.
- Statistical. A Statistical account type cannot roll up to a parent member or have any children. The purpose of the Statistical account type is to represent values such as prices and ratios. Statistical account behavior differs based on whether it is a non-calculated member or a calculated member with a reporting formula.

This chapter addresses the behavior of the Statistical account type. Statistical accounts can be used for data input, and they can be assigned as calculated members. The following sections discuss Statistical account behavior for these two uses separately and describe how this account types differs from Statistical Flow and Statistical Balance account types.

Statistical Accounts for Data Entry

Introduction

Statistical accounts data entry are unlike accounts of other types in that they do not participate in hierarchical roll-ups, time aggregation, or frequency aggregation. Given the intent of the Statistical account type, the values generated from hierarchical roll-ups, time aggregation, or frequency aggregation are nonsensical. To illustrate, consider the following example:

Facts

The facts for the Price account are shown here:

Account	Price
Analysis	Budget
Currency	USD
Customer	State & Local
Frequency	PTD
Time	Jan 2013
Source	Form Data
TRADER	External

	Game Controller	Joy Stick	Flight Stick
Central	15.00	16.00	17.00
Eastern	20.00	18.00	19.00
Western	10.00	9.00	8.00

Hierarchical Roll-Ups

If the table is expanded to include roll-up members in dimensions such as Product and Organization, then the results at the roll-up members for the Price account result in red cells (NaN: Not a Number).

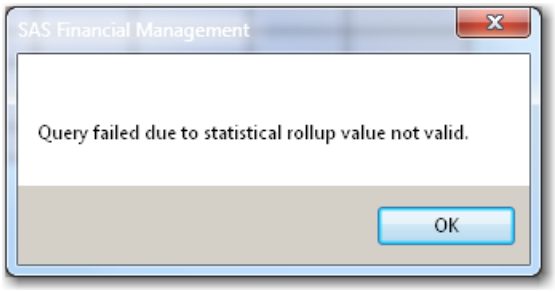
Account	Price
Analysis	Budget
Currency	USD
Customer	State & Local
Frequency	PTD
Time	Jan 2013
Source	Form Data
TRADER	External

	Hardware	Game Controller	Joy Stick	Flight Stick
Sales				
Central		15.00	16.00	17.00
Eastern		20.00	18.00	19.00
Western		10.00	9.00	8.00

The Account “Price” returns a value when it is crossed with leaf members in all other dimensions, as shown in the above example. Product members “Game Controller”, “Joy Stick”, and “Flight Stick” are leaf members. Organization members “Central”, “Eastern”, and “Western” are leaf members. Selected Slicer members also are leaf members.

The Account “Price” returns red cells (NaN) when crossed with a roll-up member in *any* dimension. Product member “Hardware” and Organization member “Sales” are both roll-up members. Therefore, they always return a red cell when crossed with the Account “Price”.

The cell information for a crossing at a roll-up point is displayed as follows:



Time Aggregation

Time aggregation results are similar to hierarchical roll-up results in that the Statistical account value results in a red cell. The cell information that is displayed at a time roll-up is the same as what is shown in “Hierarchical Roll-Ups.”

Account	Price
Analysis	Budget
Currency	USD
Customer	State & Local
Frequency	PTD
Organization	Central
Source	Form Data
TRADER	External

	Jan 2013	Feb 2013	Mar 2013	Q1 2013
Hardware				
Game Controller	15.00	15.20	15.25	
Joy Stick	16.00	15.80	16.00	
Flight Stick	17.00	17.10	17.05	

Frequency Computation

Statistical account values do not participate in frequency computation. Regardless of the frequency selected for a given Statistical account, the resulting value is the same. The following table displays the results of the Price account with the valid frequencies:

Account	Price
Analysis	Budget
Currency	USD
Customer	State & Local
Organization	Central
Product	Game Controller
Source	Form Data
TRADER	External

	Jan 2013	Feb 2013	Mar 2013	Q1 2013
PA	15.00	15.20	15.25	
MA	15.00	15.20	15.25	
QA	15.00	15.20	15.25	
YA	15.00	15.20	15.25	
LA	15.00	15.20	15.25	
PTD	15.00	15.20	15.25	
MTD	15.00	15.20	15.25	
QTD	15.00	15.20	15.25	
YTD	15.00	15.20	15.25	
LTD	15.00	15.20	15.25	

Data Entry

Because non-calculated Statistical accounts do not participate in hierarchical roll-ups, all values must be entered at the leaf level for each dimension in the model, including Source and Trader. The following data-entry table illustrates this requirement:

Organization	Central
Customer	State & Local
Account	Price
Analysis	Budget
Currency	USD
Frequency	PTD
Source	Form Data
TRADER	External

	Jan 2013	Feb 2013	Mar 2013	Q1 2013
Hardware				
Game Controller	15.00	15.20	15.25	
Joy Stick	16.00	15.80	16.00	
Flight Stick	17.00	17.10	17.05	

In the table above, the yellow cells represent crossings in which all dimension members are at a leaf level. The red cells represent crossings where statistical roll-ups are not valid.

Note: If a dimension has a default read member that is a roll-up and a default write member that is a leaf member, the dimension must still be included on the table.

Calculated Statistical Accounts

Introduction

Statistical accounts that are reporting formula-calculated members behave differently from non-calculated Statistical accounts. To illustrate the differences, here is an example of a calculated Statistical account:

Formula

Gross Profit Percentage = ["ACCOUNT"="Gross Profit"]/["ACCOUNT"="Sales"]

Gross Profit Percentage is a Statistical account. Gross Profit and Sales are Revenue accounts.

Facts

The facts for the Sales and Cost of Sales accounts are shown here:

Organization Eastern
Product Action
Analysis Budget
Currency USD
Customer Westco
Frequency PA

	Jan 2013	Feb 2013	Mar 2013	Q1 2013
Gross Profit	(4,000.00)	(6,500.00)	(8,250.00)	(18,750.00)
Sales	(10,000.00)	(15,000.00)	(20,000.00)	(45,000.00)
Cost of Sales	6,000.00	8,500.00	11,750.00	26,250.00

Hierarchical Roll-Ups

If the table is expanded to include the roll-up members in dimensions such as Product, Customer, and Organization, then the results at roll-up members for the Gross Profit Percentage account are displayed. Statistical accounts that are calculated members do display values at roll-up points, as shown here:

Organization Sales
Product Product Total
Analysis Budget
Currency USD
Customer Customer Total
Frequency PA

	Jan 2013	Feb 2013	Mar 2013	Q1 2013
Gross Profit	(4,000.00)	(6,500.00)	(8,250.00)	(18,750.00)
Sales	(10,000.00)	(15,000.00)	(20,000.00)	(45,000.00)
Cost of Sales	6,000.00	8,500.00	11,750.00	26,250.00
Gross Profit Percentage	0.40	0.43	0.41	0.42

Frequency Computation

Statistical accounts that are calculated members also participate in frequency computation. To illustrate, consider the following table:

Organization: Sales
Product: Product Total
Analysis: Budget
Currency: USD
Customer: Customer Total
Account: Gross Profit Percentage

	Jan 2013	Feb 2013	Mar 2013	Q1 2013
PA	0.40	0.43	0.41	0.42
MA	0.40	0.43	0.41	
QA	0.40	0.42	0.42	0.42
YA	0.40	0.42	0.42	0.42
LA	0.47	0.47	0.46	0.46
PTD	0.40	0.43	0.41	0.42
MTD	0.40	0.43	0.41	
QTD	0.40	0.42	0.42	0.42
YTD	0.40	0.42	0.42	0.42
LTD	0.47	0.47	0.46	0.46

Restrictions

Non-calculated Statistical accounts do not participate in hierarchical roll-ups or Time aggregation and return red cells (Not a Number). This means that the calculated Statistical account member returns a red cell at a roll-up when it refers to a non-calculated Statistical account member in its formula expression. However, a calculated Statistical account member returns a value at a roll-up when it refers to another calculated Statistical account member. If you need to use a Statistical account in the formula expression, consider using a Statistical Balance account.

CHAPTER 9

Commonly Used Functions

ANCESTOR Function ... *136*
- *Use* ... *136*
  - *Introduction* ... *136*
  - *First Argument* ... *136*
  - *Second Argument* ... *136*
  - *(Optional) Third Argument* ... *137*
- *Syntax* ... *137*
- *Example* ... *137*

CLOSINGPERIOD Function ... *138*
- *Use* ... *138*
- *Syntax* ... *138*
- *Example* ... *138*

CURRENT Function ... *138*
- *Use* ... *138*
- *Syntax with One Argument* ... *139*
- *Example with One Argument* ... *139*
- *Syntax with Two Arguments* ... *140*
- *CURRENT(<dimension type code>,PROPERTY(<dimension type code>,<property code>))* ... *140*
- *Example with Two Arguments* ... *140*

DRATE Function ... *141*
- *Use* ... *141*
- *Syntax* ... *141*
- *Example* ... *141*

IF Function ... *142*
- *Use* ... *142*
- *Syntax* ... *142*
- *Example* ... *142*

NESTIF Function ... *143*
- *Use* ... *143*
- *Syntax* ... *143*
- *Example* ... *144*

OPENINGPERIOD Function ... *145*
- *Use* ... *145*
- *Syntax* ... *145*
- *Example* ... *145*

PARENT Function ... *146*
- *Use* ... *146*
- *Syntax* ... *146*
- *Example* ... *146*

PREVIOUS Function ... *146*
- *Use* ... *146*
- *Syntax* ... *147*
- *Example* ... *147*

PROPERTY Function ... *149*
- *Use* ... *149*
  - *AccountType* ... *149*
  - *BalanceType* ... *150*
  - *AccountBehavior* ... *150*
  - *ExchangeRateType* ... *150*
  - *FunctionalCurrency* ... *150*
  - *Intercompany* ... *150*

Level ... *150*
ReportingEntity ... *151*
Syntax ... *151*
Example ... *151*
ROUND Function ... *153*
Use ... *153*
Syntax ... *153*
Example with Only the First Argument ... *153*
Example with Both Arguments ... *154*
SUM Function ... *154*
Use ... *154*
Syntax ... *154*
Example ... *154*
VIRTUALCHILD Function ... *157*
Use ... *157*
Syntax ... *157*
Example ... *157*
Adding Comments within a Formula Expression ... *157*
Using String Functions in a Bracketed Member Reference ... *157*
Using Dates and Numbers in Formula Syntax ... *158*
Codes Containing Single and Double Quotation Marks ... *158*

ANCESTOR Function

Use

Introduction

Use the ANCESTOR function to navigate a hierarchy, primarily the TIME hierarchy. This function enables a formula to use a member that is a certain number of levels above the current member or to use a specified period type such as Year.

The ANCESTOR function takes two arguments and supports an optional third argument.

First Argument

The first argument can be either a dimension type code or a function that returns a member code:

- If the first argument is a dimension type code, then this code implicitly specifies the member of that dimension type that is in the crossing where the function is evaluated.
- If the first argument is a function that returns a member code, then the returned code explicitly specifies a member.

Second Argument

The second argument is an ancestor designator. The ancestor designator can take two forms: Number of Levels or Level Name:

- Number of Levels: An integer indicates a number of hierarchical levels above the member that is specified by the first argument. You can use an integer as the second argument no matter what the first argument is.

- Level Name: If the first argument is the Time dimension type code or a function that returns a Time period code, then you can use one of the following period type values to indicate a particular level of the Time hierarchy:

```
AllYears
Year
HalfYear
QuarterYear
Month
Week
Day
```

(Optional) Third Argument

The optional third argument specifies how many time periods in the past or future. This argument is typically used to get a prior year value, as shown in the example later in this section.

Syntax

ANCESTOR("dimension type code", number_of_levels)

Returns the member code of the ancestor a given number of levels above the current member for a dimension.

ANCESTOR("dimension type code","level name")

Returns the member code of the ancestor at a named level (like “Year”) above the current member for a dimension.

Example

Beginning Balance - Equipment =
["ACCOUNT"="Ending Balance - Equipment"]["TIME"=ANCESTOR("TIME","Year",-1)]

In this formula, the ANCESTOR function is used on the Beginning Balance - Equipment account to return the prior year value of the Ending Balance - Equipment account.

Analysis	Budget
Frequency	PTD
Organization	Worldwide Operations

	2012	Jan 2013	Feb 2013
Ending Balance - Equipment	3.00	8.00	6.00
Beginning Balance - Equipment	0.00	3.00	3.00
Additions - Equipment	3.00	4.00	5.00
Disposals - Equipment	0.00	(1.00)	(3.00)
Transfers & WO - Equipment	0.00	2.00	1.00

CLOSINGPERIOD Function

Use

This function returns the code of the member that satisfies these two conditions:

- It belongs to the dimension type that is specified in the function.
- It is the last-listed leaf member that is hierarchically subordinate to the member of the specified dimension type that is in the crossing where the function is evaluated.

If the member of the specified dimension type that is in the crossing where the function is evaluated is a leaf member, then the CLOSINGPERIOD function returns the code of that member. In other words, the CLOSINGPERIOD function returns the same value as the CURRENT function in this case.

The CLOSINGPERIOD function takes one argument, a dimension type code. It is primarily used with the Time dimension type.

Syntax

CLOSINGPERIOD(<dimensionTypeCode>)

Example

In this formula, the CLOSINGPERIOD function works with the ANCESTOR to return the prior year's value for Dec 2012, which is the last leaf period of the prior year.

Sales =
["ACCOUNT"="Sales"]["TIME"=CLOSINGPERIOD(ANCESTOR("TIME","Year",-1))]

Analysis	Budget				
Frequency	PTD				
Organization	Worldwide Operations				
	Nov2012	Dec2012	2012	Jan 2013	Feb 2013
Sales	2.00	3.00	5.00	3.00	3.00

CURRENT Function

Use

The CURRENT function returns the code of the member that satisfies these two conditions:

- It belongs to the dimension type that is specified in the function.
- It is in the crossing where the function is evaluated.

The CURRENT function takes one argument, a dimension type code. It is most often used as part of a relative reference in a Time hierarchy. The CURRENT function also supports an optional second argument which is an offset for a property code type of integer.

Syntax with One Argument

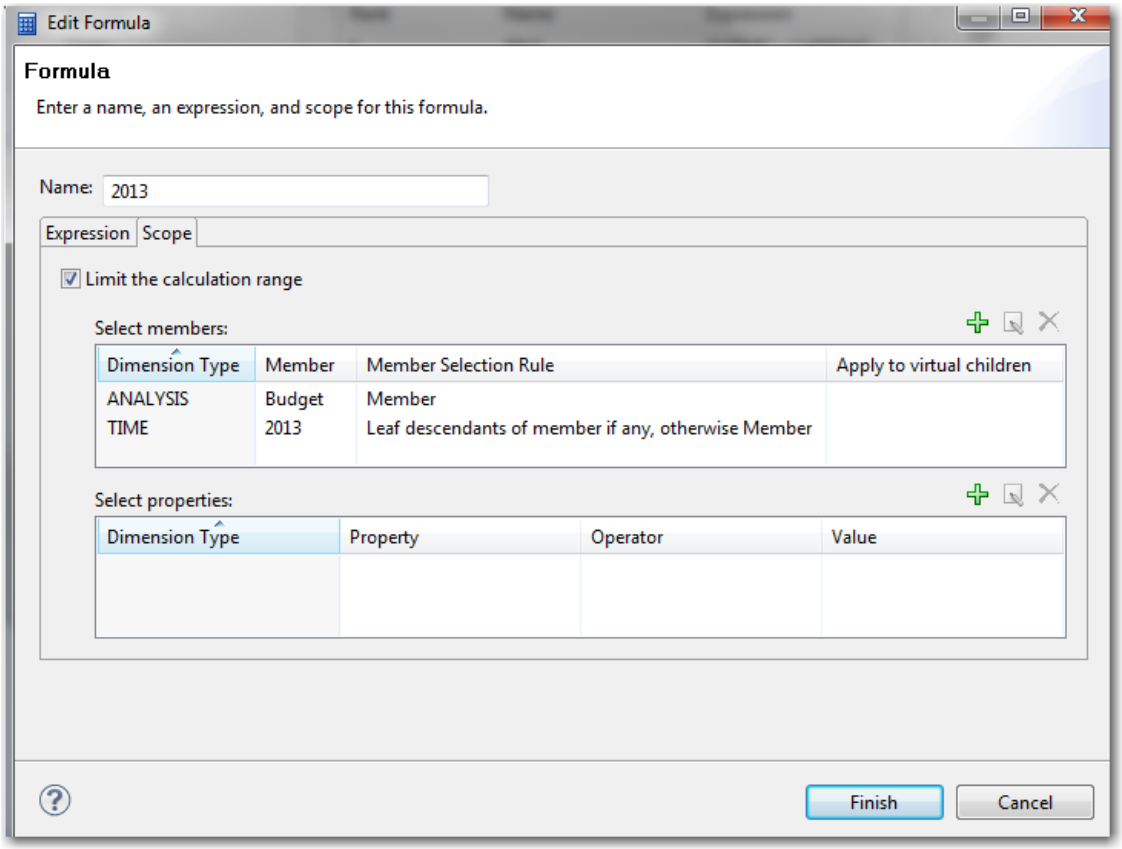
CURRENT(<dimensionTypeCode>)

Example with One Argument

Units Sold =

(["TIME"=CURRENT("TIME")-12]["ANALYSIS"="Actual"]["ACCOUNT"="Units Sold"])*1.1

The scoping is as follows.



In this example, the CURRENT function is used on the Units Sold account. It subtracts a period of 12 months to retrieve the prior year's Actual Units Sold. Then, the value is multiplied by 1.1 to derive the budgeted values for 2013.

Frequency	PTD			
Organization	Worldwide Operations			
	Actual		Budget	
	Jan 2012	Feb 2012	Jan 2013	Feb 2013
Units Sold	8.00	9.00	8.80	9.90

Syntax with Two Arguments

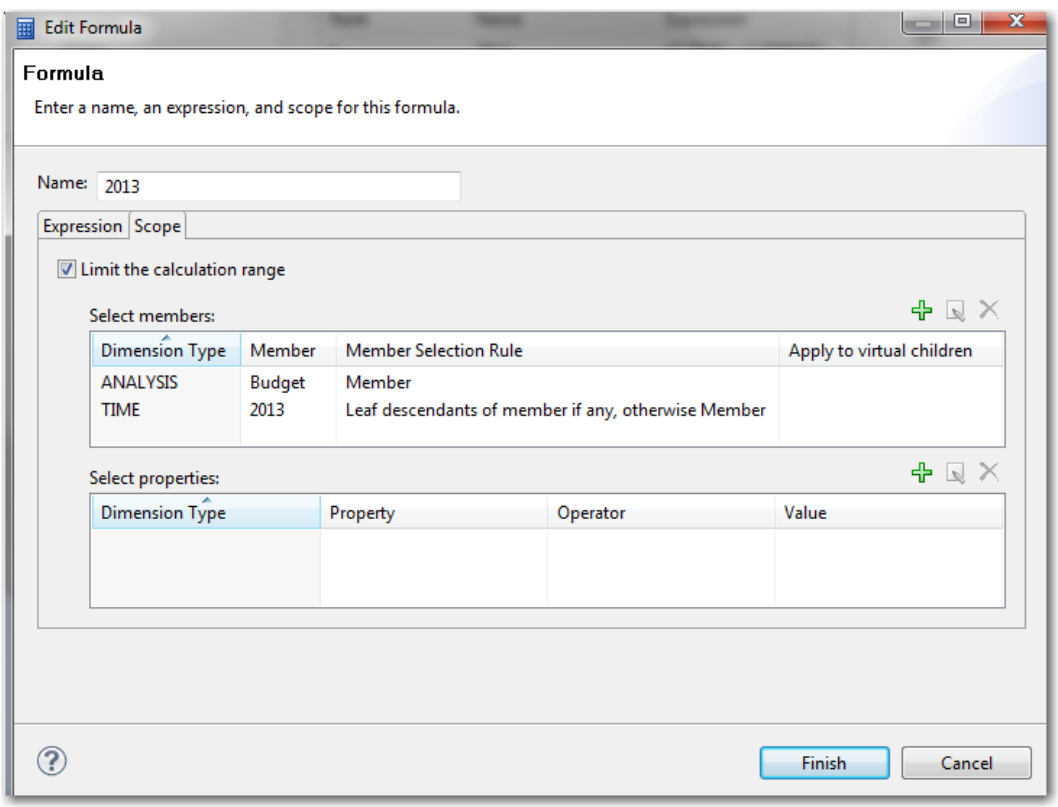
CURRENT(<dimension type code>,PROPERTY(<dimension type code>,<property code>))

Example with Two Arguments

Units Sold =

(["TIME"=CURRENT("TIME", PROPERTY("TIME","TimeOffset"))]["ANALYSIS"="Actual"]["ACCOUNT"="Units Sold"])*1.1

The scoping is the same as the previous example.



In this example, the CURRENT function is used on the Units Sold account. It subtracts a dynamic offset using the custom property "TimeOffset" to retrieve the Actual Units Sold from a previous Time period. Then, the value is multiplied by 1.1 to derive the budgeted values for 2013. Jan 2013 retrieves the value from 12 periods in the past, Feb 2013 retrieves the value from 11 periods in the past, and Mar 2013 retrieves the value from 12 periods in the past. The value of the TimeOffset custom property is displayed on the columns in the following table. The Custom Property Type must be Integer to be used with the CURRENT function.

Frequency	PTD						
Organization	USD						
	Actual			Budget			
	Jan 2012	Feb 2012	Mar 2012	Jan 2013	Feb 2013	Mar 2013	
	None	None	None	-12	-11	-12	(TimeOffset Custom Property)
Units Sold	8.00	9.00	10.00	8.80	11.00	11.00	

DRATE Function

Use

Use the DRATE function to retrieve the numeric values from the **Driver Rate Sets** tab in the Rates workspace.

A driver rate set consists of tables that are specific to driver rate types. Each table contains a "Rate" column of numeric values and several columns that represent dimension types. Each table row associates the numeric value that it contains with a specific dimension member or set of members.

The driver rate set that the DRATE function references is determined by these two things:

- the Analysis member for which the function is evaluated
- the driver rate set that is associated with that Analysis member in the model properties

Syntax

DRATE("driver rate type code")

Example

Income Tax =
IF(["ACCOUNT"="Income Before Taxes"]<0,["ACCOUNT"="Income Before Taxes"]*
DRATE("TaxRate2013")*-1,0)

In this formula, the DRATE function works on the Income Tax account member. The desired outcome is to multiply Income before Taxes by a predefined rate that varies by Organization to return an estimated Income Tax value. If Income Before Taxes is less than zero, then the Income Tax account member will be zero.

The DRATE for TaxRate2013 is displayed as the last row on this report by using an Excel calculated member with the following syntax:

=fmRate("TaxRate2013")

Analysis	Budget		
Frequency	PTD		
Time	Jan 2013		
	Central	Eastern	Western
Net Inc	(12.60)	5.00	(18.30)
Income Tax	7.40	0.00	11.70
Income before Taxes	(20.00)	5.00	(30.00)
TaxRate2013	0.37	0.38	0.39

IF Function

Use

The IF function returns one value if the condition specified evaluates to TRUE and another value if it evaluates to FALSE. Use IF to conduct conditional tests.

Syntax

IF(<condition>,<trueExpression>,<falseExpression>)

Example

Profit Margin % =

IF(["ACCOUNT"="Net Sales"]=0,0,["ACCOUNT"="Net Income"]/["ACCOUNT"="Net Sales"])

In this formula, the IF statement displays a zero when the denominator is zero. Otherwise, the Profit Margin percentage is calculated by dividing Net Income by Net Sales.

Analysis	Actual		
Time	Dec 2013		
	United States	Sales	Administration
Net Income	(34.00)	(39.00)	12.00
Net Sales	(144.00)	(130.00)	0.00
Profit Margin %	0.24	0.30	0.00

NESTIF Function

Use

The NESTIF function returns a value that depends on the truth values of one or more Boolean expressions. The NESTIF function takes an even number of arguments, which are arranged in pairs. The second member of each pair is an expression whose value might be returned. These are the even-numbered arguments of the function. The first member of each pair is a Boolean expression that is associated with the second member of the pair. These are the odd-numbered arguments of the function.

A Boolean expression can compare two character values or two numeric values. Within Boolean expressions, you can use any Boolean operators and comparison operators that are available on the Formula Editor window's symbol toolbar.

The function returns the first even-numbered argument that is associated with a true Boolean argument.

Syntax

NESTIF(<condition1>,<trueExpression1>,< condition2>,<trueExpression2>, ...)

Example

Bonus Expenses =

NESTIF(

["TIME"=ANCESTOR("TIME","Year")]["ACCOUNT"="Sales"]<=1.1*["TIME"=ANCESTOR("TIME","Year",-1)]["ANALYSIS"="Actual"]["ACCOUNT"="Sales"],

1.1*["ACCOUNT"="Bonus Expenses"]["TIME"=ANCESTOR("TIME","Year",
-1)]["ANALYSIS"="Actual"]/12,

(["TIME"=ANCESTOR("TIME","Year")]["ACCOUNT"="Sales"]>1.1*["TIME"=ANCESTOR("TIME","Year",
-1)]["ANALYSIS"="Actual"]["ACCOUNT"="Sales"]) AND (
["TIME"=ANCESTOR("TIME","Year")]["ACCOUNT"="Sales"] <=
["TIME"=ANCESTOR("TIME","Year",
-1)]["ANALYSIS"="Actual"]["ACCOUNT"="Sales"]),

["ACCOUNT"="Bonus Expenses"]["TIME"=ANCESTOR("TIME","Year",
-1)]["ANALYSIS"="Actual"]/12,

["TIME"=ANCESTOR("TIME","Year")]["ACCOUNT"="Sales"]>["TIME"=ANCESTOR("TIME","Year",
-1)]["ANALYSIS"="Actual"]["ACCOUNT"="Sales"],

0

)

This NESTIF statement works with the ANCESTOR function. The desired outcome is to calculate Sales based upon the % change from last year's Actuals. If Sales >=110% of last year, Bonus Expenses are 110% of last year. If Sales are at least equal to but less than 110% of last year, Bonus Expenses are the same as last year. If Sales are less than last year, then Bonus Expenses are zero. Note that Sales carries a credit balance so the greater than and less than signs work in conjunction with that. This is a modeling formula scoped to Budget and Forecast Analysis members and also scoped to leaf descendants of the year 2013.

Organization	Central		
Currency	USD		
Frequency	PTD		
	Actual	Budget	Forecast
	2012	2013	2013
Bonus Expenses	12.00	13.20	12.00
Sales	(1,200.00)	(1,320.00)	(1,205.00)

OPENINGPERIOD Function

Use

The OPENINGPERIOD function returns the code of the member that satisfies these two conditions:

- It belongs to the dimension type that is specified in the function.
- It is the first-listed leaf member that is hierarchically subordinate to the member of the specified dimension type that is in the crossing where the function is evaluated. It is primarily used with the Time dimension type.

Syntax

OPENINGPERIOD(<dimensionTypeCode>)

Example

Avg Receivables Balance =

SUM(["ACCOUNT"="Accounts & Notes Receivable"] ["TIME"=OPENINGPERIOD(ANCESTOR("TIME","Year"))]:

["ACCOUNT"="Accounts & Notes Receivable"] ["TIME"=CURRENT("TIME")])

/(PROPERTY("TIME","Month_number"))

In this example, the OPENINGPERIOD function works with the SUM function and the PROPERTY function to calculate the year-to-date average balance of the Accounts and Notes Receivable account.

The PROPERTY function retrieves the values of a custom property (Month_number) of the Time dimension. The Month_number custom property values are displayed in the report as the second column heading, beneath the TIME members.

Analysis	Actual		
Frequency	PTD		
Organization	Worldwide Operations		
	Jan 2012	Feb 2012	Mar 2012
	1	2	3
Accounts & Notes Receivable	15.00	20.00	22.00
Avg Receivables Balance	15.00	17.50	19.00

PARENT Function

Use

This function returns the code of the member that satisfies both of these conditions:

- It belongs to the dimension type that is specified in the function.
- It is the hierarchical parent of the member of the specified dimension type that is in the crossing where the function is evaluated.

The PARENT function takes one argument, a dimension type code.

Syntax

PARENT(<dimensionTypeCode>)

Example

Net Sales =

["ACCOUNT"="Units Sold"]* -1 * ["ACCOUNT"="Price"]["PRODUCT"=VIRTUALCHILD(PARENT(CURRENT ("PRODUCT")))]

In this example, the PARENT function works with the VIRTUALCHILD and CURRENT functions. The desired outcome is to multiply Units Sold for each product by the Price that is entered at the virtual child of the current parent in the product hierarchy.

Analysis	Forecast
Currency	USD
TRADER	EXT
Source	Form Data
Customer	HAL
Frequency	PTD
Organization	Central
Time	Jan 2013

	Hardware	Hardware (vc)	Game Controller	Joy Stick	Flight Stick
Net Sales	(45.00)	0.00	(10.00)	(15.00)	(20.00)
Units Sold	9.00	0.00	2.00	3.00	4.00
Price		5.00	0.00	0.00	0.00

PREVIOUS Function

Use

The PREVIOUS function returns the code of the member that satisfies these two conditions:

- It belongs to the dimension type that is specified in the function.
- It is at the same hierarchical level, and immediately previous to, the member of the specified dimension type that is in the crossing where the function is evaluated.

The PREVIOUS function takes one argument: a dimension type code. It is most often used as part of a relative reference in a Time hierarchy.

Syntax

PREVIOUS(<dimensionTypeCode>)

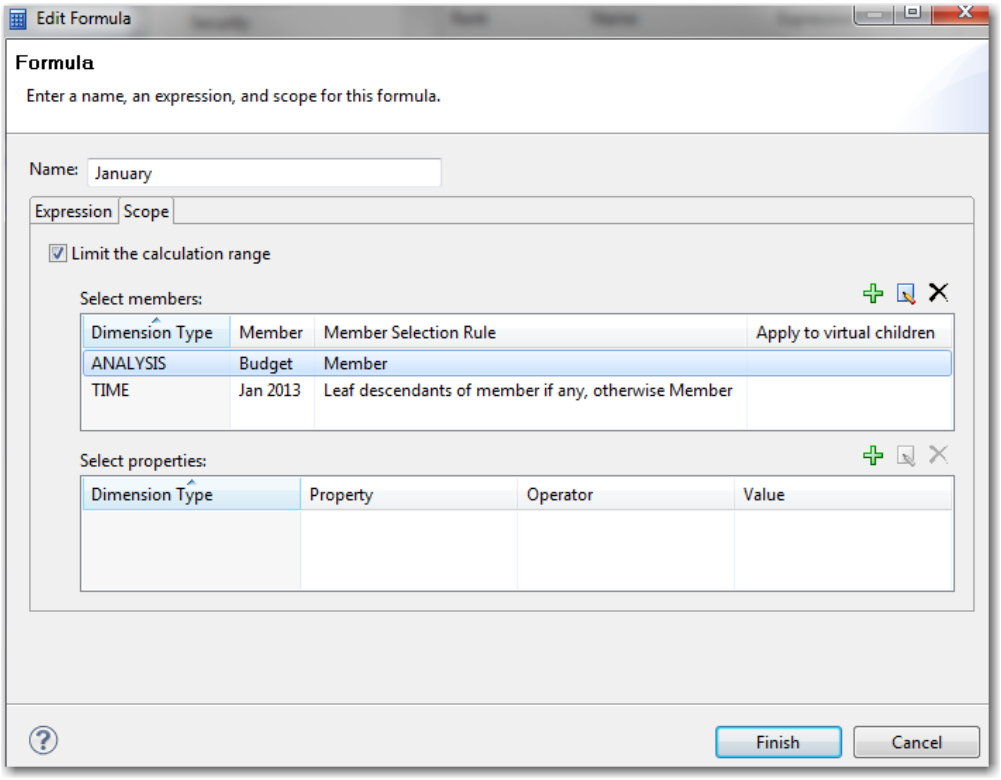
Example

The expression ranked first is named “January”, and the syntax is as follows.

Previous Period Salaries =

["ACCOUNT"="Total Salaries"]["TIME"=PREVIOUS("TIME")]["ANALYSIS"="Actual"]

The scoping for “January” is as follows.

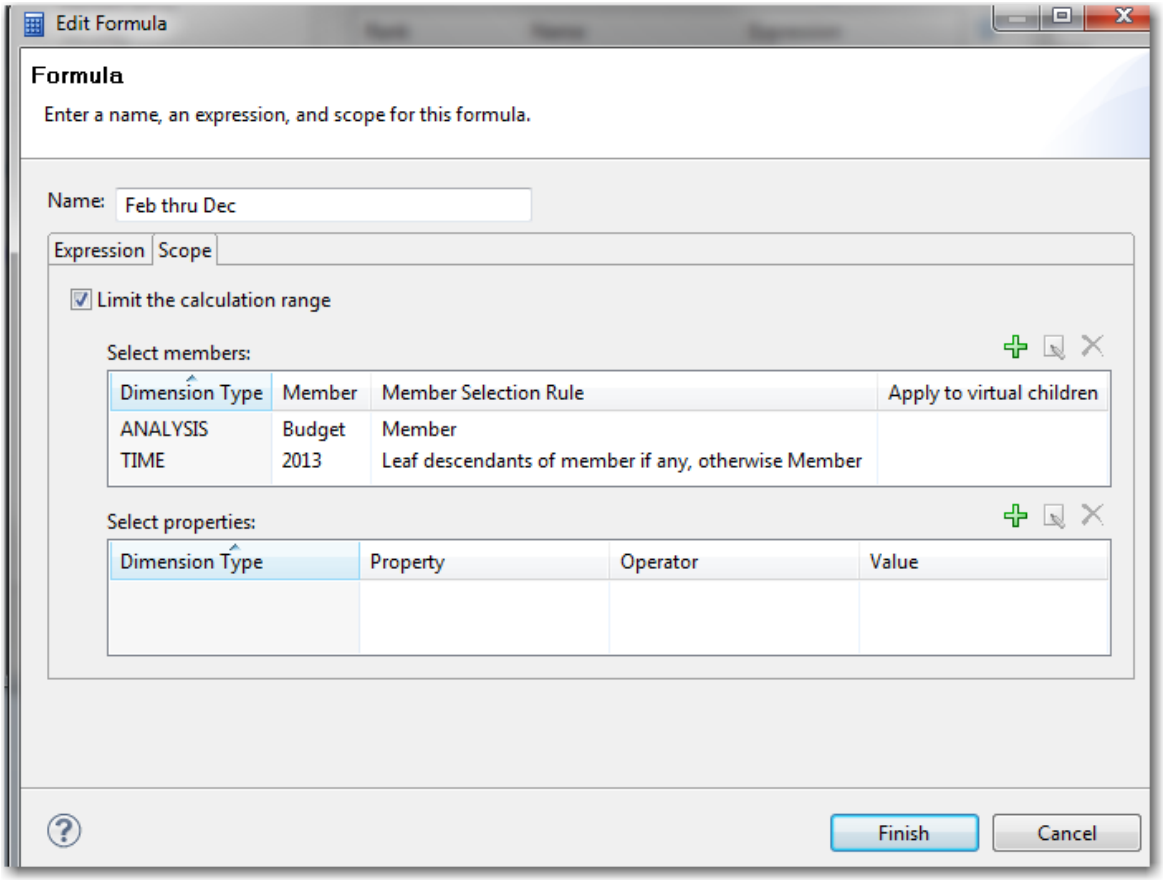


The expression ranked second is named “Feb thru Dec”.

Previous Period Salaries =

["ACCOUNT"="Total Salaries"]["TIME"=PREVIOUS("TIME")]

The scoping for “Feb thru Dec” is as follows.



In this example, the PREVIOUS function works with the Previous Period Salaries account to retrieve the previous period's value from the Total Salaries account. In Jan 2013, the previous period's value is retrieved from the Actual Analysis member. In all subsequent time periods, the previous period's value is retrieved from the Budget Analysis member.

Frequency	PTD			
Organization	Worldwide Operations			
	Actual	Budget		
	Dec 2012	Dec 2012	Jan 2013	Feb 2013
Total Salaries	100.00	80.00	105.00	115.00
Salaries	100.00	80.00	0.00	0.00
Previous Period Salari	0.00	0.00	100.00	105.00
Adj to Salaries	0.00	0.00	5.00	10.00

PROPERTY Function

Use

The PROPERTY function returns the value of a specified property that belongs to the dimension member that is specified in the function. The PROPERTY function returns a string value. In a comparison, it must be compared against a quoted string.

This is the list of standard property codes that can be used by this function and the values that the function can return for each of them. Note that these property codes are case sensitive when used by the PROPERTY function. You can use the PROPERTY function to retrieve the values of custom member properties, as illustrated in the OPENINGPERIOD example.

AccountType

The account type of an account. This property is valid only if the member is an account. The following values can be returned:

- Asset
- Liability
- Equity
- Revenue
- Expense
- RetainedEarnings
- CTA
- StatisticalBalance
- StatisticalFlow
- NonFrequency

Note: NonFrequency is another name for the Statistical account type.

BalanceType

The balance type of an account. This property is valid only if the member is an account. The following values can be returned:

- Credit
- Debit

AccountBehavior

The category to which the account type belongs. This property is valid only if the member is an account. The following values can be returned:

- Balance
- CTA
- Flow
- Hybrid
- NonFrequency

Note: NonFrequency is another name for the Statistical account type. Hybrid is another name for the RetainedEarnings account type.

ExchangeRateType

The exchange rate type of an account. This property is valid only if the member is an account. The following values can be returned:

- PeriodAverage
- PeriodClose
- PeriodOpen
- Custom1
- Custom2
- Derived
- Historic
- None

FunctionalCurrency

The functional currency of an organization. This property is valid only if the member is an organization.

Intercompany

Indicates whether an account is an intercompany account. This property is valid only if the member is an account. The following values can be returned:

- True
- False

Level

The period type of a time period. This property is valid only if the member is a time period. The following values can be returned:

- AllYears
- Year

- HalfYear
- QuarterYear
- Month
- Week
- Day

ReportingEntity

Indicates whether an organization is a reporting entity. This property is valid only if the member is an organization. The following values can be returned:

- True
- False

Syntax

PROPERTY(<dimensionTypeCode>,< propertyCode>)

Example

The first expression is named “Assets and Liabilities”, and the syntax is as follows.

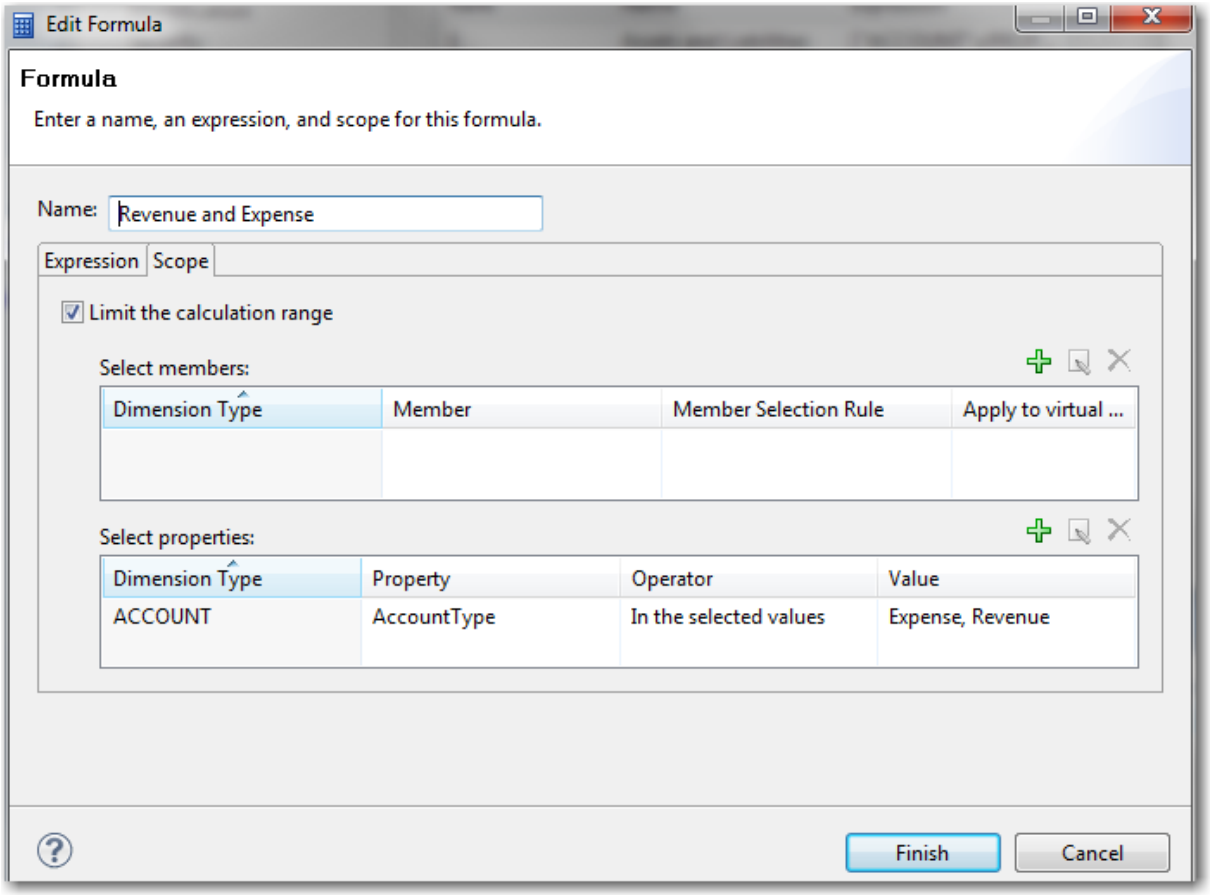
Yield =
["ACCOUNT"=PROPERTY("ACCOUNT","link")]["ANALYSIS"="Actual"]/31*365/
["ANALYSIS"="Actual"]

This formula is scoped to AccountType = Asset and Liability.

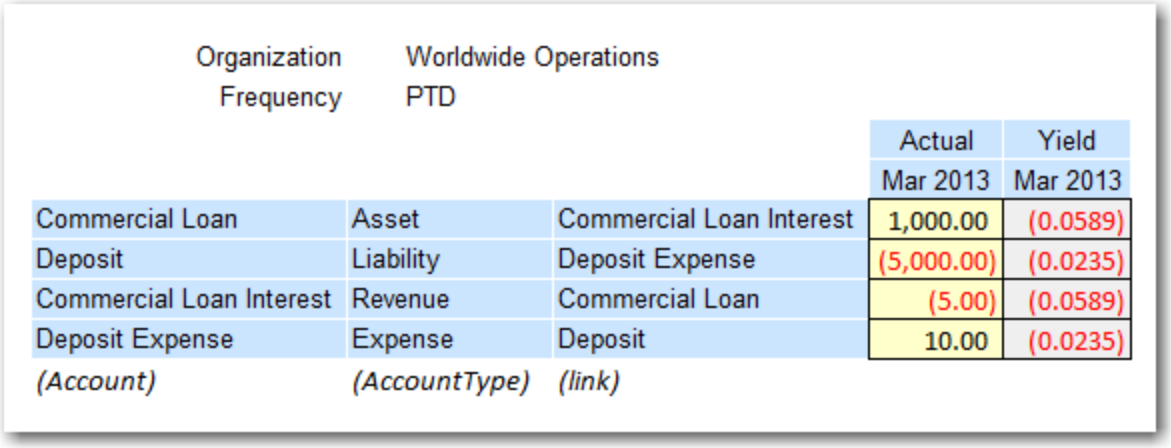
The second expression is named Revenue and Expense, and the syntax is as follows.

Yield =
["ANALYSIS"="Actual"]/31*365/["ANALYSIS"="Actual"]["ACCOUNT"=PROPERTY
("ACCOUNT","link")]

This formula is scoped to AccountType = Revenue and Expense.



In this formula, the PROPERTY function is used to look up the Account which will be used in the calculation of the yield percentage.



Organization Worldwide Operations
Frequency PTD

			Actual	Yield
			Mar 2013	Mar 2013
Commercial Loan	Asset	Commercial Loan Interest	1,000.00	(0.0589)
Deposit	Liability	Deposit Expense	(5,000.00)	(0.0235)
Commercial Loan Interest	Revenue	Commercial Loan	(5.00)	(0.0589)
Deposit Expense	Expense	Deposit	10.00	(0.0235)
(Account)	*(AccountType)*	*(link)*		

ROUND Function

Use

The ROUND function rounds an argument to a given number of digits. The first argument is required and represents the member or value to be rounded. The second argument is optional (number of digits). The arguments must have a numeric value. If the specified number is exactly midway between two integers and the second argument is not provided, then the larger integer is returned. The number 2.5 is rounded up to 3.0, and the number (2.5) is rounded to (3.0).

Syntax

ROUND(<number>, <number_of_digits>)

Example with Only the First Argument

Revenue per Employee =

ROUND(["ACCOUNT"="Net Sales"]/["ACCOUNT"="Ending Headcount"]*-1)

Time	Dec 2012			
Analysis	Budget			
Currency	USD			
	Sales	Central	Eastern	Western
Net Sales	(16.03)	(8.53)	(7.50)	0.00
Ending Headcount	4.00	4.00	0.00	0.00
Revenue per Employee	4.00	2.00		

When a modeling or reporting formula encounters division by zero, a red cell is displayed (NaN). Cell Information states that the query failed due to divide by zero exception in a formula. (See the Revenue per Employee formula crossing with Eastern and Western.)

Modifying the formula to check for zeros in the denominator results in a display of zero instead. For example, the modified formula expression shown below generates zeros.

IF(["ACCOUNT"="Ending Headcount"] ^= 0 , ROUND(["ACCOUNT"="Net Sales"]/["ACCOUNT"="Ending Headcount"]*-1),0)

Time	Dec 2012			
Analysis	Budget			
Currency	USD			
	Sales	Central	Eastern	Western
Net Sales	(16.03)	(8.53)	(7.50)	0.00
Ending Headcount	4.00	4.00	0.00	0.00
Revenue per Employee	4.00	2.00	0.00	0.00

Example with Both Arguments

Adding the second argument to the ROUND function defines how many digits to round.

Revenue per Employee = IF(["ACCOUNT"="Ending Headcount"] ^= 0, ROUND(["ACCOUNT"="Net Sales"]/ ["ACCOUNT"="Ending Headcount"]*-1,2),0)

Time	Dec 2012			
Analysis	Budget			
Currency	USD			
	Sales	Central	Eastern	Western
Net Sales	(16.03)	(8.53)	(7.50)	0.00
Ending Headcount	4.00	4.00	0.00	0.00
Revenue per Employee	4.01	2.13	0.00	0.00

SUM Function

Use

This function returns the sum of its arguments. The SUM function can take any number of arguments. All the arguments must have numeric values. For example, *SUM(1, 2, 3, 3, 16)* returns *25*.

You can use a colon to specify a range of crossings, as in the following example:

SUM(["TIME"="JAN2013"]:["TIME"="JUN2013"])

For range specifications, the first and last crossings must be at the same level in a hierarchy. The specified range can include only crossings at the same level. Range in TIME using different period types should not be used. An example is a range starting with a period type of Month and ending with a period type of Year.

The same number of dimensions should appear on both sides of the colon.

The member on the left side of the colon must have a location in the hierarchy that is before the member on the right side of the colon. This can be an issue if you use the same formula in different hierarchies.

Syntax

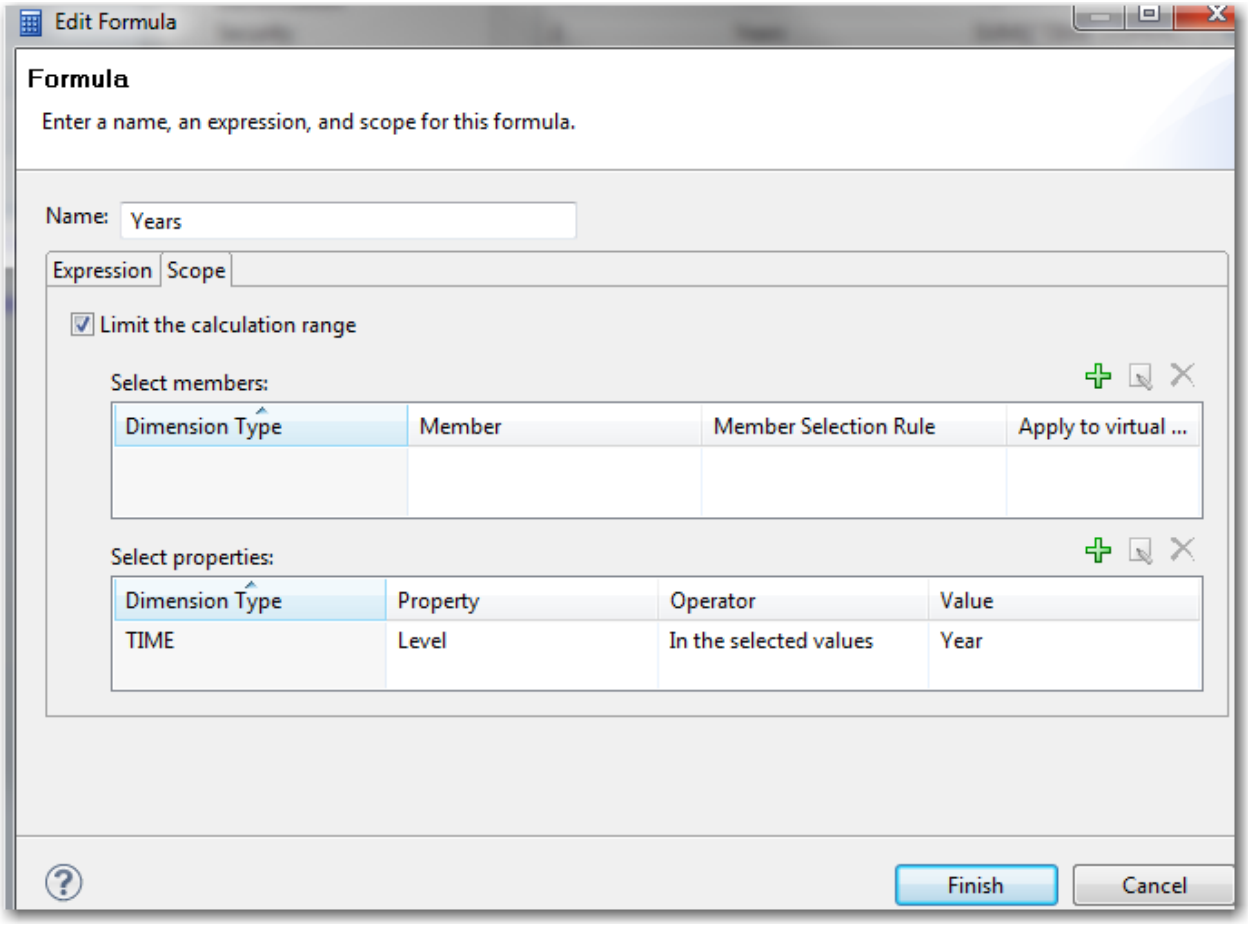
SUM(<set>)

Example

The first expression defines the calculation for the Year level in the TIME hierarchy.

Rolling = SUM(["TIME"=OPENINGPERIOD(ANCESTOR("TIME","Year"))]:["TIME"= CLOSINGPERIOD(ANCESTOR("TIME","Year"))])

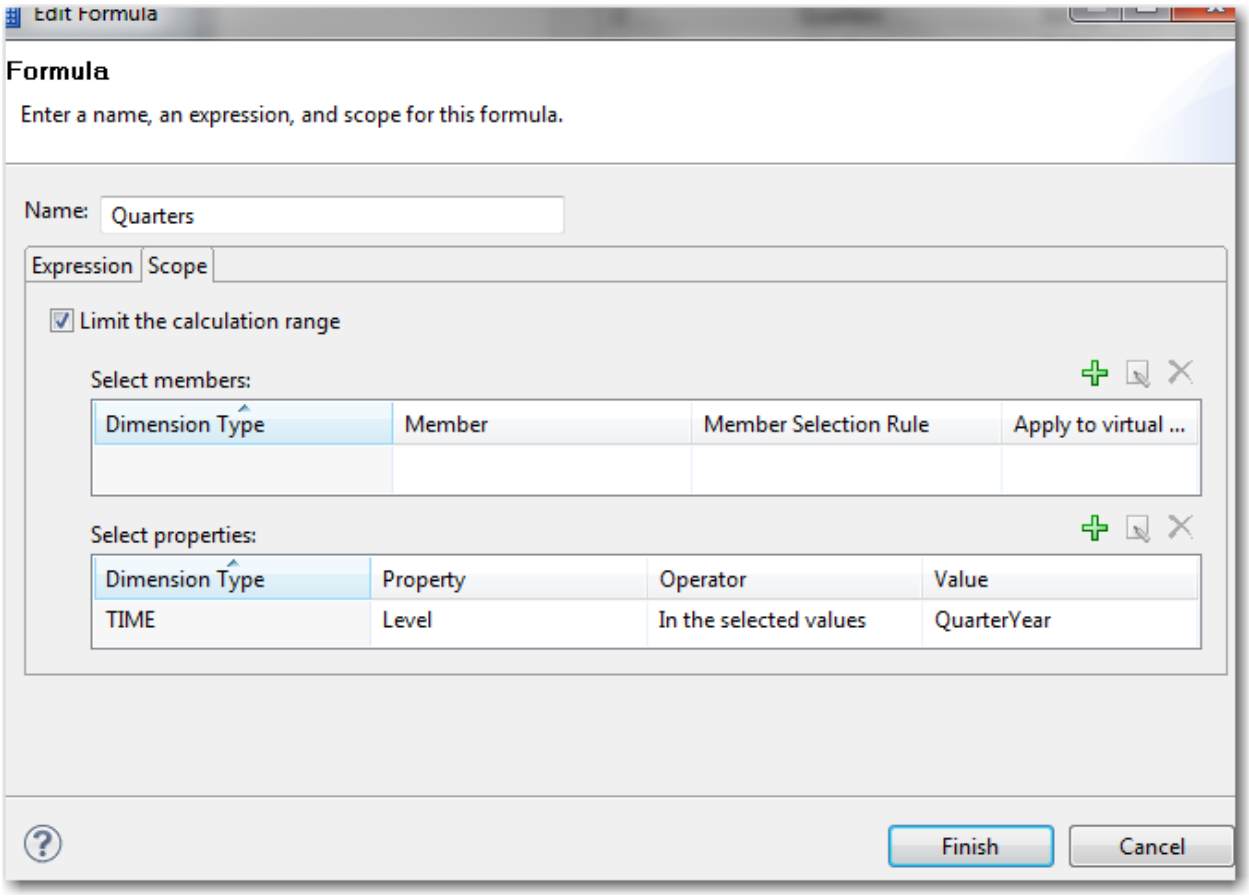
This expression is scoped as follows.



The second expression is scoped to the Quarters and has the following syntax:

SUM(["TIME"=OPENINGPERIOD(ANCESTOR("TIME","QuarterYear"))]:["TIME"=CLOSINGPERIOD(ANCESTOR("TIME","QuarterYear"))])

The scoping is defined as follows.



The third expression is not scoped and uses the following expression.

IF(PROPERTY("TIME","Flag") ="1", ["ANALYSIS"="Actual"], ["ANALYSIS"="Forecast"])

In these formulas, the SUM function is used in combination with the OPENING PERIOD, CLOSING PERIOD and ANCESTOR functions to generate a rolling period value.

	2013	Q1 2013	Jan 2013	Feb 2013	Mar 2013	Q2 2013	Apr 2013	May 2013	Jun 2013	Q3 2013	Q4 2013
	Year	QuarterYear	Month	Month	Month	QuarterYear	Month	Month	Month	QuarterYear	QuarterYear
	None	None	1	1	1	None	1	None	None	None	None
Actual	22.00	15.00	4.00	5.00	6.00	7.00	7.00	0.00	0.00	0.00	0.00
Forecast	30.00	9.00	0.00	4.00	5.00	21.00	6.00	7.00	8.00	0.00	0.00
Rolling	37.00	15.00	4.00	5.00	6.00	22.00	7.00	7.00	8.00	0.00	0.00

VIRTUALCHILD Function

Use

The VIRTUALCHILD returns a reference to the virtual child of a specified member. It takes one argument, which is a reference to the member whose virtual child you want to refer to. The member cannot be a leaf member because leaf members do not have virtual children.

Syntax

VIRTUALCHILD(<memberCode>)

Example

See PARENT.

Adding Comments within a Formula Expression

It is useful to insert comments within a formula expression to document the use and/or intent of the formula expression. To insert a comment, place the following symbols before and after the comment text: **`/*comment text *`**/

For example:

SUM(["ACCOUNT"="Accounts & Notes Receivable"]
["TIME"=OPENINGPERIOD(ANCESTOR("TIME","Year"))]:["ACCOUNT"="Accounts & Notes Receivable"]
["TIME"=CURRENT("TIME")])/(PROPERTY("TIME","Month_number"))
/ calculates the YTD average by adding the first leaf of the year through the current period divided by the number of periods using a custom property to identify the number of periods */

Using String Functions in a Bracketed Member Reference

The following string functions can be used inside a bracketed member reference:

ANCESTOR, CLOSINGPERIOD, COMPRESS, CURRENT, FIRSTCHILD, FIRSTSIBLING, LASTCHILD, LEFT, LOWCASE, NEXT, OPENINGPERIOD, PARENT, PREVIOUS, PROPERTY, REPEAT, REVERSE, RIGHT, SUBSTR, SCAN, TRIM, UPCASE, VIRTUALCHILD

See the OPENINGPERIOD example. It uses the OPENINGPERIOD, ANCESTOR, and CURRENT functions within bracketed TIME member references.

Using Dates and Numbers in Formula Syntax

To insert a date or number, you must use formats that are supported in the en_US locale, regardless of the default data locale. For example, the period in a number such as 1.123 represents a decimal point, not a thousands separator. As another example, DATE(2011/01/01), DATE(20110101), and DATE(2011-01-01) are valid, but DATE(2011.01.02) and DATE(30.12.2011) are not.

Codes Containing Single and Double Quotation Marks

A formula cannot be created referencing a dimension type code or a member code containing both single quotation marks and double quotation marks. For example, a formula cannot reference ["ACCOUNT"="Tolkien's "The Hobbit""] because the member contains both single quotation marks and double quotation marks. An unexpected token error will occur.

If a dimension type code or a member code contains double quotation marks, edit the syntax to replace the double quotation marks around the code with single quotation marks. For example, if you have ["ACCOUNT"="my "own" account"] edit the formula syntax to instead be ["ACCOUNT"='my "own" account'].

CHAPTER 10

Interactions with Load and Delete Data Options

Options for Loading Data ... *159*
Introduction ... *159*
Load new data to this cycle Option ... *160*
Load model data to this cycle Option ... *160*
Loading Data and the Source Dimension ... *160*
Scenario 1. Selecting the Base Source Member ... *161*
Introduction ... *161*
Modeling Formula ... *161*
Formula Scope ... *161*
Facts ... *162*
Results ... *163*
Formula Scope ... *165*
Scenario 2. Selecting the BaseForm Source Member ... *166*
Introduction ... *166*
Formula ... *166*
Formula Scope ... *166*
Facts ... *167*
Results ... *167*
Loading Driver Formulas Facts to the Base Member of the Source Dimension ... *168*
Deleting Data and Driver Facts ... *169*

Options for Loading Data

Introduction

There are two ways to load data in SAS Financial Management in the Periods workspace:

- load new data to this cycle
- load model data to this cycle

Load new data to this cycle Option

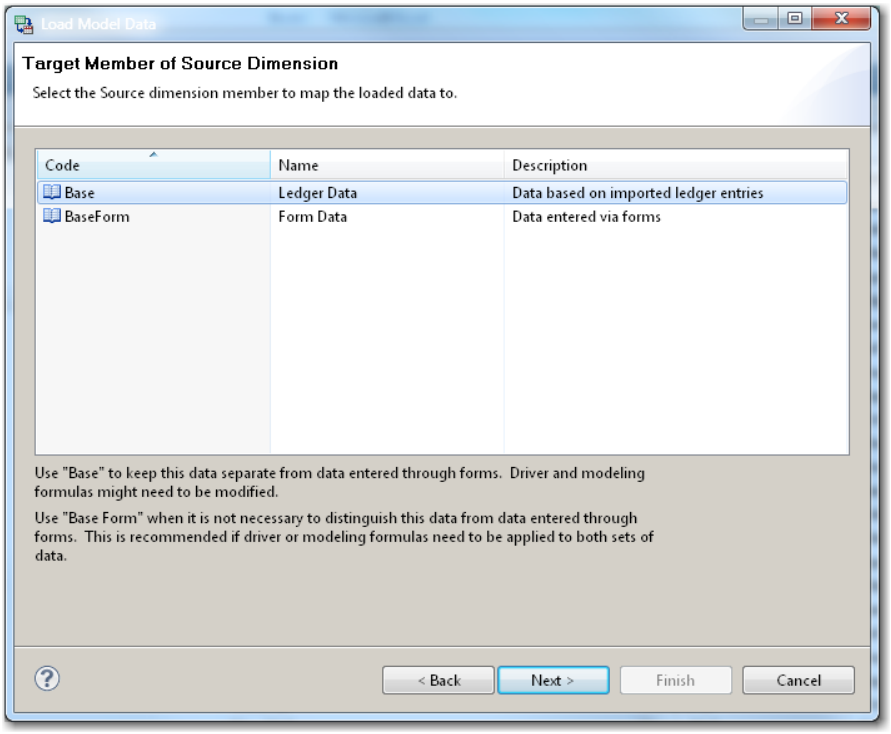
Use this option to load data from the SAS Financial Management Data Mart (SASSDM). In addition to specifying the Time and Analysis dimension, the data load can also be subset to a select group of members for the IntOrg, Trader, Account, and Custom dimensions. Loading can be either to the Base or BaseForm Source member.

Load model data to this cycle Option

This option is used to load data stored in the SASSDM. In addition to specifying the Time and Analysis dimension, the data load can be subset to a select group of members for the IntOrg, Account, and Custom dimensions. Loading can be either to the Base or BaseForm Source member. Member mapping is optional, as is the application of a multiplier to increase or decrease values.

Loading Data and the Source Dimension

Loading new data and model data offers the option to load to the Base or BaseForm Source member. Therefore, it's important to understand the differences that these selections can make in formula results. The following scenarios illustrate the potential formula result variances based on the Source member selection as illustrated in the wizard below:



Scenario 1. Selecting the Base Source Member

Introduction

The Base member of the Source dimension is typically recommended when it is necessary to keep numeric values loaded from the SASSDM separate from numeric values entered through forms. This approach might require driver and modeling formulas to explicitly state specific members of the Source dimension in the formula expression to ensure correct results. To illustrate, we use the following example:

Modeling Formula

Sales = ["ACCOUNT"="Price"]*["ACCOUNT"="Units Sold"]*-1

Formula Scope

["ANALYSIS"="Budget"]

["ANALYSIS"="Forecast"]

Facts

For the Forecast Analysis member, facts from the Jan 2013 through Mar 2013 Budget Analysis member are loaded to the same time periods for the Forecast Analysis member. The data for the Budget Analysis member to be loaded to the Forecast Analysis member is as follows:

Organization	Eastern
Product	Action
Customer	Westco
Analysis	Budget
Currency	USD
Frequency	PTD
Source	Form Data
TRADER	External

	Jan 2013	Feb 2013	Mar 2013	Q1 2013
Price	10.00	10.00	10.00	
Units Sold	100.00	125.00	150.00	375.00
Sales	(1,000.00)	(1,250.00)	(1,500.00)	(3,750.00)

The Budget data is loaded to the Forecast Analysis member, selecting to load to the Base Source member. Following the load, the values for Units Sold are modified via Data Entry to the following values:

Jan 2013: 125
Feb 2013: 150
Mar 2013: 175

The data-entry table for the modified Units Sold values for the Forecast Analysis member is shown below:

Organization	Eastern
Product	Action
Customer	Westco
Analysis	Forecast
Currency	USD
Frequency	PTD
TRADER	External
Source	Total

	Jan 2013	Feb 2013	Mar 2013	Q1 2013
Units Sold	125.00	150.00	175.00	450.00

Results

The expected results for Sales for the Forecast Analysis member are computed as follows:

Jan 2013: 10.00 * 125.00 * -1 = (1,250.00)

Feb 2013: 10.00 * 150.00 * -1 = (1,500.00)

Mar 2013: 10.00 * 175.00 * -1 = (1,750.00)

Q1 2013: (1,250.00) + (1,500.00) + (1,750.00) = (4,500.00)

The read-only table displays the following results:

Organization	Eastern
Product	Action
Customer	Westco
Analysis	Forecast
Currency	USD
Frequency	PTD

	Jan 2013	Feb 2013	Mar 2013	Q1 2013
Price				
Units Sold	125.00	150.00	175.00	450.00
Sales	(1,000.00)	(1,250.00)	(1,500.00)	(3,750.00)

In this example, the Sales account's values do not match the expected results. This is due to the fact that the data was loaded to the Base Source member for the Forecast Analysis dimension. After the data is loaded, modifications to the Units Sold account through the Form are automatically stored in the BaseForm Source member.

The contributing data records for the Units Sold account for the Total Source member for Jan 2013 show two things:

- A value of 100 is in the Base Source member.
- A net value of 25 is in the BaseForm source member.

TIME	SOURCE	PRODUCT	CUSTOMER	OBJECT_TYPE	OBJECT	Author	Date	Value	USD converted
Jan 2013	Base	Action	Westco	dataload	64667	sasdemo	2/21/2012 2:50 PM	100.00	100.00
Jan 2013	BaseForm	Action	Westco	formset	Formula Guide form set	sasdemo	2/21/2012 2:52 PM	(100.00)	(100.00)
Jan 2013	BaseForm	Action	Westco	formset	Formula Guide form set	sasdemo	2/21/2012 2:52 PM	125.00	125.00
								Total:	125.00

Facts entered via a Form are always written to the BaseForm Source member. In this example, the modified value of 125.00 entered for Source Total results in the difference between the existing amount and the modified amount to be written to the BaseForm Source member. Therefore, the value of 125.00 entered for Jan 2013 was netted against the 100.00 value currently stored in the Base member. As shown in contributing data, the 100 is negated in the BaseForm Source member, leaving a net value of 25.00 stored for the BaseForm Source member.

Modeling formulas treat the Source dimension in the same way that they treat all other dimensions. In each calculation, the Source member of all the input crossings is the Source member of the output crossing, unless you override this behavior in the formula expression. A calculation for a Base crossing gets its inputs from Base crossings, and so on. Modifying the table to include the Source dimension more clearly illustrates how the values for the Units Sold Account member were stored and the values for Jan 2013 Sales are calculated:

Organization	Eastern
Product	Action
Customer	Westco
Analysis	Forecast
Currency	USD
Frequency	PTD
Time	Jan 2013
TRADER	All

	Total	TotalBeforeElim	TotalBeforeAdj	BaseForm	TotalAfterImport	Base
Price						
Units Sold	125.00	125.00	125.00	25.00	100.00	100.00
Sales	(1,000.00)	(1,000.00)	(1,000.00)	0.00	(1,000.00)	(1,000.00)

Based on where the data is stored, the results for the Sales account are currently computed as follows:

Jan 2013: (10.00 * 100.00 * -1) + (0.00 * 25.00 * –1) = (1,000.00)

Feb 2013: (10.00 * 125.00 * -1) + (0.00 * 25.00 * –1) = (1,250.00)

Mar 2013: (10.00 * 150.00 * -1) + (0.00 * 25.00 * –1) = (1,500.00)

Q1 2013: (1,000.00) + (1,250.00) + (1,500.00) = (3,750.00)

It is important to consider the target member of the Source dimension for loading data where driver formulas and modeling formulas are involved. In order to achieve the originally expected results of (1,250.00), (1,500.00), (1,750.00) and (4,500.00), the formula expression should be modified as follows:

```
Sales = ["SOURCE"="Base"]["ACCOUNT"="Price"]
*(["SOURCE"="Base"]["ACCOUNT"="Units
Sold"]+["SOURCE"="BaseForm"]["ACCOUNT"=
"Units Sold"])*-1
```

Formula Scope

["SOURCE"="BaseForm"]

The results for Jan 2013 based on the revised formula expression are as follows:

Organization	Eastern
Product	Action
Customer	Westco
Analysis	Forecast
Currency	USD
Frequency	PTD
Time	Jan 2013
TRADER	External

	Total	TotalBeforeElim	TotalBeforeAdj	BaseForm	TotalAfterImport	Base
Price				0.00		10.00
Units Sold	125.00	125.00	125.00	25.00	100.00	100.00
Sales	(1,250.00)	(1,250.00)	(1,250.00)	(1,250.00)	0.00	0.00

Note that with modeling formulas, it might also be necessary to define a formula scope to ensure the desired results. If the formula scope is limited to the BaseForm member of the Source dimension, the resulting table appears as displayed above. Without formula scope on the Source dimension, a modeling formula runs for each Source member that is a leaf member, as displayed below:

Organization	Eastern
Product	Action
Customer	Westco
Analysis	Forecast
Currency	USD
Frequency	PTD
Time	Jan 2013
TRADER	External

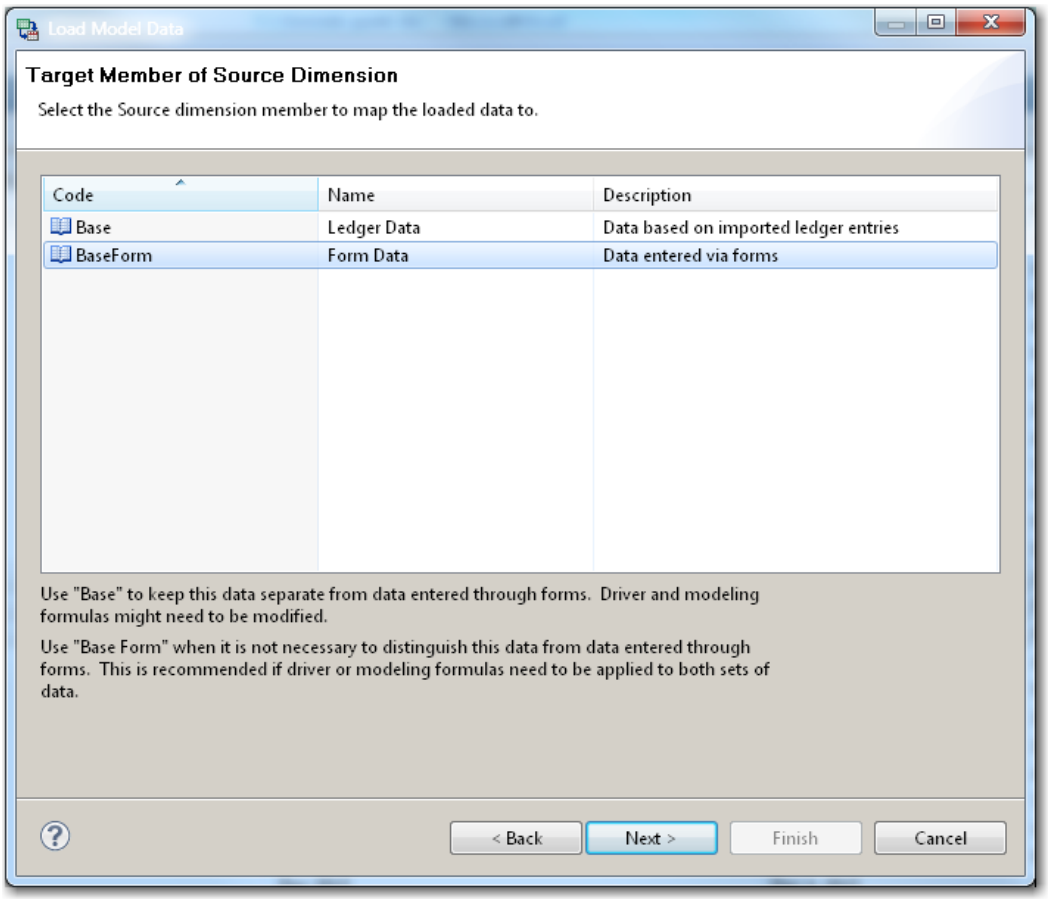
	Total	CTA	Elim	TotalBeforeElim	Adj	TotalBeforeAdj	BaseForm	TotalAfterImport	BaseJourn	Base
Price		0.00	0.00				0.00		0.00	10.00
Units Sold	125.00	0.00	0.00	125.00	0.00	125.00	25.00	100.00	0.00	100.00
Sales	(17,500.00)	(1,250.00)	(1,250.00)	(10,000.00)	(6,250.00)	(3,750.00)	(1,250.00)	(2,500.00)	(1,250.00)	(1,250.00)

As illustrated, selecting the Base Source member as the target for loading data might require additional maintenance for driver formulas and modeling formulas. We recommend this only when you need to keep data that is loaded from an external source or a model separate from data that is entered through Forms.

Scenario 2. Selecting the BaseForm Source Member

Introduction

If you do not have to keep data that is loaded from the SASSDM or a model separate from data entered through a Form, then select the BaseForm member as the target for loading data. The following scenario uses the same formula expression as Scenario 1. Selecting BaseForm as the target member provides the following details:



Formula

Sales = ["ACCOUNT"="Price"]*["ACCOUNT"="Units Sold"]* -1

Formula Scope

["ANALYSIS"="Budget"]

["ANALYSIS"="Forecast"]

Facts

For the Forecast Analysis member, facts from the Jan 2013 through Mar 2013 Budget Analysis member are loaded to the same time periods for the Forecast Analysis member. The data for the Budget Analysis member to be loaded to the Forecast Analysis member is as follows:

Organization	Eastern
Product	Puzzle
Customer	Buy Best
Analysis	Budget
Currency	USD
Frequency	PTD
TRADER	External
Source	Form Data

	Jan 2013	Feb 2013	Mar 2013	Q1 2013
Price	15.00	15.00	15.00	
Units Sold	200.00	300.00	400.00	900.00
Sales	(3,000.00)	(4,500.00)	(6,000.00)	(13,500.00)

After the data load is loaded to the Forecast Analysis member for the BaseForm Source member, the values for Units are modified as follows:

- Jan 2013: 100
- Feb 2013: 200
- Mar 2013: 300

The data-entry table for the modified Units values for the Forecast Analysis member is shown here:

Organization	Eastern
Product	Puzzle
Customer	Buy Best
Analysis	Forecast
Currency	USD
Frequency	PTD
TRADER	External
Source	Form Data

	Jan 2013	Feb 2013	Mar 2013	Q1 2013
Units Sold	100.00	200.00	300.00	600.00

Results

The expected results for Sales are computed as follows:

Jan 2013: 15.00 * 100.00 * -1 = (1,500.00)

Feb 2013: 15.00 * 200.00 * -1 = (3,000.00)

Mar 2013: 15.00 * 300.00 * -1 = (4,500.00)

Q1 2013: (1,500.00) + (3,000.00) + (4,500.00) = (9,000.00)

The read-only table displays the following results:

Organization	Eastern
Product	Puzzle
Customer	Buy Best
Analysis	Forecast
Currency	USD
Frequency	PTD
Source	BaseForm
TRADER	External

	Jan 2013	Feb 2013	Mar 2013	Q1 2013
Price	15.00	15.00	15.00	
Units Sold	100.00	200.00	300.00	600.00
Sales	(1,500.00)	(3,000.00)	(4,500.00)	(9,000.00)

In this example, the values for the Sales account match the expected results. This is because all the facts (whether loaded from another model or entered through a form) are stored in the BaseForm Source member. The contributing data records for the Units Sold account for Jan 2013 are as follows:

TIME	SOURCE	PRODUCT	CUSTOMER	OBJECT_TYPE	OBJECT	Author	Date	Value	USD converted
Jan 2013	BaseForm	Puzzle	Buy Best	dataload	64671	sasdemo	2/21/2012 4:44 PM	200.00	200.00
Jan 2013	BaseForm	Puzzle	Buy Best	formset	Formula Guide form set	sasdemo	2/21/2012 4:44 PM	(200.00)	(200.00)
Jan 2013	BaseForm	Puzzle	Buy Best	formset	Formula Guide form set	sasdemo	2/21/2012 4:44 PM	100.00	100.00
								Total:	100.00

As evidenced in Scenario 2, selecting the BaseForm member as the target for loading data proves to be the recommended approach where modeling formulas and driver formulas are involved. This selection minimizes formula expression maintenance and ensures formula accuracy.

Loading Driver Formulas Facts to the Base Member of the Source Dimension

Another consideration to keep in mind when loading data from models is driver formula facts. Because the results of driver formulas are stored as facts in the database, these facts can be part of the data that is loaded. While this might be completely acceptable, consider the following series of questions before you load data from a model:

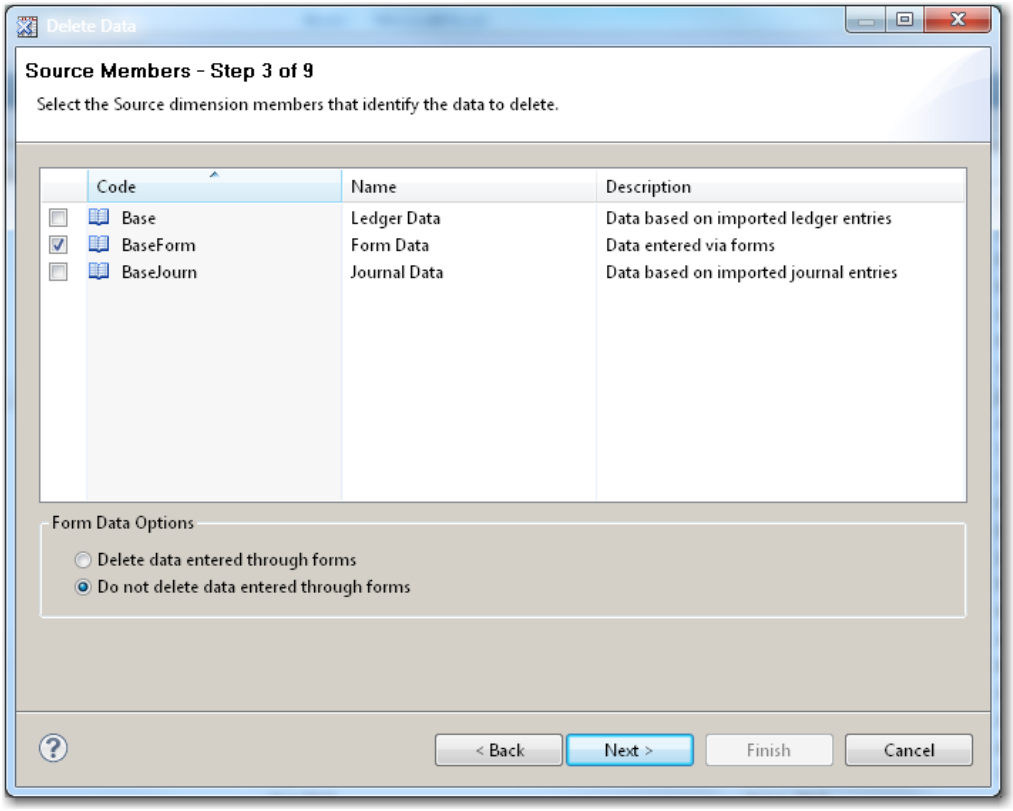
1. Does the model contain driver formulas? If it does, continue to the next question. If not, proceed to load the data.
2. Will the data be loaded to the Base member or the BaseForm member of the Source dimension? If it is loaded to the Base member, continue to the next question. If it is loaded to BaseForm, proceed to load data.
3. Are any of the driver formulas in the model scoped to exclude the target Analysis member? If so, read the following paragraph for more details. If not, proceed to load data.

If you are loading data from a model with driver formulas that are not scoped to execute on the target Analysis member that data is being loaded to, load the data to the BaseForm member of the Source dimension. This selection allows users to modify the

results loaded within the same Source member. In contrast, driver formula facts loaded to the Base member cannot be modified in the Base Source member.

Deleting Data and Driver Facts

The **Delete data from this cycle** task is similar to the load wizards in that data can be deleted by a subset of members for the Time, Analysis, IntOrg, Trader, Account, and Custom dimensions. For deletion of cycle facts, the administrator has the ability to select the following Source members: Base, BaseForm, and BaseJourn. Selecting the BaseForm Source member displays the option to delete data entered through forms. Note that selecting this option also deletes driver formula facts.



CHAPTER 11

Using Driver Formulas

Introduction *171*
Range of Execution *171*
What Triggers Execution of a Driver Formula *172*
Two Actions *172*
Driver Formulas That Are Triggered by Data Entry on a Form *172*
Overview *172*
Guidelines for Executing Driver Formulas *172*
Driver Formulas That Are Triggered by the Run driver formulas for this form set Option *173*
Form Design *173*
Form Design Restrictions *173*
Delete Behavior with Run driver formulas for this form set Option *173*
Number of Driver Formulas in a Model *174*

Introduction

Driver formulas provide a mechanism to create data using the context of a data-entry table. This chapter explains how driver formula results are created and provides recommendations for form design.

Range of Execution

The following items define the range of execution for a driver formula:

- formula scope (defined on the member).
- writable analyses (defined in the form set).
- writable crossings on the form set template after excluding crossings with hide visibility rules and crossings removed via Filter Member Combinations.
- the driver formulas included on the data-entry table in the form set.
- Driver formulas do not execute at TIME members which are locked in the cycle.

Effective with SAS Financial Management Studio 5.3, only drivers included on the data-entry table are executed. This is a change in behavior from previous versions that executed all driver formulas in a hierarchy, regardless of whether they were included on the table or not.

Also new in SAS Financial Management Studio 5.4, drivers that are removed from the table via Filter Member Combinations no longer execute. This change in behavior is consistent with the requirement that a crossing must be navigable on the table in order for the driver formula to execute.

What Triggers Execution of a Driver Formula

Two Actions

There are two ways to execute driver formulas:

- entering data into a form.
- selecting the **Run driver formulas for this form set** option in the Forms workspace. This option triggers the execution of driver formulas for the entire form set.

Note that selecting **Refresh** does not trigger the execution of driver formulas.

Driver Formulas That Are Triggered by Data Entry on a Form

Overview

In most scenarios, the results of driver formulas are generated when you enter values into a data-entry table. In these cases, no additional action is required. The next section covers other instances where additional action is required by an administrator to trigger driver formula execution.

Driver formula execution is triggered by the change in a value on a data-entry form. When the value is entered, the formula result is calculated and stored in the database.

Guidelines for Executing Driver Formulas

Based on their design and limited range of execution, driver formulas provide an efficient way to calculate values. The following guidelines apply to driver formula execution at the time of data entry:

- Only driver formulas that are included on the data-entry table in the form will run. Any driver formulas not on the data-entry table in the form will not execute at writeback or while using the Run drivers for this form set action.
- Driver formulas read from and write to the BaseForm member of the Source dimension. A driver formula can read from other members of the Source dimension if you explicitly state the members in the formula expression or implicitly state them as fixed members. A driver formula can write only to BaseForm; this is not modifiable.
- Driver formula *inputs* referenced in the formula expression must be included on a data-entry table to trigger driver formula execution at writeback. It is not necessary that the inputs are on the table for the Run driver formulas for this form set option.
- Driver formula expressions with relative time references such as ["TIME"=CURRENT("TIME")-1] can read from Time members that are not included in the form. However, the formula expressions write only to those Time members that are included in the form.

Driver Formulas That Are Triggered by the Run driver formulas for this form set Option

The **Run driver formulas for this form set** option is required for the following scenarios:

- changes to global values such as exchange rates, PRATE or DRATE rates, and formula expression inputs that are not in a given form.
- changes to a driver formula expression after data input.
- creation or deletion of a driver formula after data input.
- loading data records that affect driver formulas.
- Formula inputs are not included on the table.

Form Design

Form design defines the scope for execution of driver formulas when you select the **Run driver formulas for this form set** option. This option examines member selection for all slicers, rows, and columns. Limiting the number of slicers and members in a form limits the number of drivers that need to be executed. This improves performance. To optimize the performance of the **Run driver formulas for this form set** option, the following guidelines are recommended:

- Limit the dimension members in rows, columns, and slicers to the members that are required for data entry. Be sure to consider the Source and Trader dimensions.
- Use a separate read-only table for data that is needed in a form for information purposes only. For example, if you need to enter data for a budget, maintain the Actual data in a separate read-only table in the same form.
- Use Hide Visibility Rules or Filter Member Combinations to restrict the number of crossings required to run driver formulas.

Note: Only driver formulas included on the data-entry table are executed. This is a change in behavior from previous releases.

Form Design Restrictions

A form set whose target hierarchy is in the Account dimension cannot use the **Run driver formulas for this form set** option. This task is rendered inactive in the Forms workspace such that it cannot be selected. In this situation, you might need to consider other formula types for calculation needs.

Delete Behavior with Run driver formulas for this form set Option

When there are existing facts, the **Run driver formulas for this form set** option performs one of the following two actions depending upon the type of fact(s) that exist:

- If the fact is a driver fact from the same form set, it deletes the existing fact and replaces it with the newly executed driver fact.

- If the fact is not a driver fact from the same form set (such as form data entry, loading, or a previously executed driver fact from a different form set), the fact is *negated* and the newly executed driver fact will result.

Note: Only driver formulas which were generated by this form set are deleted.

Number of Driver Formulas in a Model

There is no limit to the number of driver formulas that can be created and used in a model. However, to optimize formula performance, you should define driver formulas on no more than 50 members.

CHAPTER 12

Optimizing Formula Performance

Overview ... *175*
Comparing the Execution of Driver Formulas and Modeling Formulas ... *175*
Managing Formulas for Performance ... *178*
Introduction ... *178*
Model Size ... *178*
Formula Scope ... *179*
Number of Formula Expression Inputs ... *179*
Using the SUBSTR Function ... *179*
Distributive Optimization for Modeling Formulas ... *179*
Visibility Rules ... *180*
Usage and Placement of Reporting Formulas ... *181*
Using CubeFormulasInfo and QueryStats Logging ... *181*
Applying Performance Improvements ... *182*
Introduction ... *182*
Example 1 ... *182*
Step 1. Create a Query with No Formulas ... *182*
Step 2. Add a Modeling Formula with Scoping on the Analysis Dimension ... *182*
Step 3. Add Additional Scoping to the Modeling Formula ... *183*
Step 4. Modify Formula Expression to Use Multiple Expressions per Member ... *184*
Example 2 ... *185*
Step 1. Create a Query with No Formulas ... *186*
Step 2, Add a Distributive Modeling Formula with Multiple Inputs and Cross-Dimensional References ... *187*
Step 3. Modify Account Hierarchy to Reduce Formula Expression Inputs ... *188*
Step 4. Remove Cross-Dimensional References ... *189*

Overview

This chapter describes the factors that can affect performance of driver, modeling and reporting formulas and the various options to improve performance. As described earlier, each formula type was designed for a specific intent. Driver and Modeling formulas are designed for data creation, whereas Reporting formulas are intended to report on the results of data in the form of ratios and other calculations.

There are few scenarios for which modeling formulas are recommended. Modeling formulas can be used for what-if analyses where inputs change frequently and results are expected instantly. However, carefully consider the performance factors described here. When possible, perform what-if analyses in a separate model to confine the performance cost.

Comparing the Execution of Driver Formulas and Modeling Formulas

A driver formula runs in the context of a data-entry form, which defines where data is being collected and therefore significantly reduces the scope for which the formula runs. A modeling formula runs across an entire model. The difference is dramatic. A form

might have hundreds or thousands of writable crossings while a model might have trillions or quadrillions.

A query for a driver formula's results returns values based on facts that were stored in the database after the last execution of the formula. Using driver formulas protects against unwanted changes after a budget is approved. A driver formula does not execute unless data is entered into a form or the **Run driver formulas for this form set** action is used.

A modeling formula runs each time a user submits a query, so the results are always based on the latest data and metadata. However, there is a performance cost. In many cases, the inputs have not changed so there is no benefit to running the formula again.

The following tables illustrate the difference in the number of crossings for execution of these two types of formulas. In this example, the modeling formula runs for 535 trillion cells while the driver formula is limited to 38,400 writable crossings.

Modeling Formula Scope:

Dimension	Model's Member Count	Create Modeling Formula with no scoping	Create Form Template (writable crossings for one Form below)
Account	1,258	1	30
Time	171	171	12
Intorg	16	16	1
Cost Center	767	767	20
Analysis	41	41	1
Source	21	21	1
Trader	18	18	1
Location	15	15	1
FOB	1,097	1,097	1
Total Cells	673,233,380,671,401,000	535,161,669,850,080	7,200

Extending the example from above for driver formulas, the following table illustrates the number of crossings for which the driver formula runs. Assume that there are 50 driver formulas in this model's hierarchy but only 10 driver formulas are included on the data-entry table. When a value is entered into a form for a single crossing, this entry triggers all driver formulas in the table for the crossing for which the value was entered. In this example, the number of accounts decreases from 30 (in the form set) to 10 (the number of driver formulas on the table). The driver formulas will be executed for all writable crossings on the form. In this example, entering a value in a form triggers the execution of driver formulas for a total of 2,400 crossings.

The form set uses INTORG as the target hierarchy, and there are 16 INTORG members. Therefore, the **Run driver formulas for this form set** action triggers formula execution for 38,400 crossings: the number of writable crossings for one form (12 Time periods * 20 Cost Centers) times the number of driver formulas (10) times the number of target members (16).

Driver Formula Scope:

Dimension	Create Form Template (writable crossings for one Form below)	Driver execution per Entered Value	Run Driver Formulas for Formset	
Account	30	10	10	
Time	12	12	12	
Intorg	1	1	16	
Cost Center	20	20	20	
Analysis	1	1	1	
Source	1	1	1	
Trader	1	1	1	
Location	1	1	1	
FOB	1	1	1	
Total Cells	7,200	2,400	38,400	Number of Cells Executed

The following factors affect the number of crossings where a driver formula runs when a user enters one numeric value in a form or when you select the **Run Driver formulas for this form set** option:

- number of driver formulas included on the data entry table
- number of writable crossings for each dimension defined in the form set on the data entry table

When you design a form set, limit the members in rows, columns, and slicers to relevant members that require input. Data that is required in a form for informational purposes should be displayed only in a separate read-only table. Use Visibility Rules to restrict the number of crossings.

The writable crossings for the Analysis dimension are based on the Writable Analysis Members selected in the Form Set. The writable crossings are further subset by a formula's scoping. After the formula's scope is applied, the writable crossings are further subset by the "selected" Analysis members. The Analysis members are selected through member selection rules on the data-entry template in Microsoft Excel.

The following table compares the number of crossings where a modeling formula runs with the number of crossings where a driver formula runs in the example just described.

Total Number of Cells / Crossings for formula execution

Action:	Modeling Formula	Driver Formula
Query	535,161,669,850,080	0
Enter data in a form	535,161,669,850,080	2,400
Use "Run driver formulas for this form set" action	N/A	38,400

Managing Formulas for Performance

Introduction

There are several factors that can affect the query and execution time for Reporting, Modeling, and Driver formulas. Formula performance is primarily attributed to the following:

- Model Size
- Formula Scope
- Number of formula expression inputs
- Use of the SUBSTR function
- Ability to use Distributive Optimization
- Visibility Rules
- Usage and Placement of Reporting Formulas

Model Size

The first step to understanding the performance impact of Modeling and Driver formulas is determining the size of the model. Model size is defined by the number of dimensions assigned to the model and the number of members in the selected hierarchies for those dimensions. To determine the total number of crossings in a given model, multiply the number of members in each dimension's hierarchy. In the following example, the model contains over 153 trillion crossings, computed as 9,189*118*83*45*43*21*7*6.

Dimension	Model's Member Count
Product	9,189
Time	118
Account	83
Trader	45
IntOrg	43
Source	21
Location	7
Analysis	6
Total Cells	153,595,292,630,220

To reduce model size, we suggest using Custom Properties rather than custom dimensions. An unlimited number of Custom Properties can be assigned to dimension members without increasing the model size or impacting formula performance.

Based on the example above, modifying the Model and Cycle design to make the Product custom dimension a Custom Property of the INTORG dimension results in an adjusted cube size of approximately 17 billion crossings, computed as 118*83*45*43*21*7*6. That's a reduction of more than 153 trillion crossings!

Formula Scope

Following consideration for the model design and size, the use of formula scoping is an effective means of reducing the number of crossings where a formula executes. Knowledge of where relevant data exists is crucial to maximizing the use of scoping, which can dramatically improve the performance of modeling formulas. The most common dimensions to consider for scoping are the Source, Trader, and Analysis dimensions.

To illustrate, refer to the example above with the model size of 153 trillion crossings. Scoping for an individual modeling formula on the Source, Trader, and Analysis dimensions reduces the number of crossings from 153 trillion to approximately 326 million.

Number of Formula Expression Inputs

While the previous two factors address limiting the number of crossings, the next three factors apply specifically to the formula expression. Following careful review of model size and formula scoping to limit formula execution, minimizing the number of formula inputs referenced in the formula expression is another means of optimizing formula performance. Where possible, design a hierarchy that enables formulas to refer to one parent instead of many children. For example, a formula containing ["ACCOUNT"="Parent"] runs faster than a formula containing ["ACCOUNT”="child1"] + ["ACCOUNT"="child2"] + ["ACCOUNT”= "child3"].

Using the SUBSTR Function

Limiting the use of certain functions in the formula expression such as SUBSTR improves formula performance. In most cases, designing a hierarchy such that scoping can be applied or using Custom Properties eliminates the need for the SUBSTR function in formula expression. For example, using scoping with a member selection rule such as “Leaf descendants of Member, otherwise Member” or using property scoping results in a more efficient formula expression (less time required to execute).

Distributive Optimization for Modeling Formulas

For eligible formula expressions, SAS Financial Management applies distributive optimizations for enhanced formula execution processing. These optimizations are performed on a per-dimension basis. An example of a formula that qualifies for distributive optimization is as follows:

> Estimated Benefits = ["ACCOUNT"="Salaries"] * ["ACCOUNT"="Estimated Benefit Percentage"]`

The following characteristics of a modeling formula expression prevent the use of distributive optimizations in any dimension:

- The expression uses the DRATE function, the XRATE function, or the ISLEAF function.
- The expression is on an account whose account type is Statistical.
- The expression refers to an account whose account type is Statistical.
- The expression uses the IF function or the NESTIF function. Any formula that contains a member-based IF function or NESTIF function should be replaced with multiple expressions on the same member, each appropriately scoped.

- The expression adds a constant to another term. An example is ["ACCOUNT"="A"] + 2.
- The expression multiplies or divides the value at one crossing by the value at another crossing. An example is ["ACCOUNT"="A"]/["ACCOUNT"="B"].

If a distributive expression contains a reference to a member of any dimension other than the Account dimension (either explicitly or as a fixed member), then the distributive optimization is not used in that dimension. This becomes increasingly expensive as the number of such references and the size of the relevant dimensions increases.

Note: A formula is never distributive in the Time dimension.

If a modeling formula expression cannot take advantage of the distributive optimizations, then it is especially important to apply formula scoping (where possible). This limits the number of times that the expression is executed.

Visibility Rules

The number of cells for which the "Run driver formulas for this form set" action executes can be reduced dramatically by applying Visibility Rules. The following table demonstrates the impact when visibility rules are utilized.

Dimension	Form's Writeable Crossings without Visibility Rules Enabled	Form's Writeable Crossings with Visibility Rules Enabled
Time	12	12
IntOrg	1	1
Location	2,000	4
Cost Center	10	10
Analysis	1	1
Source	1	1
Trader	1	1
FOB	11	11
Number of driver formulas on data entry table	10	10
Total Cell Executions per Form	26,400,000	52,800
Total Forms	500	500
Total Cells	13,200,000,000	26,400,000
		Reduction of > 99%

Note: IntOrg is the Target Hierarchy for the form set.

In the table above, the reduction of members in the Location dimension is due to Visibility Rules created between the IntOrg and Location dimension. For each form, there are approximately four valid locations that for each IntOrg. Applying visibility rules removes non-relevant combinations between these two dimensions, thereby reducing the total number of crossings where drivers could execute.

Note: Visibility Rules are most effective when the entire set of visibility rules references members in **no more than two** dimensions. If it is necessary to include members of more than two dimensions, performance time to run the query and run drivers will be negatively impacted.

Usage and Placement of Reporting Formulas

Reporting formulas are designed to calculate the results of data and report on those results. Since reporting formulas do not contribute to rollup values, Retained Earnings, or Cumulative Translation Adjustment account types, they are not recommended as a means of data creation and should not be used as such.

Creating a reporting formula with extensive scoping to run at only leaf levels similar to modeling and driver formulas will significantly impact query and execution time. This design results in the need to create additional reporting formulas in each dimension to re-create rollup logic to include the values generated by other reporting formulas.

Note: It is strongly discouraged to utilize reporting formulas in this manner because of performance implications. Also, we suggest limiting the number of calculated members assigned as reporting formulas to **no more than two** dimensions, where possible.

Using CubeFormulasInfo and QueryStats Logging

If you set the com.sas.solutions.odcs.formulas.CubeFormulasInfo package in the logging.xml file to INFO, information is logged about each formula and whether it is marked as distributive in any dimension.

For example, this formula is distributive in five dimensions:

(["ACCOUNT"="01"] + ["ACCOUNT"="02"] + ["ACCOUNT"="03"]) * .3 * -1

This distribution is shown by CubeFormulasInfo logging:

[ACCOUNT].[3530000899](test) id=325 RefsRollup rhs=[]
distrib=[3_COSTCTR,4_INTORG,5_TRADER,6_FOB,7_LOCATION]

The following formula is not distributive because it contains a reference to a cell value from a conditional statement:

IF (ABS (["SOURCE"="Base"]) > .01, 0 ,
(["ACCOUNT"="01"]["LOCATION"="000"] ["SOURCE"="Total"] +
["ACCOUNT"="02"]["LOCATION"="000"] ["SOURCE"="Total"] +
["ACCOUNT"="33"]["LOCATION"="000"] ["SOURCE"="Total"]) * .21)

CubeFormulasInfo does not display any dimensions as being distributive:

[ACCOUNT].[3530000899](test2) id=340 RefsRollup rhs=[] **distrib=[]**

If you set the com.sas.solutions.odcs.query.QueryStats package in the logging.xml file to INFO, information such as processing time and cell count for formula execution is logged for a query:

```
2007-07-11 14:11:49,985 INFO [QueryStats] Total: 00:21.759 (3040 cells),
Fact Map Build (209 chunks): 00:00.016, Graph Build: 00:00.016, Graph
Execution: 00:21.634, Reporting Graph Build: 00:00.000, Reporting Graph
Execution: 00:00.000, Fact Processing (tot/exc/skip/ice/rel)
(60235/0/59760/0/475): 00:00.078

Graph (14 nodes/112042240 cells): Accounting 13/103423840, LeafFormula
1/8618400
```

Applying Performance Improvements

Introduction

This section applies the recommended performance measures on a step-by-step basis, comparing execution time and cell count by step to illustrate the impact these measures have on formula performance.

Example 1

- Step 1: Create a query with no formulas.
- Step 2: Add a Modeling formula with scoping on the Analysis dimension.
- Step 3: Add additional scoping to the modeling formula.
- Step 4: Modify formula expression to use multiple expressions per member.

The results of these steps are displayed below, followed by the details per step:

		Distributive/Non-Distributive	Cell Count	Execution Time
Step 1	No formulas	n/a	144,127,747,431,360	00:00.0
Step 2	Add a modeling formula with scoping on the Analysis dimension	Non-Distributive	76,622,938,560	OOM
Step 3	Scope on Source & Trader	Non-Distributive	266,978,880	00:21.5
Step 4	Replace IF function using Multiple Expressions per Member	Non-Distributive	266,978,880	00:01.2

Step 1. Create a Query with No Formulas

The initial query contains 144,127,747,431,360 crossings based on the following cardinalities: 209,21,9,41,41,2720,133,6. Based on the QueryStats information, the resulting query rendered in Excel in less than one second as displayed below:

```
2007-07-12 09:39:31,617 INFO [Query] Processed query in 0ms

2007-07-12 09:39:31,617 INFO [QueryStats] Total: 00:00.000 (3040 cells),
Fact Map Build (13 chunks): 00:00.000, Graph Build: 00:00.000, Graph
Execution: 00:00.000, Reporting Graph Build: 00:00.000, Reporting Graph
Execution: 00:00.000, Fact Processing (tot/exc/skip/ice/rel)
(60235/0/60235/0/0): 00:00.000

Graph (1 nodes/3040 cells): Accounting 1/3040
```

		Distributive/Non-Distributive	Cell Count	Execution Time
Step 1	No formulas	n/a	144,127,747,431,360	00:00.0

Step 2. Add a Modeling Formula with Scoping on the Analysis Dimension

From the initial query, a single Modeling formula is added with the following IF statement expression:

Expression

IF((CURRENT("INTORG") = "100" | CURRENT("INTORG") = "300" | CURRENT("INTORG") = "400"),

IF(["ACCOUNT"="2.1.3.EXT"] ^= 0,(["ACCOUNT"="1.2.1"]+["ACCOUNT"="1.2.2"]+["ACCOUNT"="1.2.3"]+["ACCOUNT"="1.2.4"]+["ACCOUNT"="1.2.5"]+["ACCOUNT"="1.2.6"]+["ACCOUNT"="1.2.8"]), ["ACCOUNT"="2.1.4"]),

(["ACCOUNT"="2.3.1"]+["ACCOUNT"="2.3.2"]+["ACCOUNT"="2.3.3"]))

Formula

The formula is scoped to a single member of the Analysis dimension:

ANALYSIS: FORECAST_PRO - Member

Based on the use of the IF function in the formula expression, this formula is considered Non-Distributive. The following information is provided:

Cube FormulasInfo

```
Planning Formulas formulas=1 chunks=1
[ACCOUNT].[2.1.3](all orgs) id=1117 rhs=[] distrib=[]
```

QueryStats Logging

```
java.lang.OutOfMemoryError
```

Due to the cell count size, adding a single Modeling formula with minimal scoping on the Analysis dimension results in an Out of Memory error. In this situation, no values are returned for the query.

		Distributive/Non-Distributive	Cell Count	Execution Time
Step 1	No formulas	n/a	144,127,747,431,360	00:00.0
Step 2	Add a modeling formula with scoping on the Analysis dimension	Non-Distributive	76,622,938,560	OOM

Step 3. Add Additional Scoping to the Modeling Formula

In order to eliminate the Out of Memory error, further scoping is required to limit the execution range of the formula expression. The following additional dimension members from the Source and Trader dimension are included for formula scoping:

Formula

TRADER: EXT - Leaf descendants of member if any, otherwise Member
SOURCE: Base - Leaf descendants of member if any, otherwise Member
SOURCE: BaseForm - Leaf descendants of member if any, otherwise Member
SOURCE: BaseJourn - Leaf descendants of member if any, otherwise Member

While there is no change in the distributive property of the formula expression and the CubeFormulasInfo detail remains unchanged, the query no longer runs out of memory. The QueryStats Logging detail is as follows:

```
2007-07-12 09:49:44,558 INFO [Query] Processed query in 21452ms

2007-07-12 09:49:44,558 INFO [QueryStats] Total: 00:21.452 (3040 cells),
Fact Map Build (209 chunks): 00:00.032, Graph Build: 00:00.031, Graph
Execution: 00:21.374, Reporting Graph Build: 00:00.000, Reporting Graph
```

```
Execution: 00:00.000, Fact Processing (tot/exc/skip/ice/rel)
(60235/0/59760/0/475): 00:00.015

Graph (14 nodes/112042240 cells): Accounting 13/103423840, LeafFormula
1/8618400
```

The additional formula scoping renders a reasonable cell count with an execution time of approximately 21 seconds:

		Distributive/Non-Distributive	Cell Count	Execution Time
Step 1	No formulas	n/a	144,127,747,431,360	00:00.0
Step 2	Add a modeling formula with scoping on the Analysis dimension	Non-Distributive	76,622,938,560	OOM
Step 3	Scope on Source & Trader	Non-Distributive	266,978,880	00:21.5

Step 4. Modify Formula Expression to Use Multiple Expressions per Member

Modifying the formula expression and using multiple expressions further improves the query performance. While the initial formula scoping on the Analysis, Source, and Trader dimensions remains, additional scoping in the INTORG dimension is applied on the first formula.

This additional scoping does not impact the cell count due to the fact that the second formula does not also scope on the INTORG dimension members. This means that the cell count remains unchanged from step 3. The modified formulas expressions and scoping are as follows:

Formula 1

Name: orgs 100,300,400

Expression

IF(["ACCOUNT"="2.1.3.EXT"] ^= 0,(["ACCOUNT"="1.2.1"]+["ACCOUNT"="1.2.2"]+["ACCOUNT"="1.2.3"]+["ACCOUNT"="1.2.4"]+["ACCOUNT"="1.2.5"]+["ACCOUNT"="1.2.6"]+["ACCOUNT"="1.2.8"]), ["ACCOUNT"="2.1.4"])

Scope

TRADER: EXT - Leaf descendants of member if any, otherwise Member
SOURCE: Base - Leaf descendants of member if any, otherwise Member
SOURCE: BaseForm - Leaf descendants of member if any, otherwise Member
SOURCE: BaseJourn - Leaf descendants of member if any, otherwise Member
ANALYSIS: FORECAST_PRO - Member
INTORG: 100 - Leaf descendants of member if any, otherwise Member
INTORG: 300 - Leaf descendants of member if any, otherwise Member
INTORG: 400 - Leaf descendants of member if any, otherwise Member

Formula 2

Name: all other orgs

Expression

(["ACCOUNT"="2.3.1"]+["ACCOUNT"="2.3.2"]+["ACCOUNT"="2.3.3"])

Scope

TRADER: EXT - Leaf descendants of member if any, otherwise Member
SOURCE: Base - Leaf descendants of member if any, otherwise Member
SOURCE: BaseForm - Leaf descendants of member if any, otherwise Member
SOURCE: BaseJourn - Leaf descendants of member if any, otherwise Member
ANALYSIS: FORECAST_PRO - Member

While there is no change in the distributive property of the formula expression and the cell count remains unchanged, the time to execute the query has been significantly reduced.

```
2007-07-12 11:16:43,972 INFO [Query] Processed query in 1188ms

2007-07-12 11:16:43,972 INFO [QueryStats] Total: 00:01.188 (3040 cells),
Setup: 00:00.015, Fact Map Build (531 chunks): 00:00.015, Graph Build:
00:00.032, Graph Execution: 00:01.141, Reporting Graph Build: 00:00.000,
Reporting Graph Execution: 00:00.000, Fact Processing
(tot/exc/skip/ice/rel) (60235/0/59760/0/303): 00:00.000

Graph (15 nodes/7136320 cells): Accounting 13/6373600, LeafFormula
2/762720
```

This example illustrates how the combined use of formula scoping and multiple expressions per member can render queries with execution times virtually equal to that of queries containing no formulas.

		Distributive/Non-Distributive	Cell Count	Execution Time
Step 1	No formulas	n/a	144,127,747,431,360	00:00.0
Step 2	Add a modeling formula with scoping on the Analysis dimension	Non-Distributive	76,622,938,560	OOM
Step 3	Scope on Source & Trader	Non-Distributive	266,978,880	00:21.5
Step 4	Replace IF function using Multiple Expressions per Member	Non-Distributive	266,978,880	00:01.2

Example 2

- Step 1: Create a query with no formulas.
- Step 2: Add a Distributive Modeling formula with multiple inputs and cross-dimensional references.
- Step 3: Modify Account hierarchy to reduce formula expression inputs.
- Step 4: Remove cross-dimensional references.

The results of these steps are displayed below, followed by the details per step.

		Distributive/Non-Distributive	Cell Count	Execution Time
Step 1	No formulas	n/a	757,387,553,255,325,000	00:00.0
Step 2	Add a distributive modeling formula with multiple inputs and cross-dimensional references	Distributive	199,639,025,796,420	01:02.7
Step 3	Modify Account hierachy to reduce formula expression inputs	Distributive	199,639,025,796,420	00:11.7
Step 4	Remove cross-dimensional references	Distributive	199,639,025,796,420	00:00.6

Step 1. Create a Query with No Formulas

In this example, the initial query contains 757,387,553,255,325,000 crossings based on the following cardinalities: 1258, 21, 41, 767, 18, 18, 1097, 15, 171, 1. Based on the QueryStats information, the resulting query rendered in Excel in less than one second as displayed below:

```
2007-07-12 12:27:35,035 INFO [Query] Processed query in 46ms

2007-07-12 12:27:35,035 INFO [QueryStats] Total: 00:00.046 (15680 cells),
Fact Map Build (71 chunks): 00:00.000, Graph Build: 00:00.015, Graph
Execution: 00:00.015, Reporting Graph Build: 00:00.000, Reporting Graph
Execution: 00:00.000, Fact Processing (tot/exc/skip/ice/rel)
(160720/0/159033/0/1146): 00:00.016

Graph (1 nodes/15680 cells): Accounting 1/15680
```

		Distributive/Non-Distributive	Cell Count	Execution Time
Step 1	No formulas	n/a	757,387,553,255,325,000	00:00.0

Step 2, Add a Distributive Modeling Formula with Multiple Inputs and Cross-Dimensional References

From the initial query, a single Modeling formula is added with the following formula expression and scoping:

Expression

["ACCOUNT"="02"]+(["ACCOUNT"="03"]["INTORG"="ROLLUP WITH 7 CHILDREN"]["COSTCTR"="99"])+["ACCOUNT"="04"]+["ACCOUNT"="06"]+ ["ACCOUNT"="07"]+["ACCOUNT"="08"]+["ACCOUNT"="09"]+["ACCOUNT"="10"]

Scope

INTORG: 75 - Leaf descendants of member if any, otherwise Member
INTORG: 20 - Leaf descendants of member if any, otherwise Member
INTORG: 40 - Leaf descendants of member if any, otherwise Member
INTORG: 17 - Leaf descendants of member if any, otherwise Member
INTORG: 63 - Leaf descendants of member if any, otherwise Member
INTORG: 64 - Leaf descendants of member if any, otherwise Member
INTORG: 65 - Leaf descendants of member if any, otherwise Member
COSTCTR: BAL - Leaf descendants of member if any, otherwise Member
COSTCTR: 1000 - Leaf descendants of member if any, otherwise Member
COSTCTR: 1001 - Leaf descendants of member if any, otherwise Member
COSTCTR: 1002 - Leaf descendants of member if any, otherwise Member
COSTCTR: 100300 - Leaf descendants of member if any, otherwise Member
COSTCTR: 100400 - Leaf descendants of member if any, otherwise Member
COSTCTR: 1100 - Leaf descendants of member if any, otherwise Member
COSTCTR: 15 - Leaf descendants of member if any, otherwise Member
COSTCTR: 20 - Leaf descendants of member if any, otherwise Member
COSTCTR: 25 - Leaf descendants of member if any, otherwise Member
COSTCTR: 30 - Leaf descendants of member if any, otherwise Member
COSTCTR: 35 - Leaf descendants of member if any, otherwise Member
COSTCTR: 40 - Leaf descendants of member if any, otherwise Member
COSTCTR: 45 - Leaf descendants of member if any, otherwise Member
COSTCTR: 50 - Leaf descendants of member if any, otherwise Member
COSTCTR: 55 - Leaf descendants of member if any, otherwise Member
COSTCTR: 60 - Leaf descendants of member if any, otherwise Member
COSTCTR: 65 - Leaf descendants of member if any, otherwise Member
COSTCTR: 70 - Leaf descendants of member if any, otherwise Member
COSTCTR: 75 - Leaf descendants of member if any, otherwise Member
COSTCTR: 80 - Leaf descendants of member if any, otherwise Member
COSTCTR: 85 - Leaf descendants of member if any, otherwise Member
COSTCTR: 90 - Leaf descendants of member if any, otherwise Member
COSTCTR: 95 - Leaf descendants of member if any, otherwise Member
COSTCTR: NEW - Leaf descendants of member if any, otherwise Member

The CubeFormulasInfo shows the formula expression is Distributive in four dimensions:

```
2007-07-12 12:36:27,992 INFO [CubeFormulasInfo]

Planning Formulas formulas=1 chunks=1

[ACCOUNT].[3530000899](Location 139) id=343 RefsRollup rhs=[]
distrib=[1_SOURCE,5_TRADER,6_FOB,7_LOCATION]
```

Despite its distributivity and scoping, Query Stats Logging shows that query execution time is now 1.02 minutes.

```
2007-07-12 12:37:30,771 INFO [Query] Processed query in 62732ms

2007-07-12 12:37:30,771 INFO [QueryStats] Total: 01:02.732 (2240 cells),
Fact Map Build (195 chunks): 00:00.141, Graph Build: 00:00.000, Graph
Execution: 01:02.575, Reporting Graph Build: 00:00.000, Reporting Graph
Execution: 00:00.000, Fact Processing (tot/exc/skip/ice/rel)
(160720/0/160120/0/600): 00:00.016

Graph (10 nodes/75354440 cells): Accounting 9/65937032, LeafFormula
1/9417408
```

		Distributive/Non-Distributive	Cell Count	Execution Time
Step 1	No formulas	n/a	757,387,553,255,325,000	00:00.0
Step 2	Add a distributive modeling formula with multiple inputs and cross-dimensional references	Distributive	199,639,025,796,420	01:02.7

Step 3. Modify Account Hierarchy to Reduce Formula Expression Inputs

To reduce query execution time, design an Account hierarchy that uses SAS Financial Management's roll-up logic. This hierarchy significantly improves query performance time for formula expressions that reference roll-ups instead of the members that contribute to those roll-ups. In this step, the revised hierarchy allows the formula expression to be written as follows:

Expression

(["ACCOUNT"="03"]["INTORG"="ROLLUP WITH 7 CHILDREN"]["COSTCTR"="99"])+["ACCOUNT"="new rollup"]

Scope

INTORG: 75 - Leaf descendants of member if any, otherwise Member
INTORG: 20 - Leaf descendants of member if any, otherwise Member
INTORG: 40 - Leaf descendants of member if any, otherwise Member
INTORG: 17 - Leaf descendants of member if any, otherwise Member
INTORG: 63 - Leaf descendants of member if any, otherwise Member
INTORG: 64 - Leaf descendants of member if any, otherwise Member
INTORG: 65 - Leaf descendants of member if any, otherwise Member
COSTCTR: BAL - Leaf descendants of member if any, otherwise Member
COSTCTR: 1000 - Leaf descendants of member if any, otherwise Member
COSTCTR: 1001 - Leaf descendants of member if any, otherwise Member
COSTCTR: 1002 - Leaf descendants of member if any, otherwise Member
COSTCTR: 100300 - Leaf descendants of member if any, otherwise Member
COSTCTR: 100400 - Leaf descendants of member if any, otherwise Member
COSTCTR: 1100 - Leaf descendants of member if any, otherwise Member
COSTCTR: 15 - Leaf descendants of member if any, otherwise Member
COSTCTR: 20 - Leaf descendants of member if any, otherwise Member
COSTCTR: 25 - Leaf descendants of member if any, otherwise Member
COSTCTR: 30 - Leaf descendants of member if any, otherwise Member
COSTCTR: 35 - Leaf descendants of member if any, otherwise Member

COSTCTR: 40 - Leaf descendants of member if any, otherwise Member
COSTCTR: 45 - Leaf descendants of member if any, otherwise Member
COSTCTR: 50 - Leaf descendants of member if any, otherwise Member
COSTCTR: 55 - Leaf descendants of member if any, otherwise Member
COSTCTR: 60 - Leaf descendants of member if any, otherwise Member
COSTCTR: 65 - Leaf descendants of member if any, otherwise Member
COSTCTR: 70 - Leaf descendants of member if any, otherwise Member
COSTCTR: 75 - Leaf descendants of member if any, otherwise Member
COSTCTR: 80 - Leaf descendants of member if any, otherwise Member
COSTCTR: 85 - Leaf descendants of member if any, otherwise Member
COSTCTR: 90 - Leaf descendants of member if any, otherwise Member
COSTCTR: 95 - Leaf descendants of member if any, otherwise Member
COSTCTR: NEW - Leaf descendants of member if any, otherwise Member

While the cell count remains unchanged based on this modification, query execution time is significantly improved as displayed in the QueryStats Logging Information below:

```
2007-07-12 12:45:11,106 INFO [Query] Processed query in 11733ms
```

```
2007-07-12 12:45:11,106 INFO [QueryStats] Total: 00:11.733 (2240 cells),
Fact Map Build (51 chunks): 00:00.015, Graph Build: 00:00.000, Graph
Execution: 00:11.718, Reporting Graph Build: 00:00.000, Reporting Graph
Execution: 00:00.000, Fact Processing (tot/exc/skip/ice/rel)
(160720/0/160120/0/600): 00:00.000
```

```
Graph (4 nodes/18849992 cells): Accounting 3/9432584, LeafFormula
1/9417408
```

		Distributive/Non-Distributive	Cell Count	Execution Time
Step 1	No formulas	n/a	757,387,553,255,325,000	00:00.0
Step 2	Add a distributive modeling formula with multiple inputs and cross-dimensional references	Distributive	199,639,025,796,420	01:02.7
Step 3	Modify Account hierachy to reduce formula expression inputs	Distributive	199,639,025,796,420	00:11.7

Step 4. Remove Cross-Dimensional References

Modifying the existing formula expression to remove cross-dimensional references to the INTORG and COSTCTR dimensions significantly improves performance since now the INTORG and COSTCTR dimensions are considered distributive. The revised formula and resulting CubeFormulasInfo Logging detail are displayed below:

Expression

["ACCOUNT"="03"]+["ACCOUNT"="new rollup"]

Scope

INTORG: 75 - Leaf descendants of member if any, otherwise Member
INTORG: 20 - Leaf descendants of member if any, otherwise Member
INTORG: 40 - Leaf descendants of member if any, otherwise Member
INTORG: 17 - Leaf descendants of member if any, otherwise Member
INTORG: 63 - Leaf descendants of member if any, otherwise Member
INTORG: 64 - Leaf descendants of member if any, otherwise Member
INTORG: 65 - Leaf descendants of member if any, otherwise Member
COSTCTR: BAL - Leaf descendants of member if any, otherwise Member
COSTCTR: 1000 - Leaf descendants of member if any, otherwise Member
COSTCTR: 1001 - Leaf descendants of member if any, otherwise Member
COSTCTR: 1002 - Leaf descendants of member if any, otherwise Member
COSTCTR: 100300 - Leaf descendants of member if any, otherwise Member
COSTCTR: 100400 - Leaf descendants of member if any, otherwise Member
COSTCTR: 1100 - Leaf descendants of member if any, otherwise Member
COSTCTR: 15 - Leaf descendants of member if any, otherwise Member
COSTCTR: 20 - Leaf descendants of member if any, otherwise Member
COSTCTR: 25 - Leaf descendants of member if any, otherwise Member
COSTCTR: 30 - Leaf descendants of member if any, otherwise Member
COSTCTR: 35 - Leaf descendants of member if any, otherwise Member
COSTCTR: 40 - Leaf descendants of member if any, otherwise Member
COSTCTR: 45 - Leaf descendants of member if any, otherwise Member
COSTCTR: 50 - Leaf descendants of member if any, otherwise Member
COSTCTR: 55 - Leaf descendants of member if any, otherwise Member
COSTCTR: 60 - Leaf descendants of member if any, otherwise Member
COSTCTR: 65 - Leaf descendants of member if any, otherwise Member
COSTCTR: 70 - Leaf descendants of member if any, otherwise Member
COSTCTR: 75 - Leaf descendants of member if any, otherwise Member
COSTCTR: 80 - Leaf descendants of member if any, otherwise Member
COSTCTR: 85 - Leaf descendants of member if any, otherwise Member
COSTCTR: 90 - Leaf descendants of member if any, otherwise Member
COSTCTR: 95 - Leaf descendants of member if any, otherwise Member
COSTCTR: NEW - Leaf descendants of member if any, otherwise Member

Logging CubeFormulasInfo

```
Planning Formulas formulas=1 chunks=1

[ACCOUNT].[3530000899](Distrib test) id=345 RefsRollup rhs=[]
distrib=[1_SOURCE,3_COSTCTR,4_INTORG,5_TRADER,6_FOB,7_LOCATION]

   ["ACCOUNT"="03"]+["ACCOUNT"="new rollup"]
```

The resulting revised formula expression significantly improves query execution time as displayed below in the QueryStats Logging information:

```
2007-07-12 12:47:34,333 INFO [Query] Processed query in 593ms

2007-07-12 12:47:34,333 INFO [QueryStats] Total: 00:00.593 (2240 cells),
Fact Map Build (71 chunks): 00:00.015, Graph Build: 00:00.000, Graph
Execution: 00:00.578, Reporting Graph Build: 00:00.000, Reporting Graph
Execution: 00:00.000, Fact Processing (tot/exc/skip/ice/rel)
(160720/0/160120/0/600): 00:00.000

Graph (4 nodes/1360520 cells): Accounting 3/907760, LeafFormula 1/452760
```

		Distributive/Non-Distributive	Cell Count	Execution Time
Step 1	**No formulas**	n/a	757,387,553,255,325,000	00:00.0
Step 2	**Add a distributive modeling formula with multiple inputs and cross-dimensional references**	Distributive	199,639,025,796,420	01:02.7
Step 3	**Modify Account hierachy to reduce formula expression inputs**	Distributive	199,639,025,796,420	00:11.7
Step 4	**Remove cross-dimensional references**	Distributive	199,639,025,796,420	**00:00.6**

This example illustrates how minimizing formula inputs and cross-dimensional references in the formula expression, as well as applying formula scoping can render queries with execution times virtually equal to that of queries containing no formulas at all.

CHAPTER 13

Troubleshooting

Introduction *193*
Troubleshooting in SAS Financial Management Studio *193*
Reviewing Formula Validation within a Model *193*
Troubleshooting in Microsoft Excel *194*
Verifying Formula Results with the Excel Add-In *195*
Introduction *195*
Step 1: Verify That the Formula Expression Is Executing for the Analyzed Cell *195*
Step 2: Include All Operands on the Table; Review Formula Input Values *196*
Step 3: Ensure That Inputs Match the Formula Expression *196*
Formula Results in a NaN or Zero *197*
Invalid Reporting and/or Modeling Formula Results in NaN or Red Cell *197*
Invalid Driver Formula Results in Zero *197*
Troubleshooting Excel-based Calculated Members *198*

Introduction

This chapter discusses the tools and information available in the following locations to investigate invalid calculated members and formula results for valid calculated members:

- SAS Financial Management Studio
- the Microsoft Excel Add-in

Troubleshooting in SAS Financial Management Studio

Reviewing Formula Validation within a Model

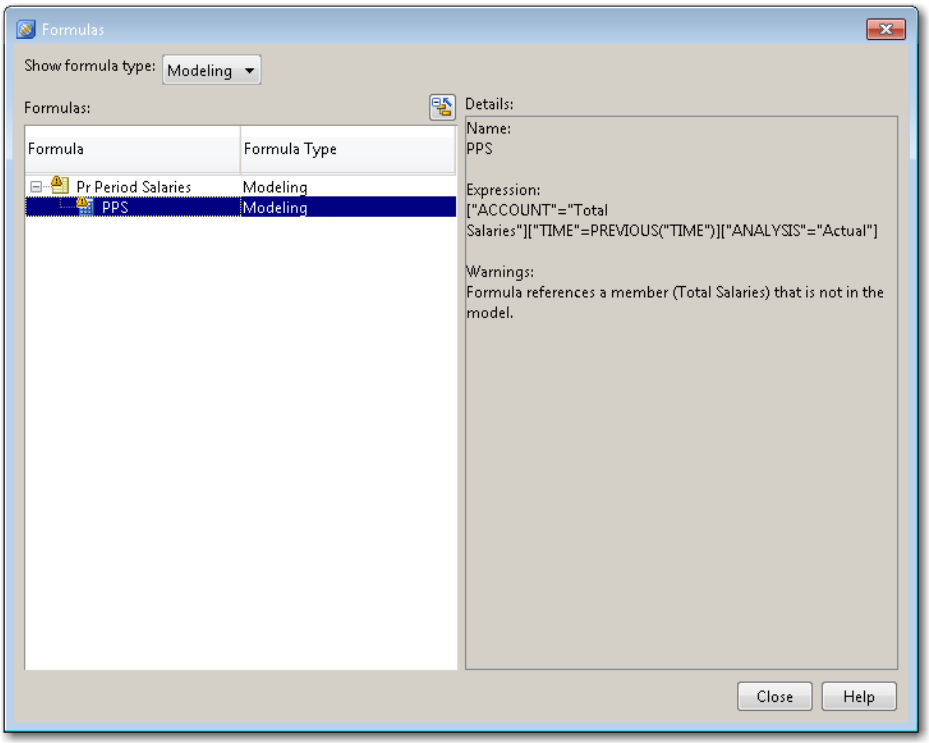
Formula syntax validation is performed when a formula expression is created on a calculated member in the Dimensions workspace in SAS Financial Management Studio. The validation ensures that the dimensions and members referenced in the formula expression exist and that the expression is syntactically correct. While the calculated member might pass validation in the Dimensions workspace, it does not ensure that the formula is valid within the context of a model.

To illustrate, the following formula example is valid in the context of the Dimensions workspace based on its syntax and accurate reference to dimension members:

Previous Period Salaries =
["ACCOUNT"="Total Salaries"]["TIME"=PREVIOUS("TIME")]["ANALYSIS"="ACTUAL"]

In a model where the Account hierarchy does not contain the Account member “Total Salaries”, the formula expression is invalid within the context of that model. Formula validation information about a model basis can be viewed in Model Properties on the **Formulas** tab. Select **Show Formulas**. A warning icon is displayed next to any calculated member that contains an invalid formula expression.

In the example provided above, the following warning is displayed on the formula expression that refers to that Account because “Total Salaries” is not in the model’s Account hierarchy:



To minimize the number of invalid formula expressions displayed in a table, it is recommended that you resolve all Model formula warnings before viewing formula results in Excel.

Troubleshooting in Microsoft Excel

In some cases, formula results in the context of an Excel can still be invalid, even when they are valid in the context of a Model. The results of these invalid table-based formula expressions are either a NaN for Reporting or Modeling formulas or a zero for Driver formulas, Invalid table-based formula expressions are typically the result of one of the following issues:

- Missing exchange rates.
- Missing DRATES.
- Security does not allow viewing rights.
- The denominator for the expression is zero.

The following sections discuss troubleshooting invalid formula expressions in both Excel and.

Verifying Formula Results with the Excel Add-In

Introduction

If a formula returns an unexpected value, follow these steps to determine how the value was derived.

Step 1: Verify That the Formula Expression Is Executing for the Analyzed Cell

1. In an Excel table, select the cell containing the unexpected value.
2. Right-click the cell.
3. Select **Tools ▶ Cell Information** to confirm the formula expression executing for this crossing.

   If a formula is executing for this cell, the formula type, expression, Fixed Members, if any, and the name of the formula expression is displayed. Note that only the first 200 characters of a formula expression are displayed followed by a "…" to indicate when the entire expression extends beyond the maximum number of allowable characters.

If a different formula expression is displayed, go to the Dimensions workspace in SAS Financial Management Studio to review the scoping and rank of this formula expression compared to the scoping and rank of the formula expression you were expecting to execute for this cell. If the formula is a Reporting formula, you might need to change the Dimension rank in the Model Properties dialog box on the Formulas page if you have two reporting formulas in different dimensions executing for the same crossing.

If no formula information is provided in Cell Information, then the value in this cell is not being generated by a formula. If you are expecting a modeling or driver formula to execute here, follow these steps:

1. Ensure that you are at a leaf crossing by putting every dimension on the table and selecting a leaf member for each dimension on the slicers.
2. Right-click a cell containing all leaf-level members.
3. Select **Tools ▶ Cell Information**.

If no formula information is provided, follow these steps to review the Model Properties for the related model.

1. In SAS Financial Management Studio, select the Dimensions page under Model Properties.
2. Highlight the dimension that contains the formula.
3. Click the **Preview** button by the **Hierarchy** drop-down list.
4. Ensure that the member is in the hierarchy and that it has a calculated member icon.
5. Select **Formulas** on Model Properties and select the **Show Formulas** button.
6. Find the formula in this window and confirm that the expression and scoping are correct.

Note: If the model is not using the current version of this hierarchy, the expression in the Model properties might be different than the one you see in the **Dimensions** workspace. The expression in the **Dimensions** workspace is the current version.

Step 2: Include All Operands on the Table; Review Formula Input Values

If the formula is a modeling or driver formula, perform validation in the functional currency of the ORG member, as explained earlier in the Formula Computation chapter.

Reporting formula results can be validated in any currency.

Step 3: Ensure That Inputs Match the Formula Expression

For example, the following formula expression does not appear to be executing:

Account C = ["ACCOUNT"="Account A"]*["ACCOUNT"="Account B"]

Organization	Corporate Legal
Customer	Customer Total
Time	Mar 2013
Analysis	Actual
Currency	USD
Frequency	PTD

	Product Total
Account A	3.00
Account B	4.00
Account C	0.00

After expanding the table to display leaf crossings, it is apparent that the formula inputs were entered at different Products (Simulation and Arcade) resulting in a value of zero for Account C.

Organization	Corporate Legal
Customer	Customer Total
Time	Mar 2013
Analysis	Actual
Currency	USD
Frequency	PTD

	Product Total	Video Games	Action	Simulation	Arcade	Puzzle	Hardware	Publications
Account A	3.00	3.00	0.00	3.00	0.00	0.00	0.00	0.00
Account B	4.00	4.00	0.00	0.00	4.00	0.00	0.00	0.00
Account C	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00

Formula Results in a NaN or Zero

Invalid Reporting and/or Modeling Formula Results in NaN or Red Cell

When the result of a Reporting and/or Modeling formula is a NaN or red cell, users can select **Tools ▶ Cell Information** to get more information regarding the cause of error.

Invalid Driver Formula Results in Zero

Unlike invalid Reporting and Modeling formulas, invalid Driver formulas do not generate NaNs or red cells. This difference is based on performance and issues related to fact storage. If a Driver formula returns a zero when a different result is expected, it is helpful to modify the formula type to Modeling and select Refresh All in Excel to see if a NaN or red cell is returned. In that case, Cell Information provides more detailed information regarding the reason for the NaN. If the formula is not scoped, the Modeling formula might be too large to run.

Changing the formula type to Reporting might help.

However, keep in mind that the formula computation logic for Driver and Modeling formulas is different than the logic for Reporting formulas, and a NaN might not be encountered as a Reporting Formula.

Selecting **Tools ▶ Contributing Data** for a cell is also helpful in determining the source of the data records. To view, click the cell where a Driver formula has executed. The fact (or facts) which make up the value are displayed, along with the Object_Type (Data load or form set) and Object (Data load ID or Form Set Name).

In the following example, data was loaded from another Model. The value of 3.60 from Data Load was then negated by a Driver formula for (3.60), and the Driver Formula generated a value of 9.00.

ACCOUNT	ANALYSIS	CURRENCY	INTORG	TIME	TRADER	SOURCE	OBJECT_TYPE	OBJECT	Value	USD converted
Previous Period Salaries	Budget	USD	Corporate Legal	Mar 2013	EXT	BaseForm	dataload	6724	3.60	3.60
Previous Period Salaries	Budget	USD	Corporate Legal	Mar 2013	EXT	BaseForm	formset	Salary Expense	9.00	9.00
Previous Period Salaries	Budget	USD	Corporate Legal	Mar 2013	EXT	BaseForm	formset	Salary Expense	(3.60)	3.60
									Total:	9.00

Note that only driver formula results actually generate a fact that is stored and viewable via Contributing Data. Modeling and Reporting formulas are computed on demand. Therefore, their results are not stored as facts.

Troubleshooting Excel-based Calculated Members

Syntax validation for Excel-based calculated members occurs at the time of creation in the Excel Add-in. Since the context of the model is known and referenced at the time these members are created, invalid results typically are limited to the following:

- missing exchange rates
- missing DRATES
- incorrect reference to Dimension code names, Standard and Custom Properties