THE POWER TO KNOW.

SAS® Financial Management 5.1
Formula Guide

SAS® Documentation

The correct bibliographic citation for this manual is as follows: SAS Institute Inc. 2010. *SAS® Financial Management 5.1: Formula Guide*. Cary, NC: SAS Institute Inc.

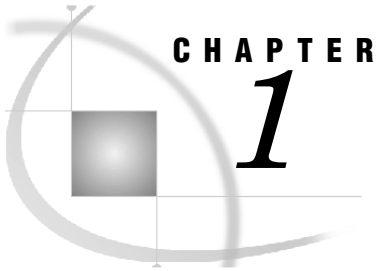
SAS® Financial Management 5.1: Formula Guide

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

1st electronic book, March 2010
2nd electronic book, August 2010

SAS® Publishing provides a complete selection of books and electronic products to help customers use SAS software to its fullest potential. For more information about our e-books, e-learning products, CDs, and hard-copy books, visit the SAS Publishing Web site at **support.sas.com/publishing** or call 1-800-727-3228.

Contents



CHAPTER 1

Introduction

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: User's Guide* or the online Help for each client.

Example Data

The examples provided in this document were performed using SAS Financial Management 5.1. The query results are displayed in the SAS Financial Management Add-In for Microsoft Excel and through Web-based data entry tables.

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
Canada
Admin Canada
HR Canada
IT Support – Canada
Canada Sales
Mexico
Admin Mexico
HR Mexico
IT Support – Mexico
Mexico Sales

ACCOUNT

- **Net Income**
  - **Income Before Taxes**
    - **Operating Expense**
      - **Administrative Expense**
        - Office Supplies
        - Other Administrative Expenses
        - Postage
        - **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 and 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

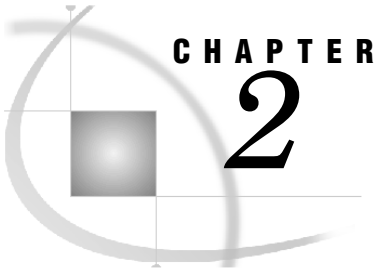
- **Product Total**
  - **Video Games**
    - Action
    - Simulation
    - Arcade
    - Puzzle
  - **Hardware**
    - Game Controller
    - Joy Stick
    - Flight Stick
  - **Publications**
    - How to Use Simulator
    - Arcade Secrets
    - Puzzle Tricks

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**
  - **2008**
    - **Q1 2008**
      - Jan 2008
      - Feb 2008
      - Mar 2008
    - **Q2 2008**
      - Apr 2008
      - May 2008
      - Jun 2008
    - **Q3 2008**
      - Jul 2008
      - Aug 2008
      - Sep 2008
    - **Q4 2008**
      - Oct 2008
      - Nov 2008
      - Dec 2008
  - **2009**
    - **Q1 2009**
      - Jan 2009
      - Feb 2009
      - Mar 2009
    - **Q2 2009**
      - Apr 2009
      - May 2009
      - Jun 2009
    - **Q3 2009**
      - Jul 2009
      - Aug 2009
      - Sep 2009
    - **Q4 2009**
      - Oct 2009
      - Nov 2009
      - Dec 2009



CHAPTER 2

Formula Types

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, meaning 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
- Validity with Operational Planning
- 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, and 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 Member 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.

Validity with Operational Planning

Any reporting formula referenced in a hierarchy for an Operation Planning cycle is ignored in an Operational Planning cycle. This information is provided on the Formula tab in the Model Properties dialog box as well as at the time of Form set publish. The warning message states "Reporting formulas are not supported in operational planning forms." If included on an Operational Planning form set, a calculated member is displayed as gray, non-writeable cells.

For Excel-based calculated members, all functions are available with Operational Planning. See Chapter 5, "Formula Basics for Excel-Based Members" for more information about the different Excel-based calculated member functions.

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 a server-side calculation. 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. 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 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. Also, Cell Information says that the query succeeded.

Dimension Type

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

Valid with Operational Planning

Similar to Reporting formulas, any modeling formulas referenced in a hierarchy for an Operation Planning cycle are ignored in an Operational Planning cycle. This information is provided on the Formula tab in the Model Properties dialog box as well as at the time of Form set publish. The warning message provided states “Modeling formulas are not supported in operational planning forms.” If included on an Operational Planning form set, these calculated members are displayed as gray, non-writeable cells.

Driver formulas are permitted for Operational planning on a limited basis. To be valid for Operational Planning, the formula expression, as well any default members for an expression, can include only members from the Account and Time dimension. Any Driver formulas that reference a member in a dimension other than Account or Time are ignored in an Operational Planning cycle. Information is provided on the Formula tab of the Model Properties dialog box as well as at the time of form set publish. The warning message states “Driver formulas with cross-dimensional references are not supported in operational planning form sets.” If included on an Operational Planning form set, these calculated members are displayed as gray, non-writeable cells.

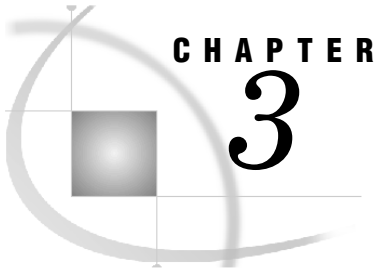
Reuse

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

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 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 a 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 2008	2.0
Feb 2008	2.5
Mar 2008	3.0

Rates

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

Jan 2008	0.90
Feb 2008	0.91
Mar 2008	0.92

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

Jan 2008	0.895
Feb 2008	0.905
Mar 2008	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 2008	Feb 2008	Mar 2008	Q1 2008
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 Expenses	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
Training Expense - XB	1,000.00	1,250.00	1,500.00	1,500.00
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
Cumulative Translation Adjustment	0.00	0.00	0.00	0.00
Average 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 2008 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 2008	Feb 2008	Mar 2008	Q1 2008
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 Expenses	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
Training Expense - XB	1,000.00	1,250.00	1,500.00	1,500.00
Balance Sheet	1,260.00	2,852.50	4,784.50	4,784.50
Stockholder's Equity	1,260.00	2,852.50	4,784.50	4,784.50
Retained Earnings	1,260.00	2,852.50	4,784.50	4,784.50
Cumulative Translation Adjustment	(7.00)	(1.75)	19.25	19.25
Average 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 2008
    (1400.00 * 0.895) - (1400.00 * 0.90) = 1253.00 - 1260.00 = (7.00)
  - Feb 2008
    ((1400+1750) * 0.905) - ((1400 * 0.9) + (1750 * 0.91)) = 2850.75 -2852.50 = (1.75)
  - Mar 2008 and Q1 2008
    ((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

In versions before 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.

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 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 2008	100
Jan 2009	200
Feb 2009	250
Mar 2009	400

Rates

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

Dec 2008	0.85
Jan 2009	0.86
Feb 2009	0.88
Mar 2009	0.90

Step 1

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

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

	Dec 2008	Jan 2009	Feb 2009	Mar 2009	Q1 2009
Accounts Receivable	117.65	232.56	284.09	444.44	444.44
Doubtful Accounts %	(0.25)	(0.25)	(0.25)	(0.25)	(0.25)

Step 2

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

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

	Dec 2008	Jan 2009	Feb 2009	Mar 2009	Q1 2009
Accounts Receivable	117.65	232.56	284.09	444.44	444.44
Doubtful Accounts %	(0.25)	(0.25)	(0.25)	(0.25)	(0.25)
Allowance for Doubtful Accounts	(29.41)	(58.14)	(71.02)	(111.11)	(111.11)

Step 3

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

Organization	Admin Canada
Product	Product Total
Customer	Customer Total
Analysis	Budget
Currency	CAD
Frequency	PA

	Dec 2008	Jan 2009	Feb 2009	Mar 2009	Q1 2009
Accounts Receivable	117.65	116.28	56.82	166.67	333.33
Doubtful Accounts %	(0.25)	0.00	0.00	0.00	0.00
Allowance for Doubtful Accounts	(29.41)	(28.73)	(12.88)	(40.09)	(81.70)

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	Admin Canada
Product	Product Total
Customer	Customer Total
Analysis	Budget
Currency	CAD
Accounts Receivable	Allowance for Doubtful Accounts

	Dec 2008	Jan 2009	Feb 2009	Mar 2009	Q1 2009
PTD	(25.00)	(50.00)	(62.50)	(100.00)	(100.00)
LTD	(25.00)	(50.00)	(62.50)	(100.00)	(100.00)
YTD	(25.00)	(50.00)	(62.50)	(100.00)	(100.00)
QTD	(25.00)	(50.00)	(62.50)	(100.00)	(100.00)
MTD	(25.00)	(50.00)	(62.50)	(100.00)	
PA	(25.00)	(24.71)	(11.34)	(36.08)	(73.53)
LA	(25.00)	(50.00)	(62.50)	(100.00)	(100.00)
YA	(25.00)	(24.71)	(36.62)	(73.53)	(73.53)
QA	(25.00)	(24.71)	(36.62)	(73.53)	(73.53)
MA	(25.00)	(24.71)	(11.34)	(36.08)	

Reporting Currency Approach

The reporting currency approach is used by reporting formulas and Excel-based calculated members only. There are basically 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 ...	Puzzle
Customer ...	Buy Best
Analysis ...	Budget
Currency ...	CAD
Frequency ...	YTD

	Dec 2008	Jan 2009	Feb 2009	Mar 2009	Q1 2009
Gross Margin	(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 2008	0.860
Jan 2009	0.855
Feb 2009	0.870
Mar 2009	0.890

Step 1

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

Organization	Canada Sales
Product	Puzzle
Customer	Buy Best
Analysis	Budget
Currency	USD
Frequency	YTD

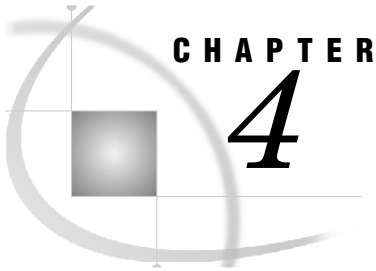
	Dec 2008	Jan 2009	Feb 2009	Mar 2009	Q1 2009
Gross Margin	(30,100.00)	(1,539.00)	(3,583.50)	(6,342.50)	(6,342.50)
Sales	(77,400.00)	(4,275.00)	(9,495.00)	(17,060.00)	(17,060.00)
Cost of Sales	47,300.00	2,736.00	5,911.50	10,717.50	10,717.50

Step 2

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

Organization	Canada Sales
Product	Puzzle
Customer	Buy Best
Analysis	Budget
Currency	USD
Frequency	YTD

	Dec 2008	Jan 2009	Feb 2009	Mar 2009	Q1 2009
Gross Margin	(30,100.00)	(1,539.00)	(3,583.50)	(6,342.50)	(6,342.50)
Sales	(77,400.00)	(4,275.00)	(9,495.00)	(17,060.00)	(17,060.00)
Cost of Sales	47,300.00	2,736.00	5,911.50	10,717.50	10,717.50
Gross Margin Percentage	0.39	0.36	0.38	0.37	0.37



CHAPTER 4

Formula Basics for Server-Side Calculated Members

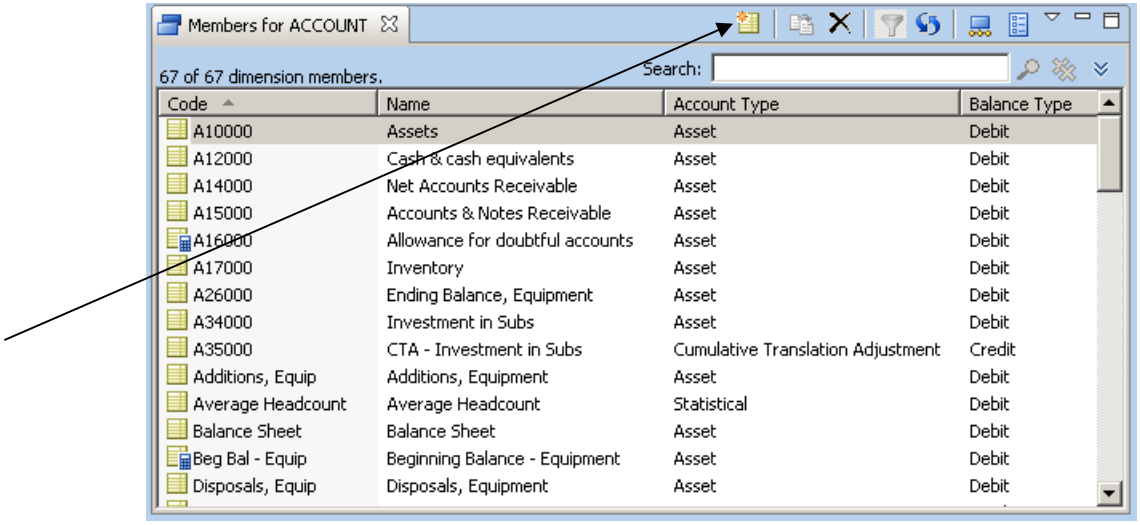
Introduction

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

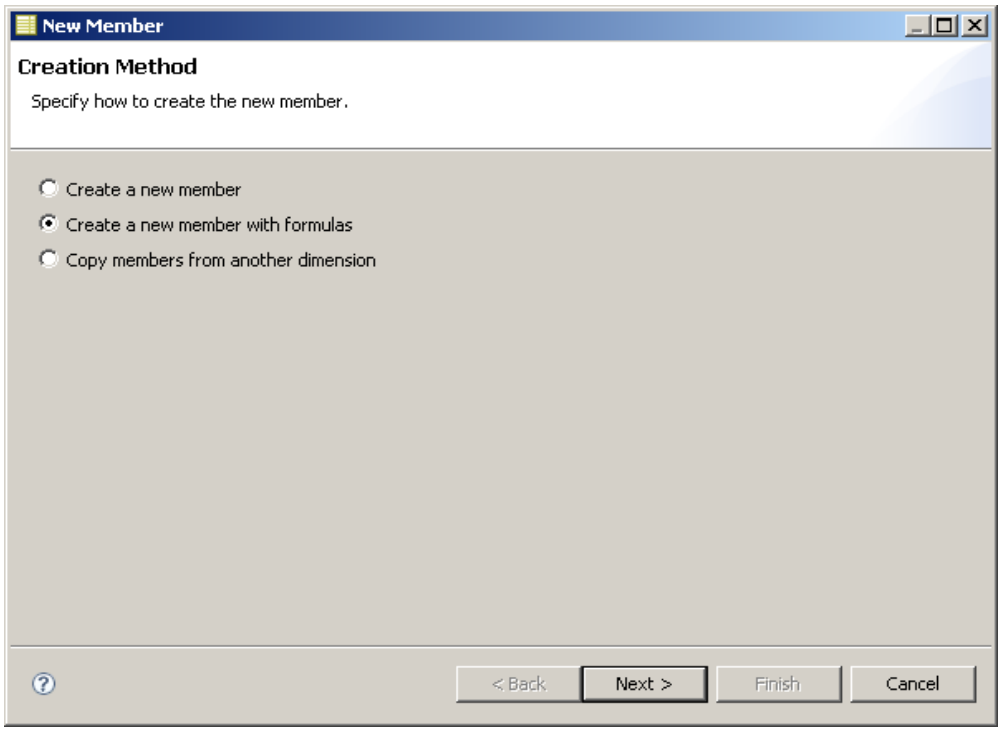
Creating a Server-Side Calculated Member

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.

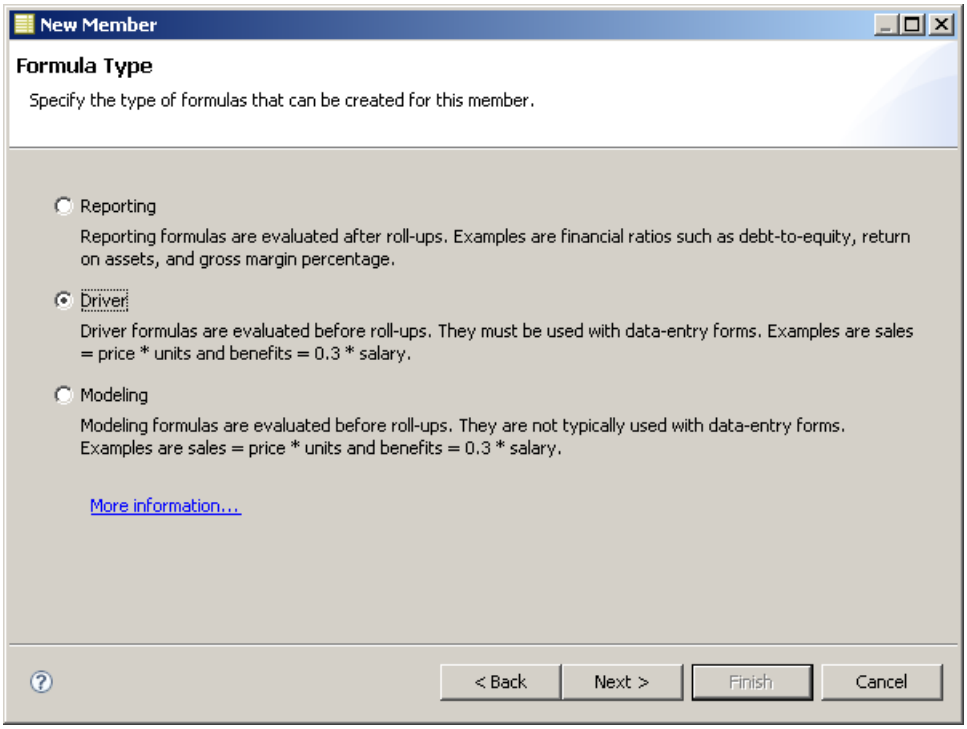
1 To create a new member in the Account dimension, click **`Add a member to this hierarchy`** on the **`Hierarchies`** tab or **`Create a new member`** on the **`Members`** tab (shown below).



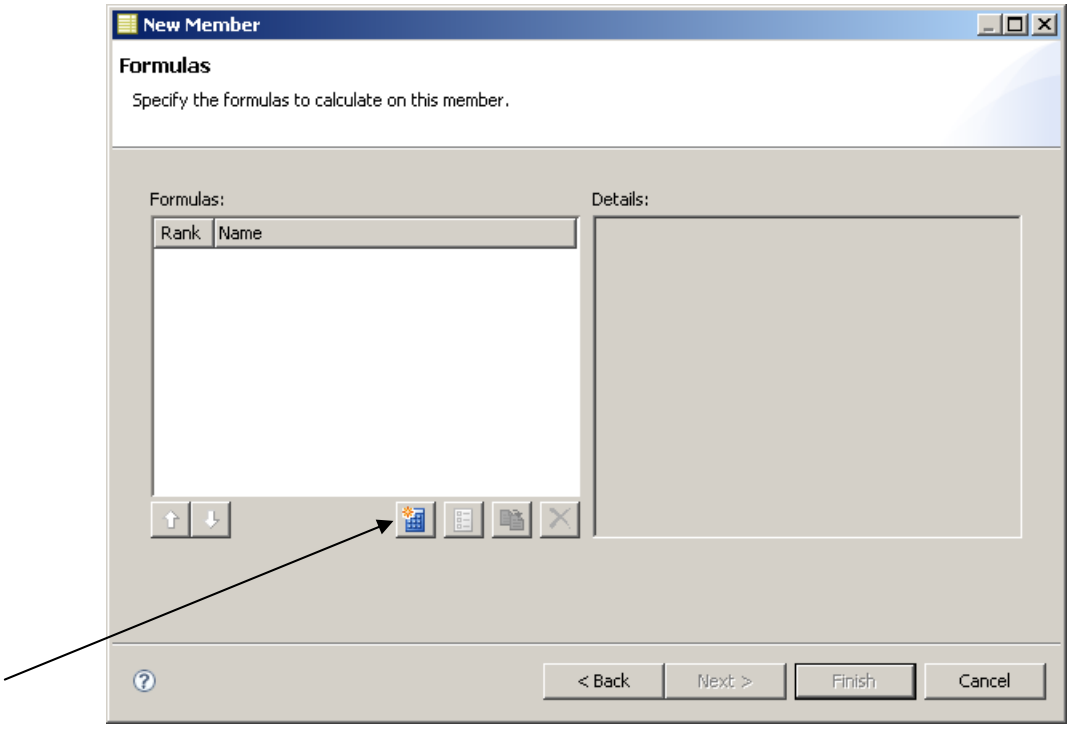
2 Select **`Create a new member with formulas`** and click **`Next`**.



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

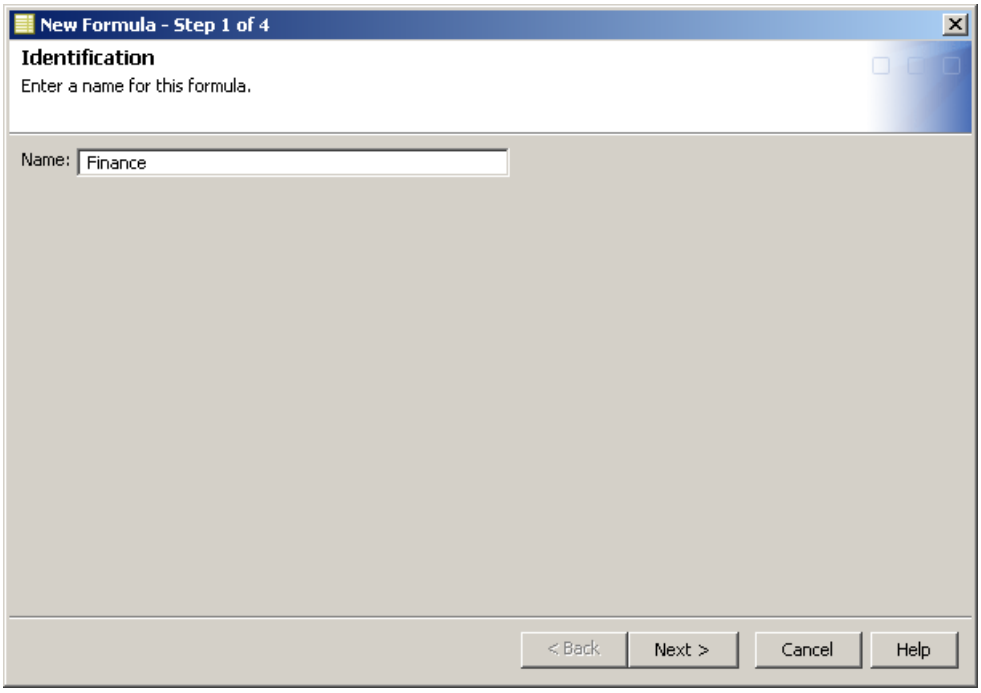


4 On the Formulas page, click the **`Create a new formula`** icon.

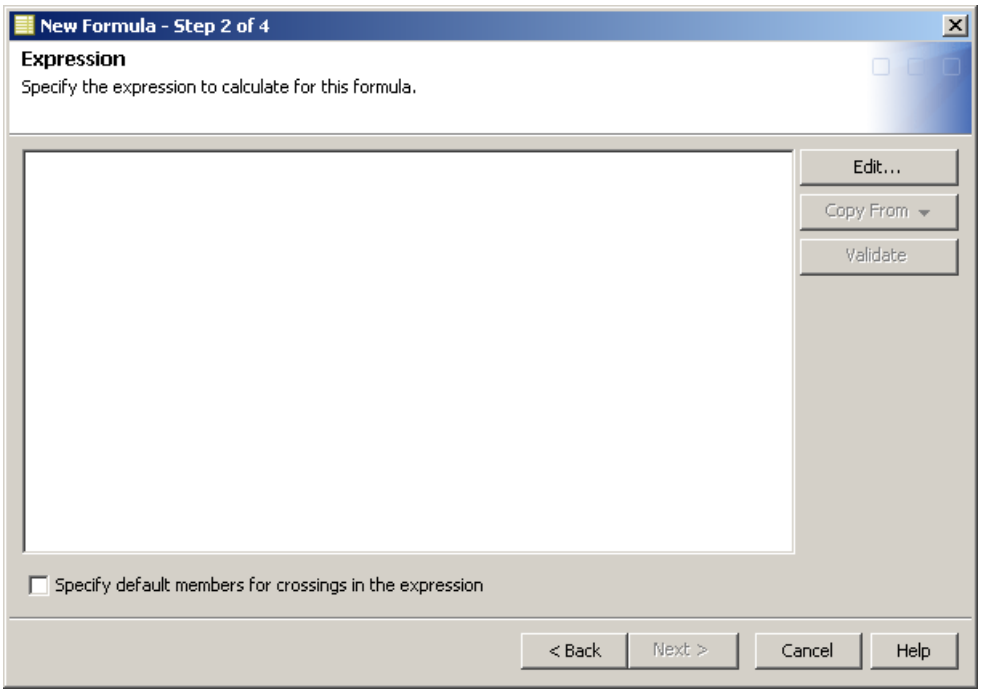


5 On the Identification page, name the formula and click **`Next`**.

The name given here is displayed via **`Cell Information`** in the Excel Add-In.



6 On the Expression page, click **`Edit`** to open the Formula Editor window.



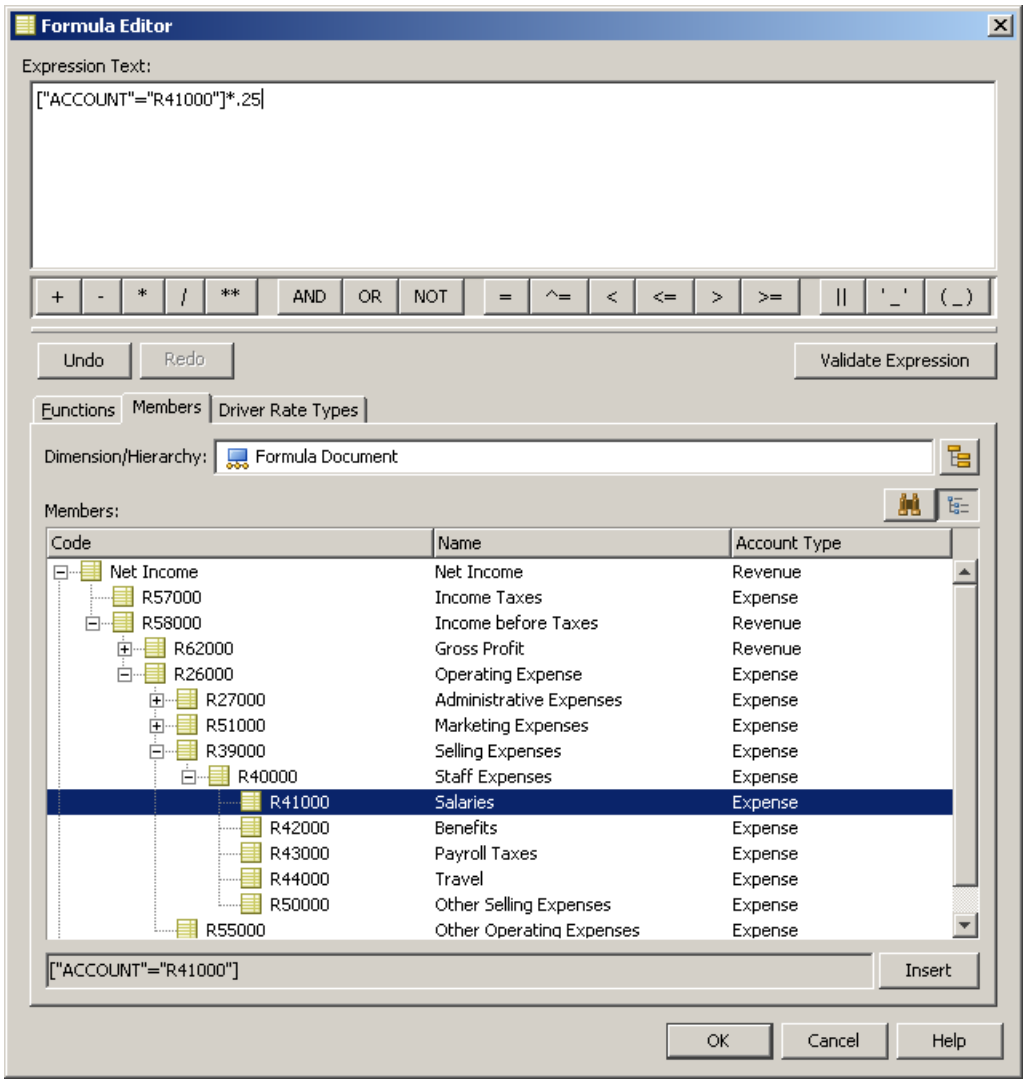
The Formula Editor window has three tabs for creating and managing formula expressions:

- 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.

7 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.

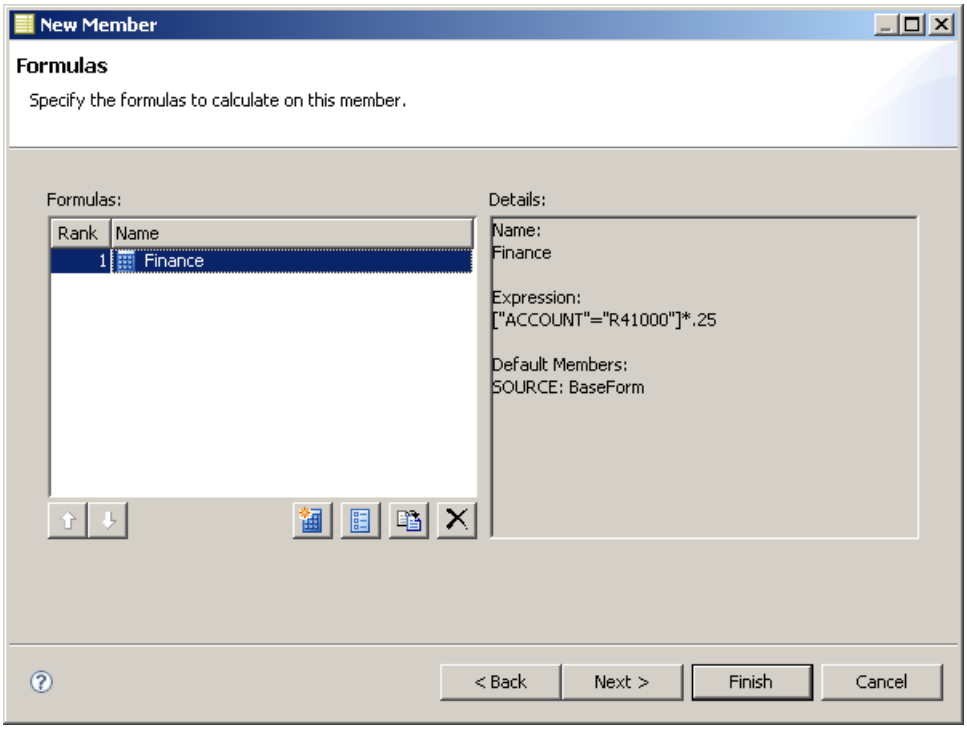


8 Click **`Validate Expression`** or **`OK`** to validate that the expression is correct.

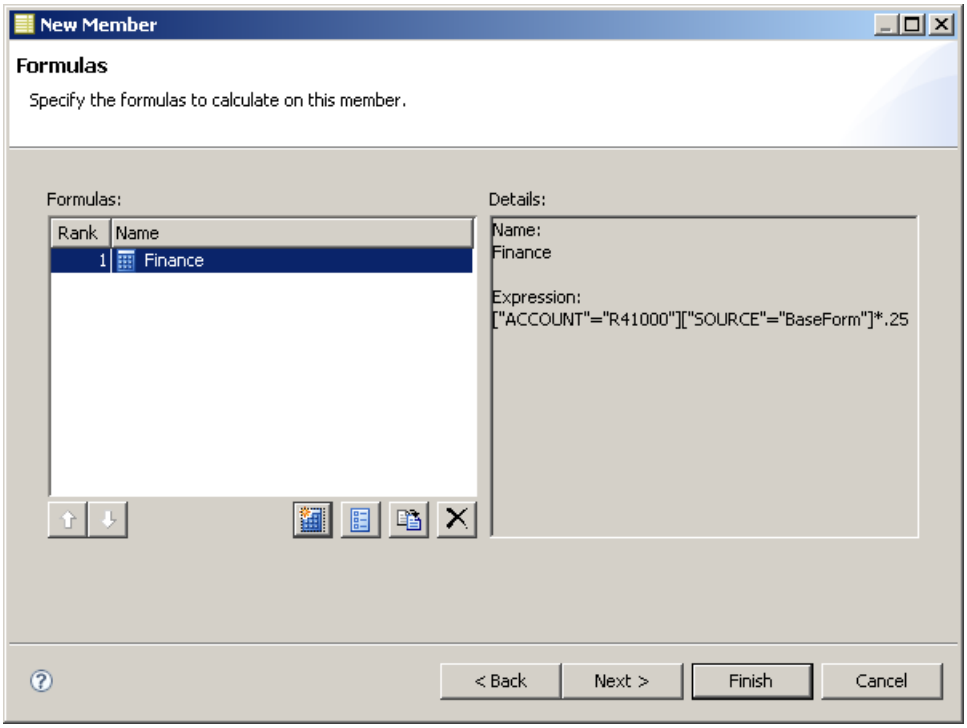
Upon validation, you can assign default members to the formula expression. Default members are considered an extension of a formula expression. The selection of a default member is applied universally to all of the formula inputs for the current formula expression.

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 default members:

Option 1: With Default Members

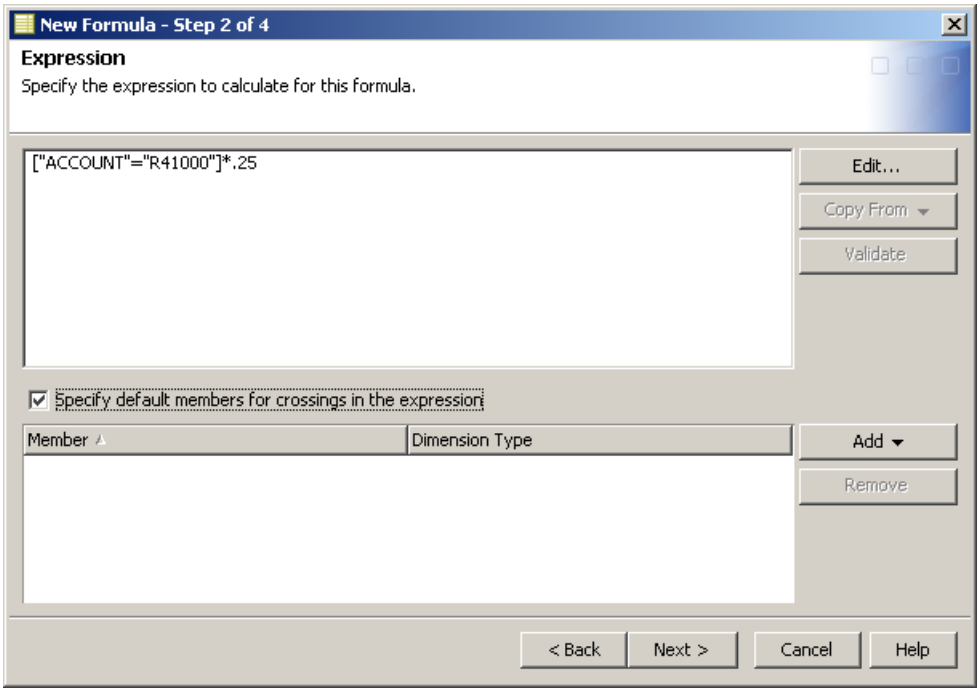


Option 2: Without Default Members



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

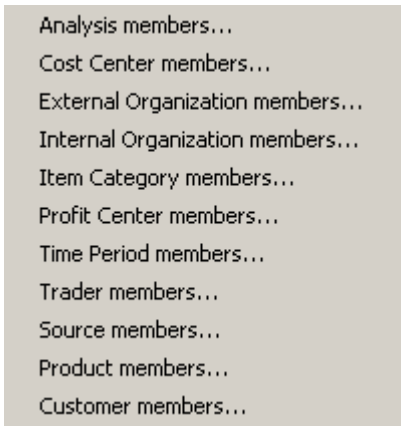
9 To use default members, select the check box labeled **`Specify default members for crossings in the expression`**.



10 Click **Add**.

A pop-up list of available dimension types appears.

11 Select the desired dimension type by clicking it in the list.

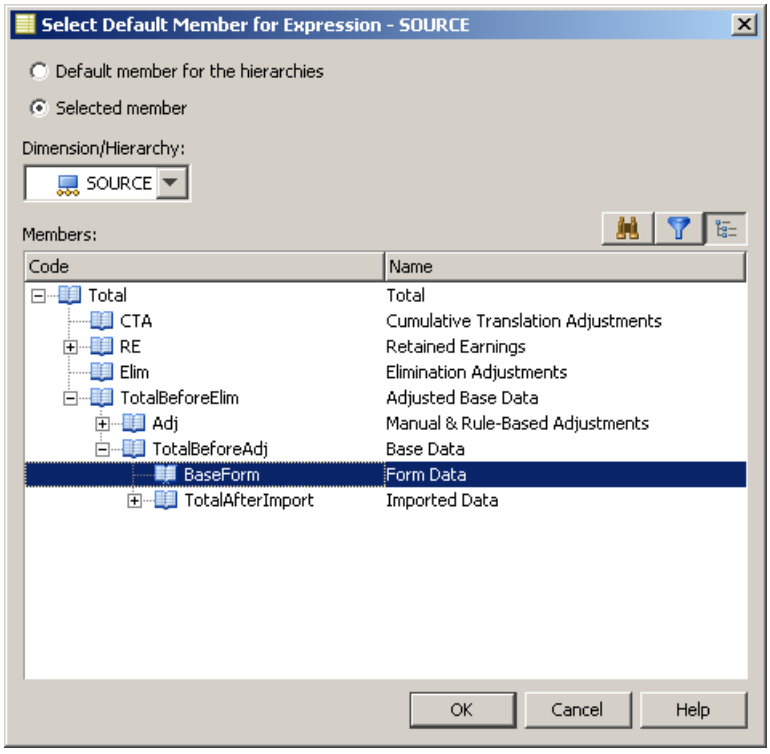


Members are displayed in the context of the current view of the selected dimension and hierarchy.

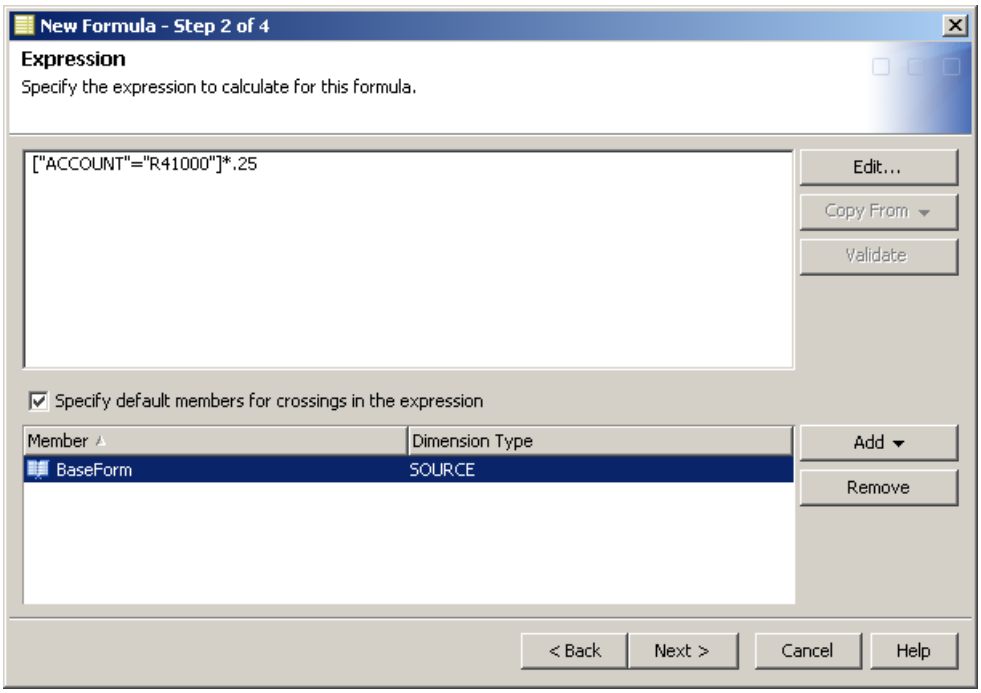
12 Select the default member of a given hierarchy or a different member.

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

13 Highlight the desired member and click **OK**.

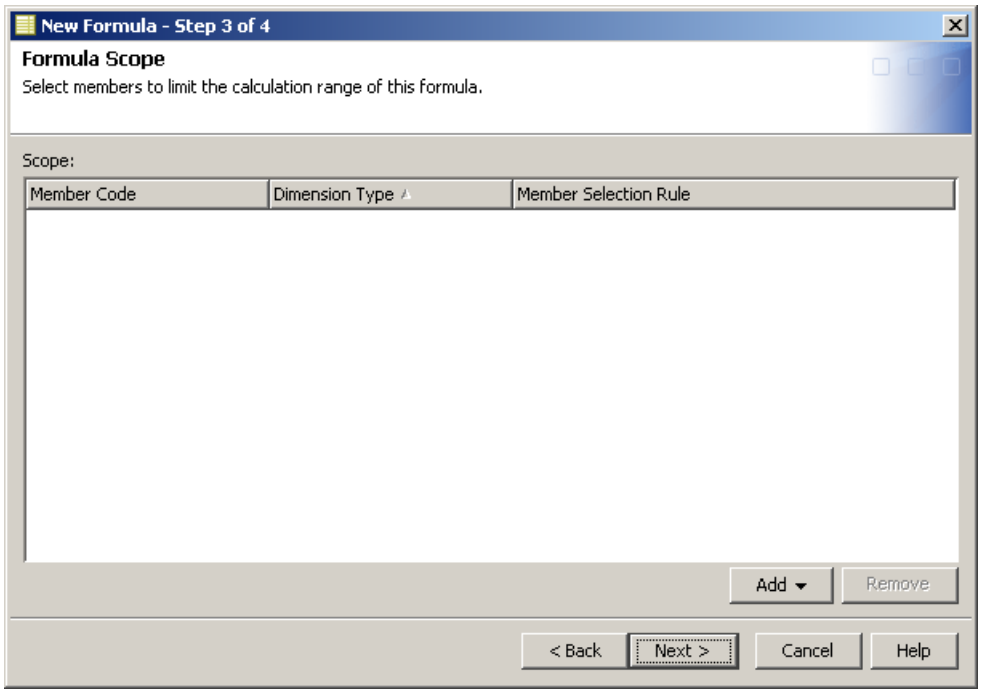


The default member that you selected is displayed on the Expression page. For more information about the use of default members with calculated members, see Chapter 6, “Advanced Formula Concepts.”



14 Click **`Next`** to advance to the Formula Scope page.

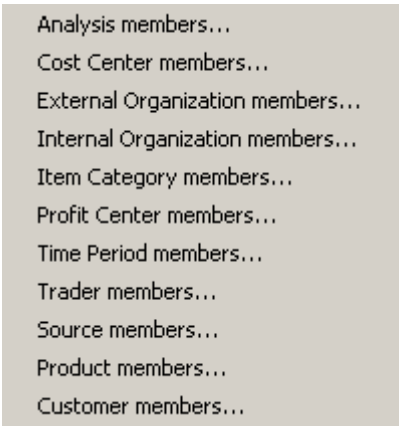
By specifying a scope for a formula, you can restrict the set of crossings where the formula runs. Depending on the formula type, this can improve formula performance. It is also important when you define multiple formulas on the same member. For details about defining multiple formulas on a member,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



15 Click **Add**.

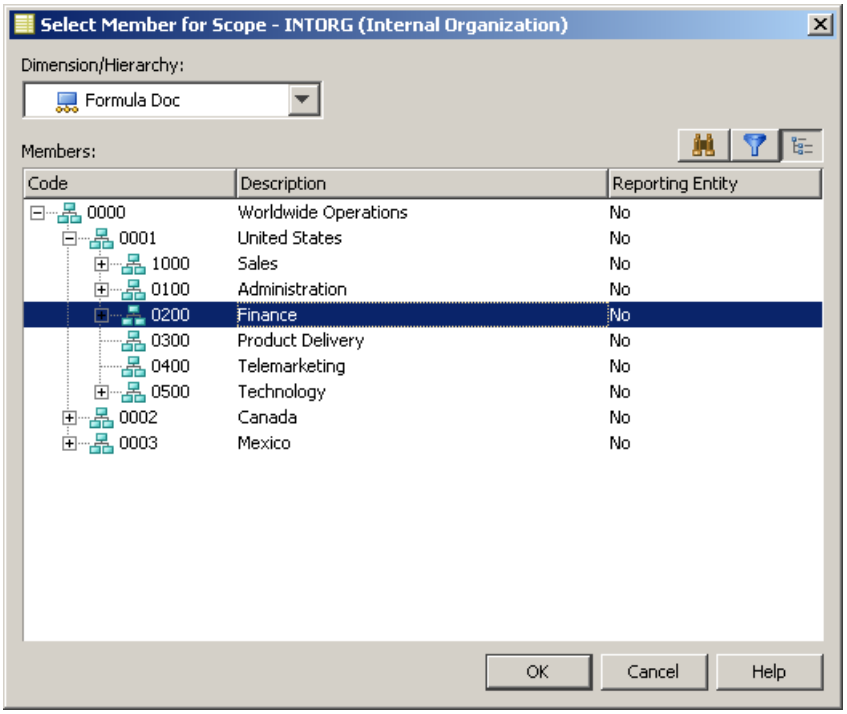
A pop-up list of available dimension types appears.

16 Click the desired dimension type in the list.



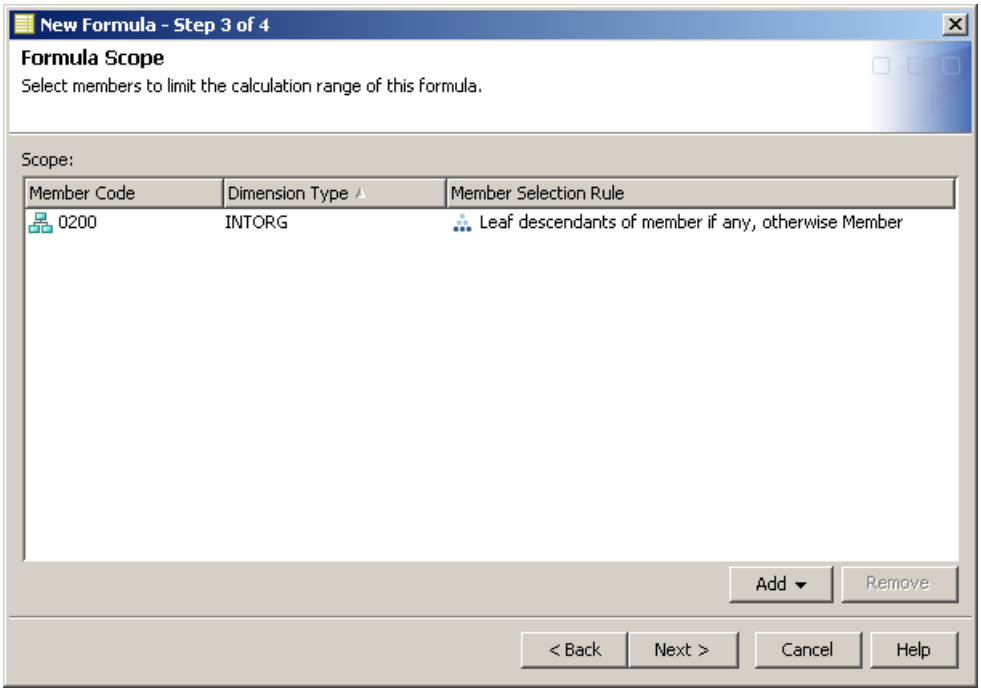
Members are displayed in the context of the current view of the selected dimension and hierarchy. You can select from the available hierarchies and choose multiple members in each dimension.

17 Highlight the desired member(s) and click **OK**.



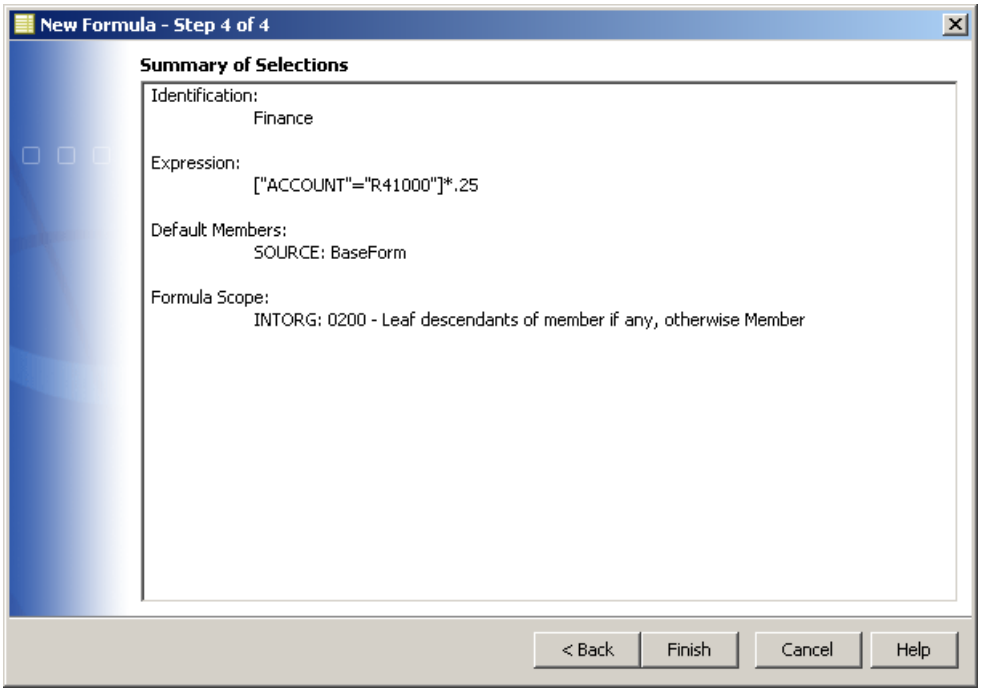
The results of specifying a scope for the formula are displayed on the Formula Scope page. You can specify additional members.

18 Click **`Next`** when the formula scope specification is complete.



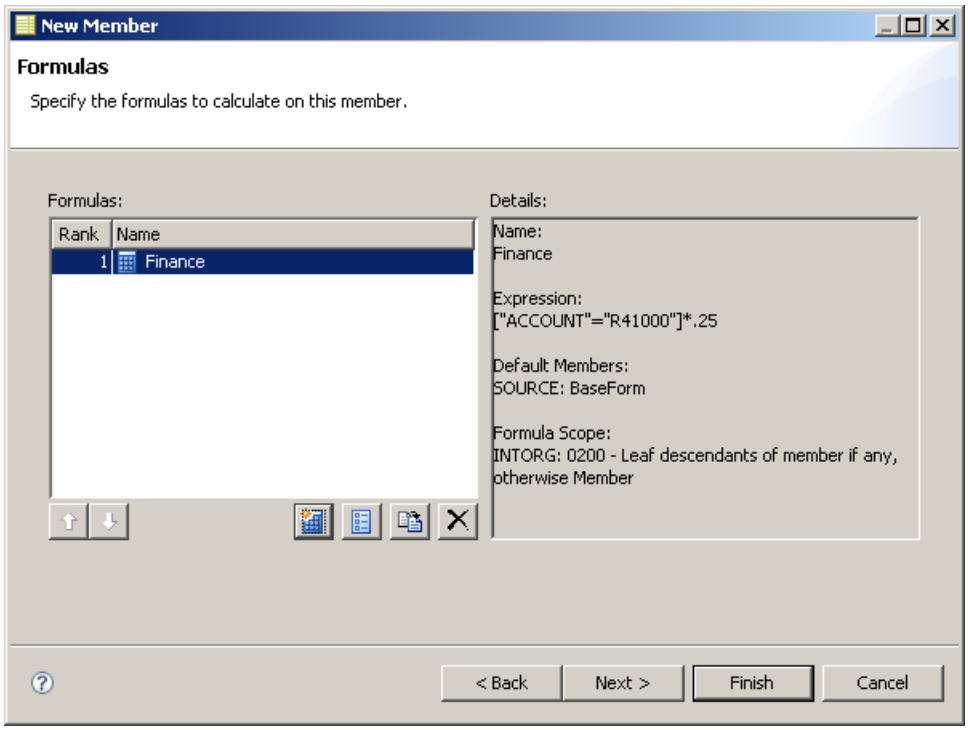
A summary window shows the expression text, default members, and scope of the formula that you have created.

19 Click **`Finish`** if the information is correct.

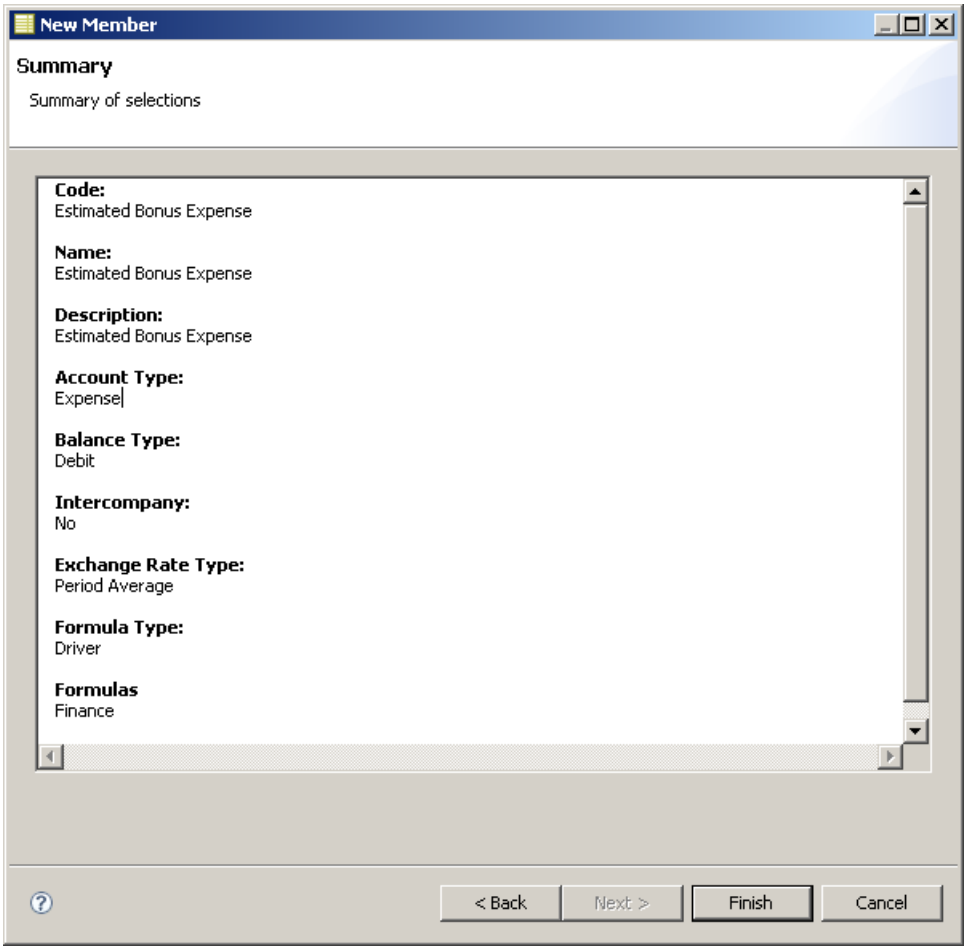


This completes the creation of a single formula on a member. At this point, you have not yet completed all the required steps for the addition of the calculated member. You are returned to the Formulas page, where you can create additional formulas, edit existing formulas, or view the properties of existing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

Repeat steps 3 (page 25) through 18 to create multiple formulas on the same member. When you are finished creating and editing formulas, click **Next**.



The following page displays the summary information for the calculated member. This page shows the additional property information assigned to the member as well as the formula name(s).

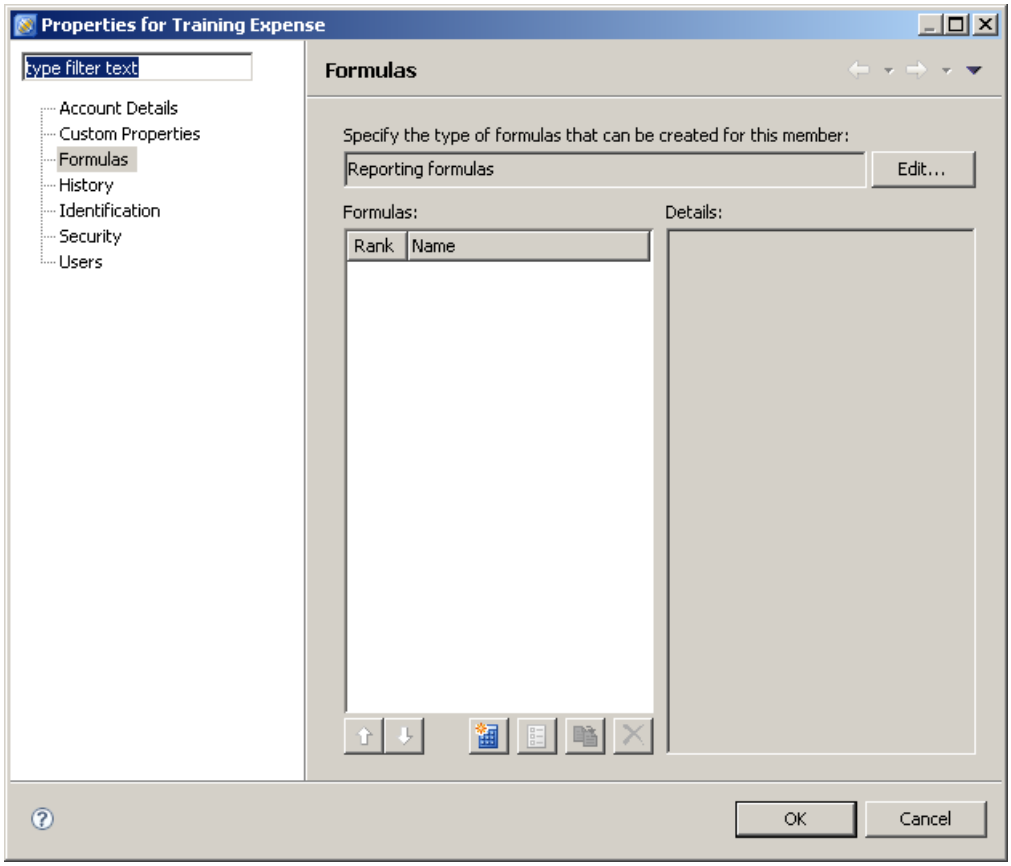


20 Click **`Finish`**. This completes the creation of the new member with formula(s).

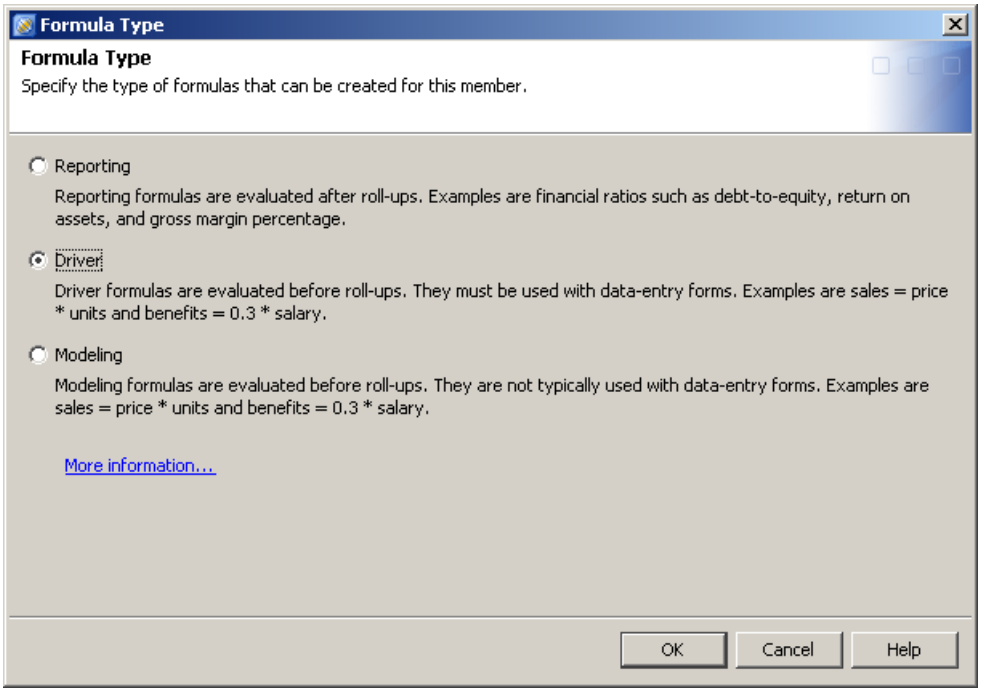
Adding Formulas to a Member

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.

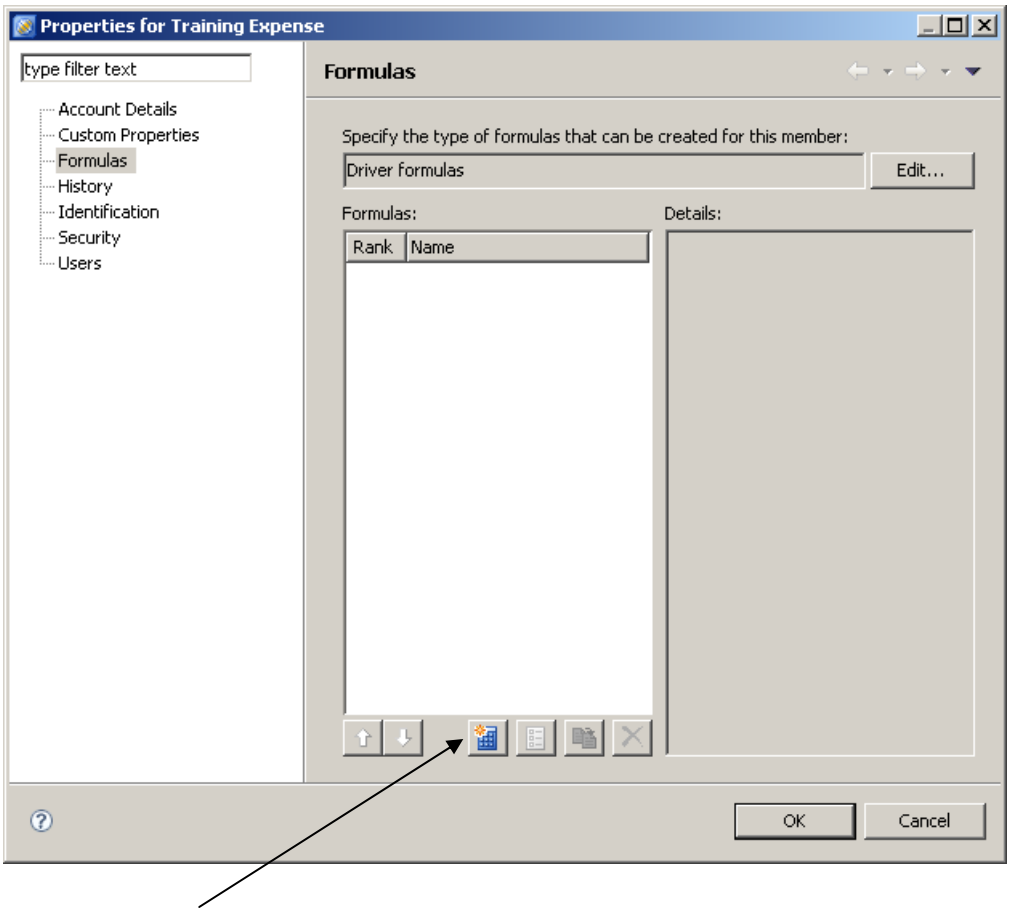
To add a formula to an existing member, highlight the member and either double-click the member or select the Show properties icon. On the Formulas page, the Reporting formula type is selected as the default. To select a different formula type, click **`Edit`**.



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



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



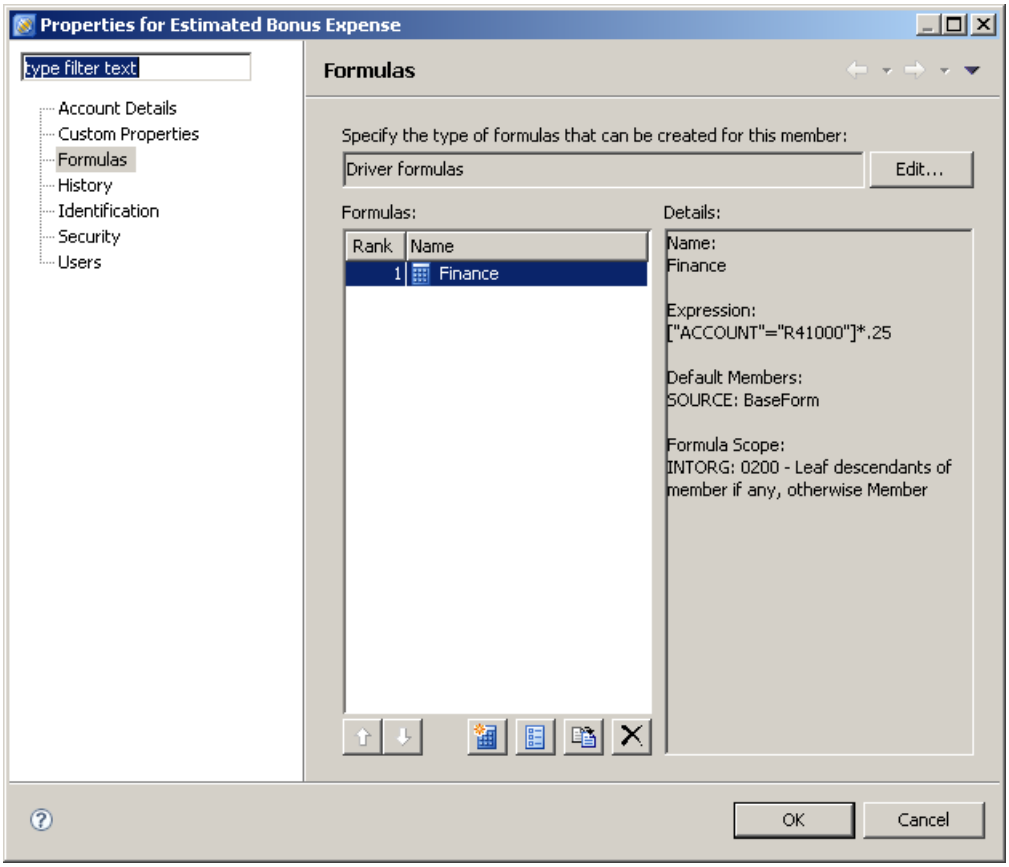
3 See steps 3 (page 25) through 18 in the preceding section for detailed information about creating formulas.

Removing Formulas from a Member

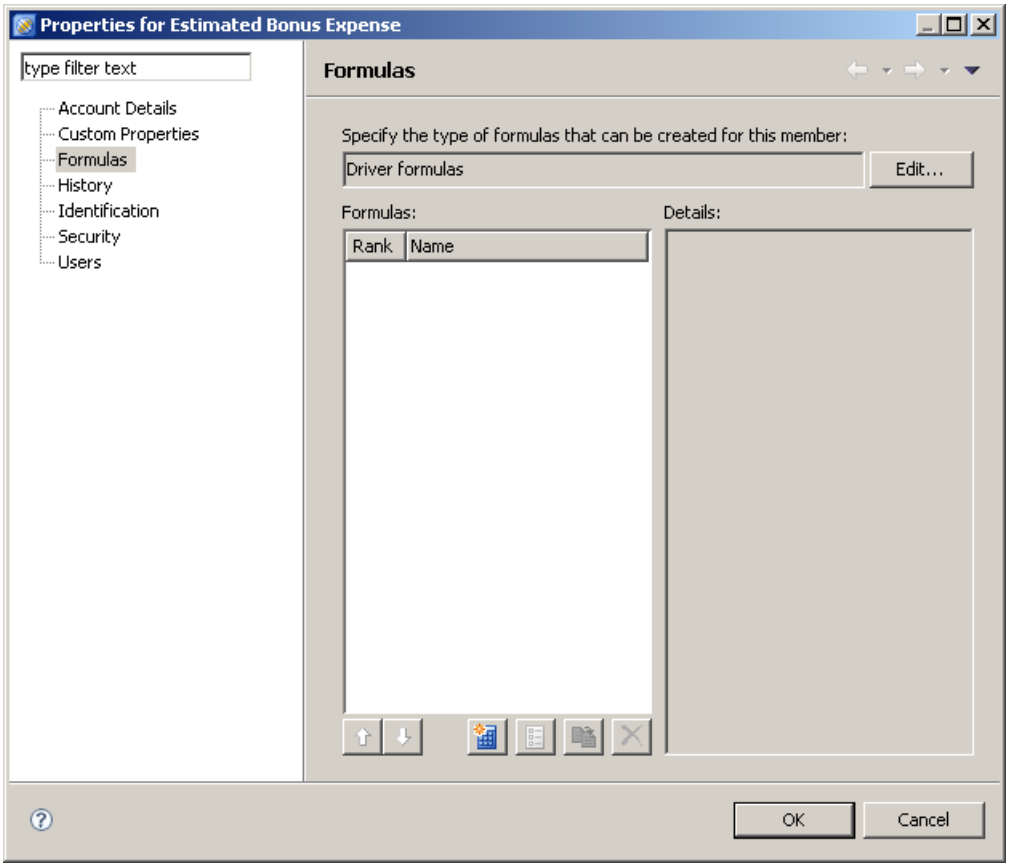
The following steps show how to remove a formula from a member of the Account dimension.

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.

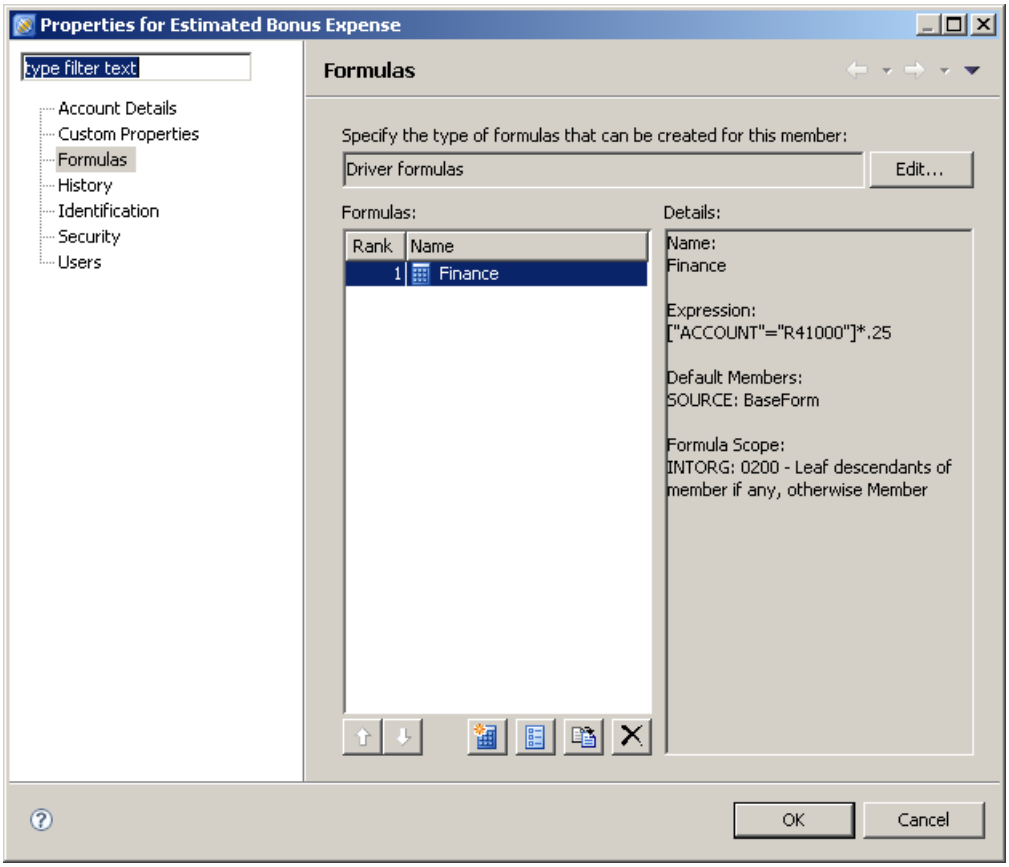
Note: If you delete a driver formula for which results have been computed and stored, the stored results are not deleted. To delete the stored results, select the **Run driver formulas for this form set** option for the appropriate form set.

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
- Default members
- Formula 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 will still use 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:

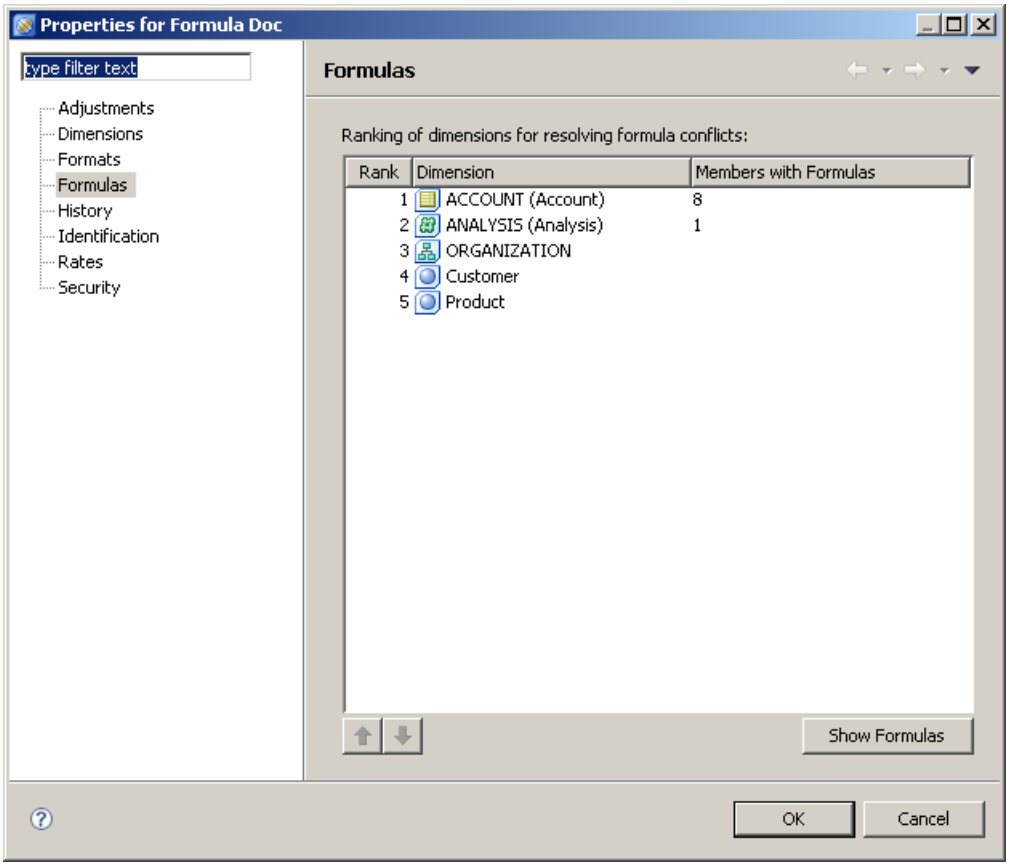
- Balance account types aggregate and then convert results.
- Flow account types convert and then aggregate results.
- Statistical accounts do not participate in aggregation or conversion.

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

Resolving Conflicts between Dimensions

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

We follow a scenario for each expression on the following pages.

Scenario 1: Account Dimension Has Higher Rank than Analysis Dimension

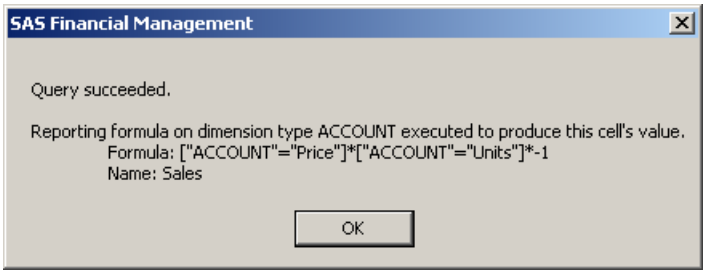
The Account dimension is ranked higher than the Analysis dimension, as shown in the preceding window. This means that the Account formula runs at the crossing where the formulas conflict, as shown here:

ORGANIZATION	Canada Sales
Product ...	Simulation
Customer ...	Westco
Currency ...	CAD
FREQUENCY ...	PTD
Time ...	Jul 2009
Source ...	Form Data
ORGANIZATION_TRADER ...	EXT

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

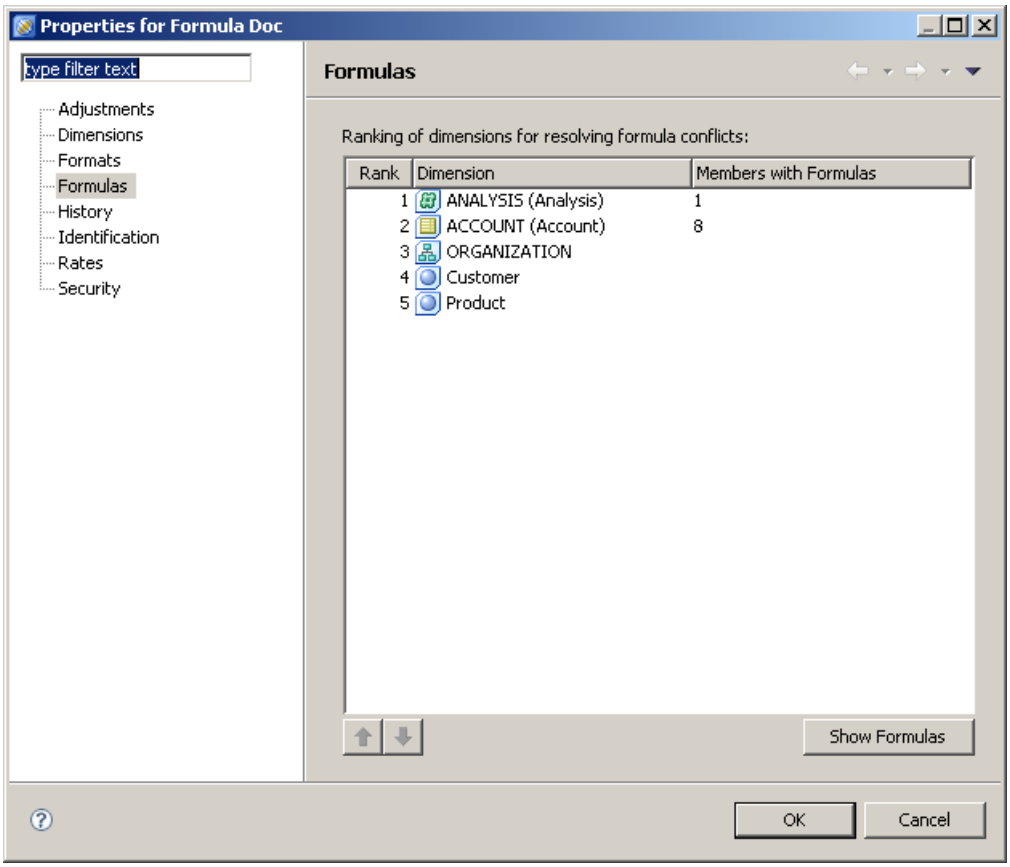
With the Account dimension ranked first, Variance for Sales is computed as 0 * 20 * -1 = 0.

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:



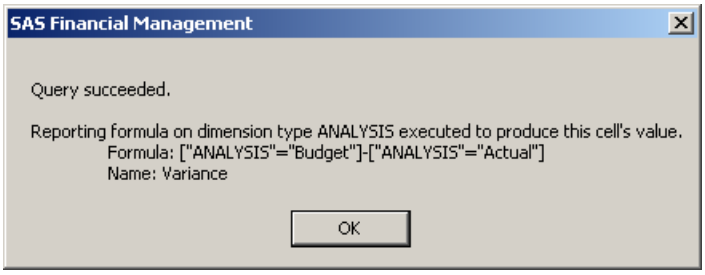
The resulting read-only table appears as follows:

ORGANIZATION	Canada Sales
Product	Simulation
Customer	Westco
Currency	CAD
FREQUENCY	PTD
Time	Jul 2009
Source	Form Data
ORGANIZATION_TRADER	EXT

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

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:

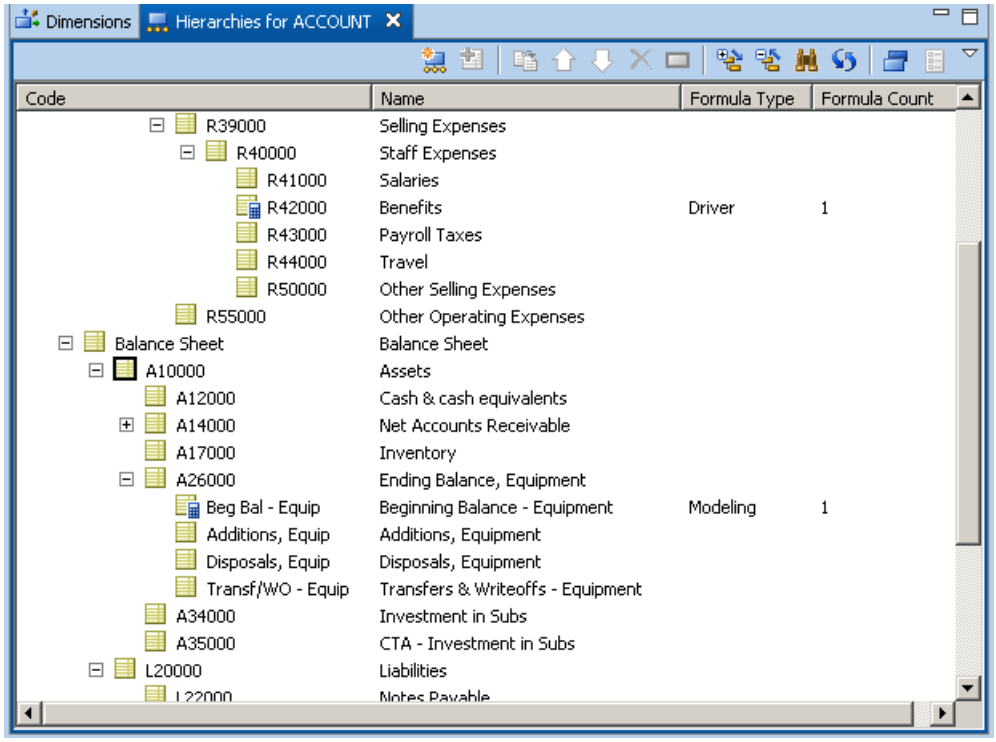
- IntOrg
- Account
- Custom

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.

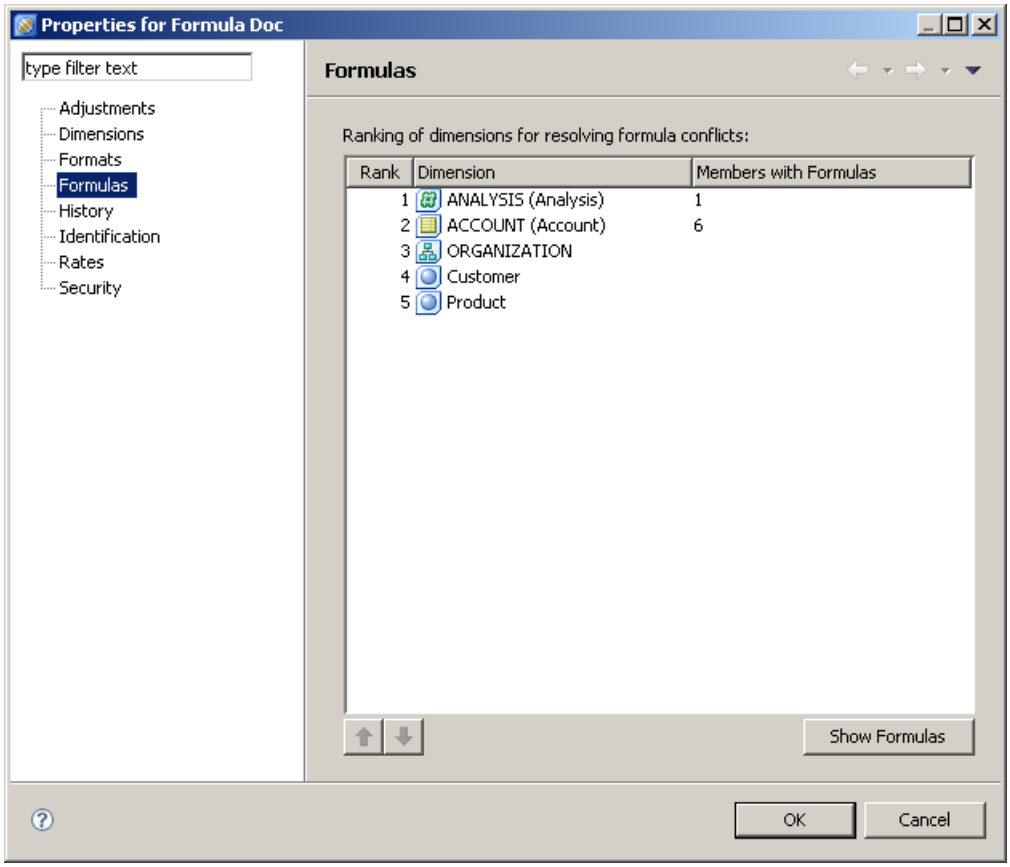
Viewing Formula Information in SAS Financial Management Studio

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.”

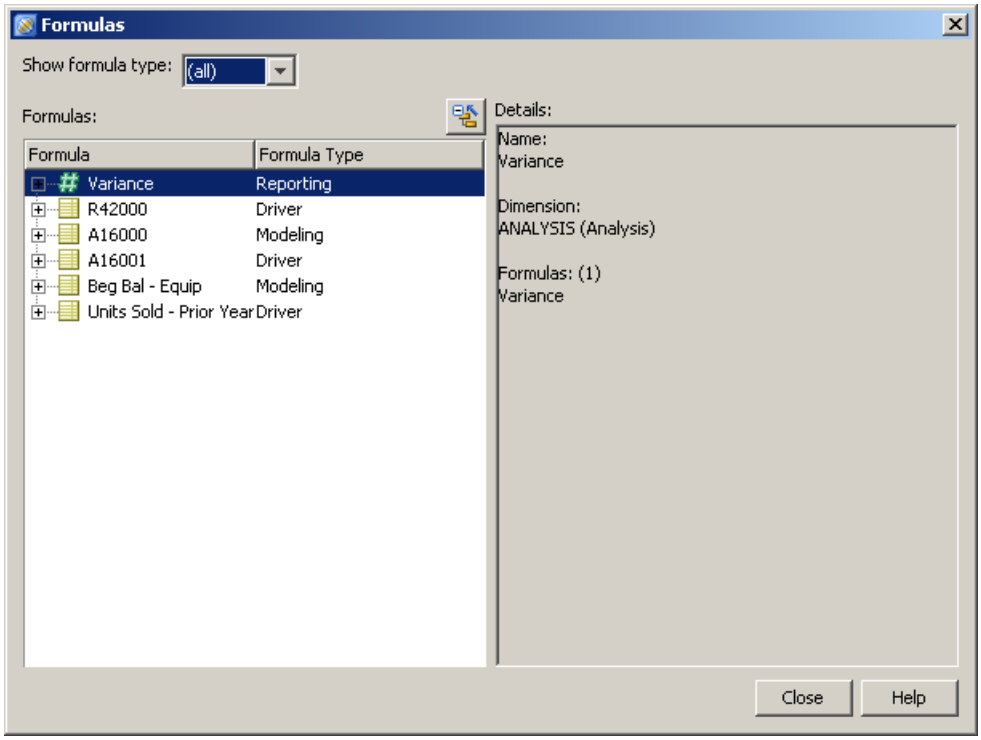
The following window illustrates the optional headers displayed for the Account dimension type:



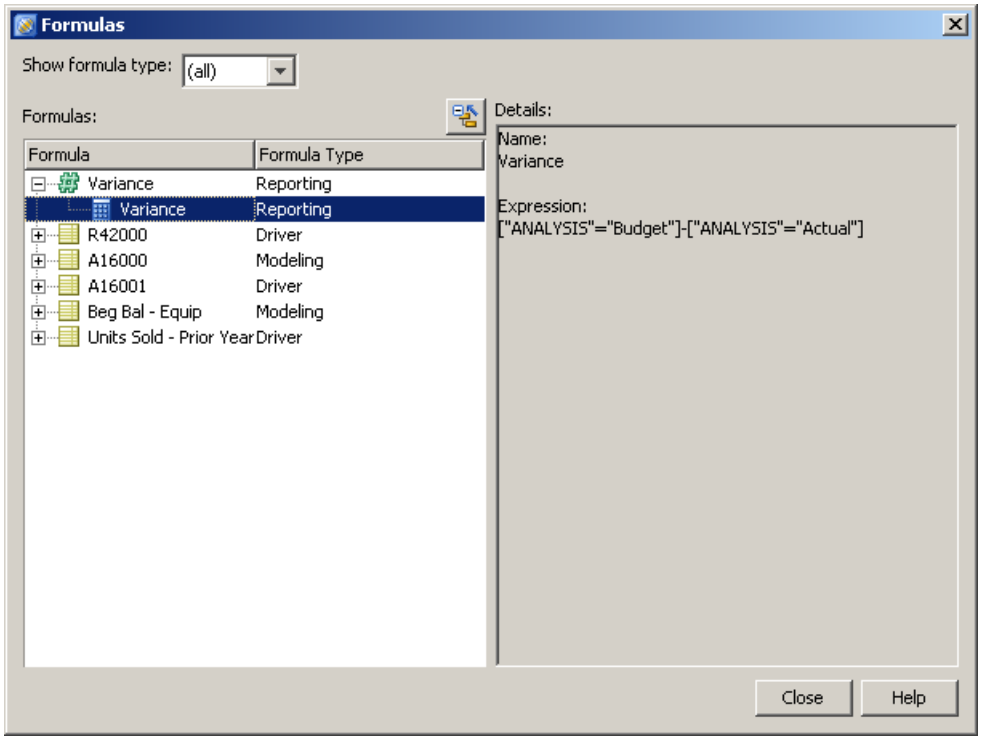
In the Models workspace, Model Properties displays the number of members with formulas on the Formulas page for each dimension in the model, based on the selected hierarchy and as-of-date.



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 default member information is also included in the Details window.

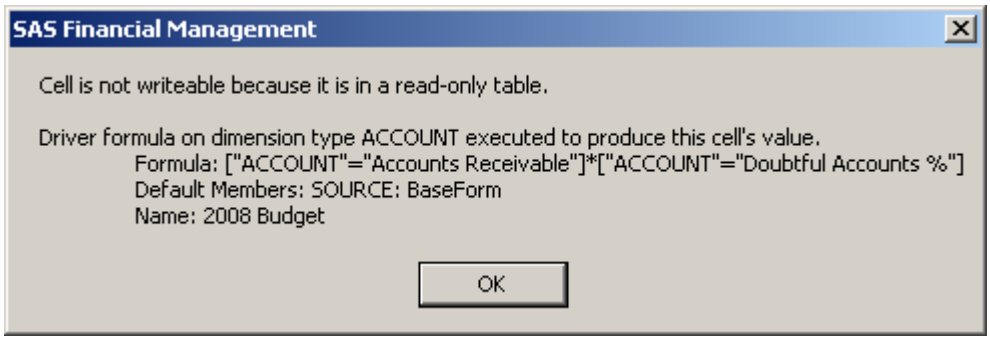


Viewing Formula Information in Excel

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 Solutions** menu to view formula information. The following formula information is displayed:

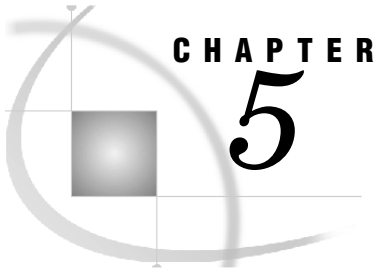
- Formula type
- Dimension of calculated member
- Formula expression
- Name of formula
- Default members, if any

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

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:

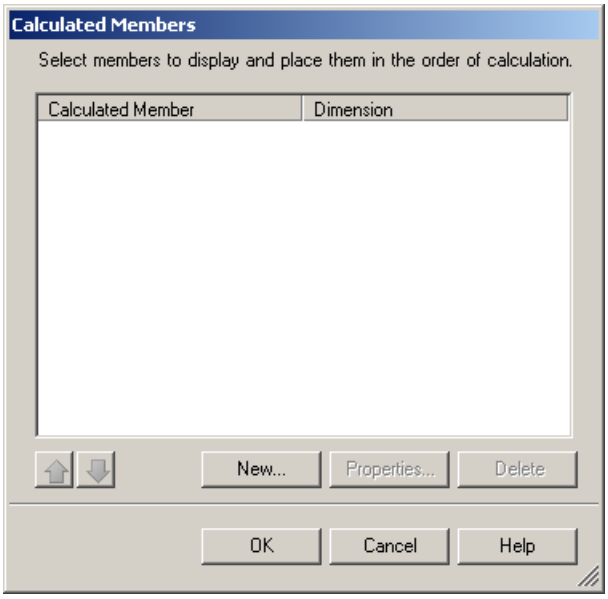
- reference to any member in the same dimension
- absolute reference to any crossing in a table in the same workbook
- absolute reference to any cell in the same workbook
- any Excel function or valid Excel expression

Creating an Excel-Based Calculated Member

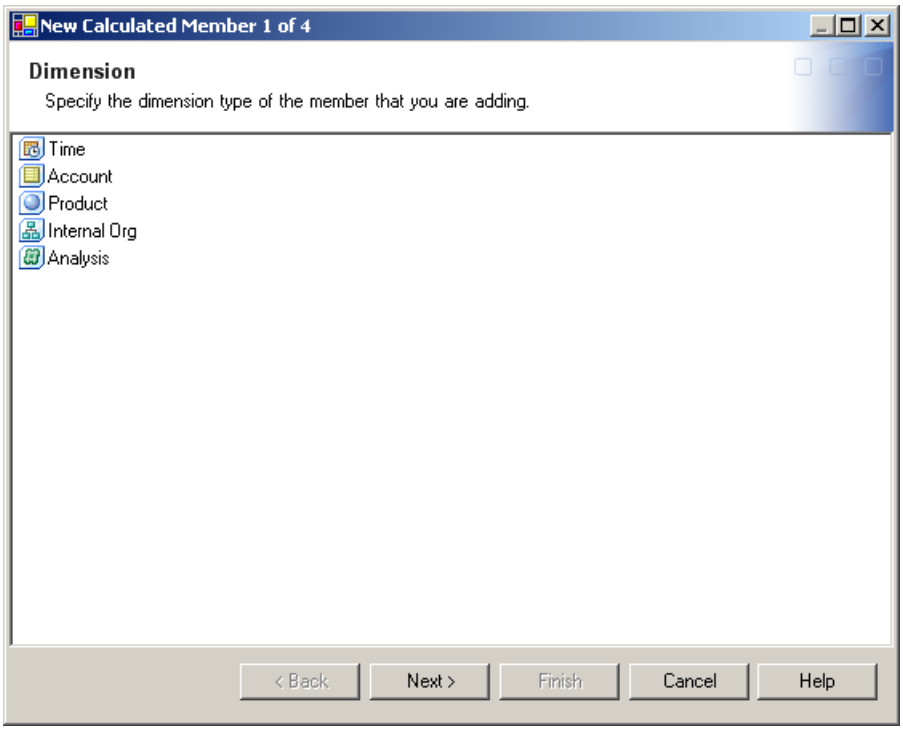
The following steps show how to create an Excel-based calculated member in the Account dimension.

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

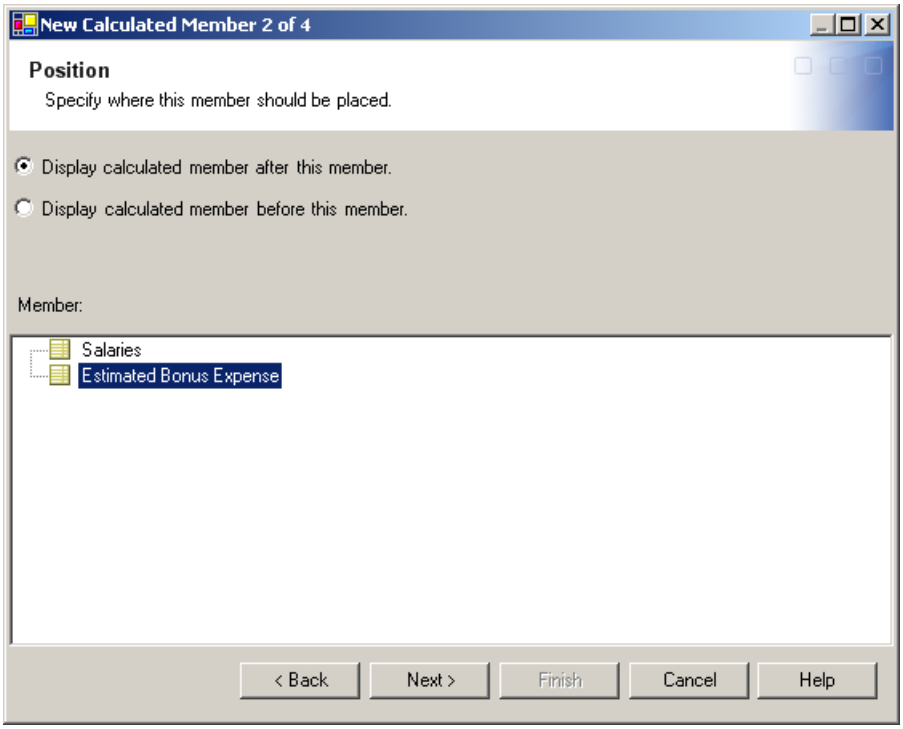
The following window appears:



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

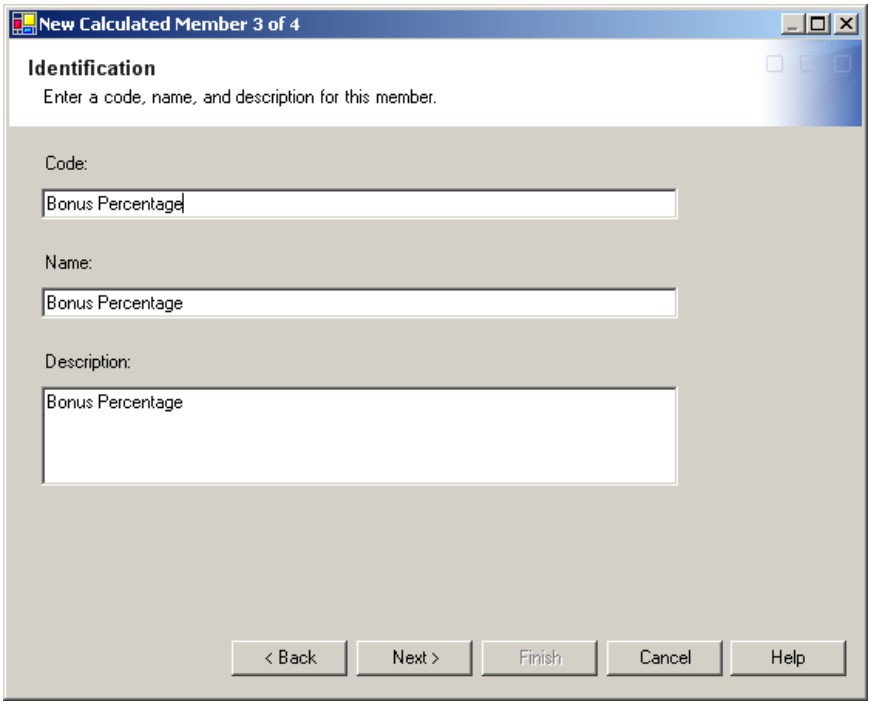


The Position page displays the hierarchical view of the current members selected for display in the table. To determine the placement of the calculated member, highlight an available member and choose whether to display the calculated member before or after the selected member.



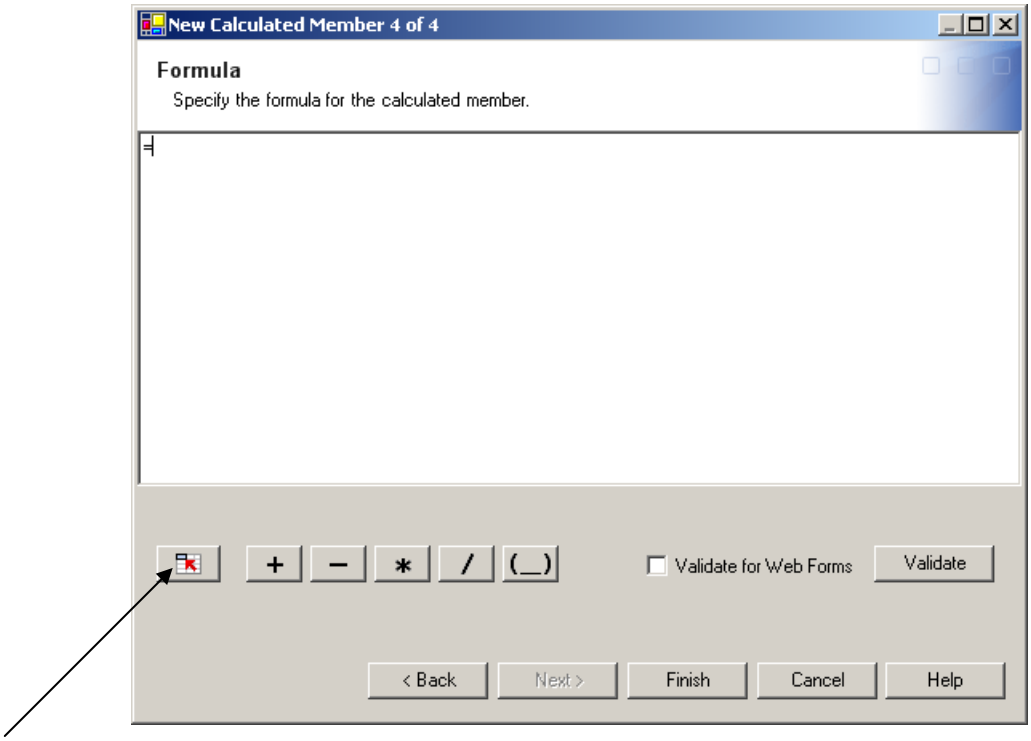
3 Click **Next** to advance.

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



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

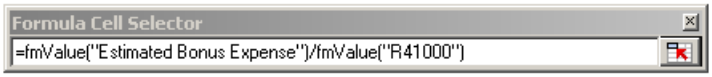
Using the Formula Cell Selector (the icon in the lower left) is an easy way to select dimension members and cell references.



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.

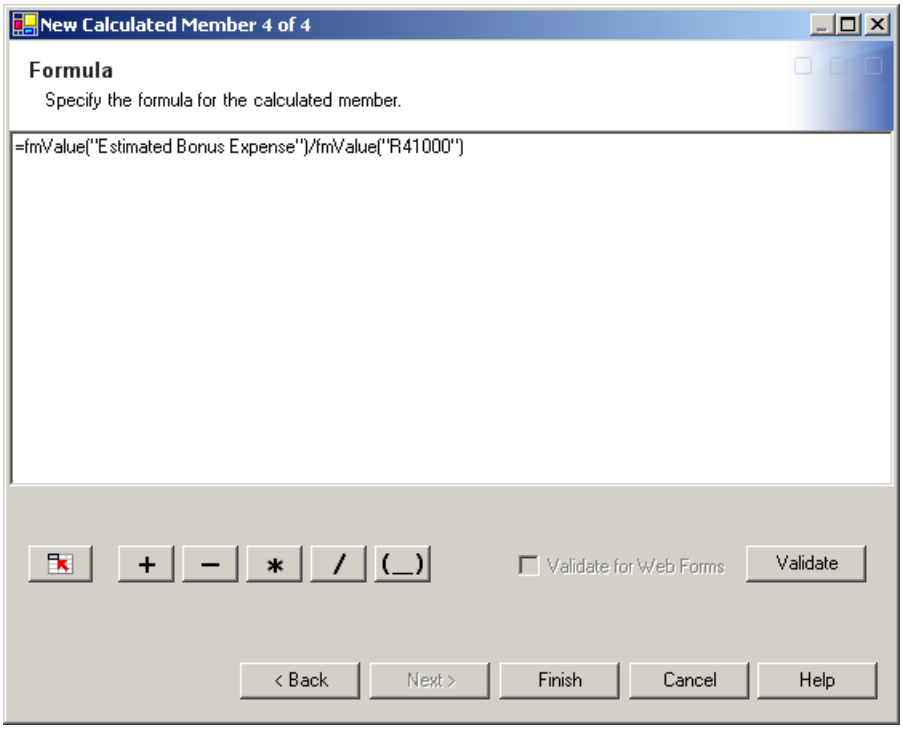
Product	Product Total
Customer	Customer Total
Analysis	Budget
Currency	USD
Frequency	PTD
Time	2009

	Finance	Accounting	Corporate Legal	Payroll	Purchasing
R41000 (Salaries)	25,000.00	5,000.00	10,000.00	4,000.00	6,000.00
Estimated Bonus Expense	6,250.00	1,250.00	2,500.00	1,000.00	1,500.00

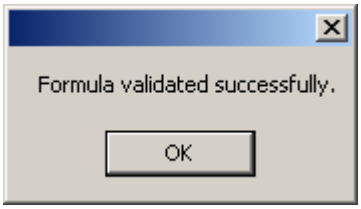


6 Click the **Formula Cell Selector** icon.

You are returned to the Formula window.

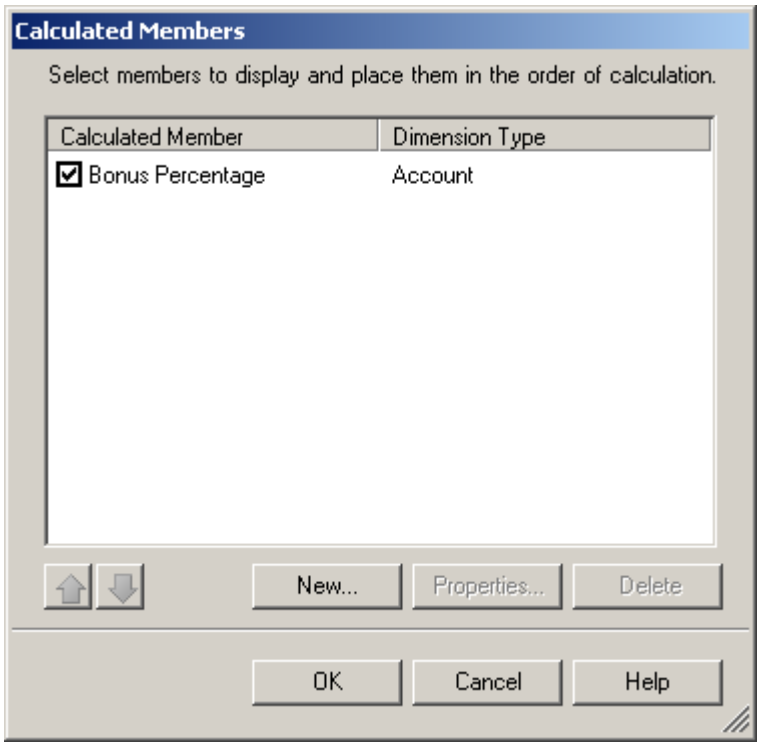


7 After you complete the formula expression, select **Validate** to ensure its accuracy.



8 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.



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

The results are displayed in the following table:

Product	Product Total
Customer	Customer Total
Analysis	Budget
Currency	USD
Frequency	PTD
Time	2009

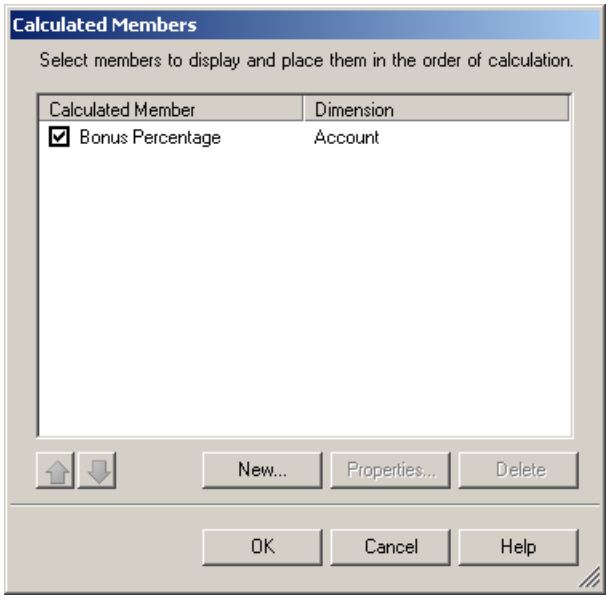
	Finance	Accounting	Corporate Legal	Payroll	Purchasing
R41000 (Salaries)	25,000.00	5,000.00	10,000.00	4,000.00	6,000.00
Estimated Bonus Expense	6,250.00	1,250.00	2,500.00	1,000.00	1,500.00
Bonus Percentage	0.25	0.25	0.25	0.25	0.25

Removing an Excel-Based Calculated Member

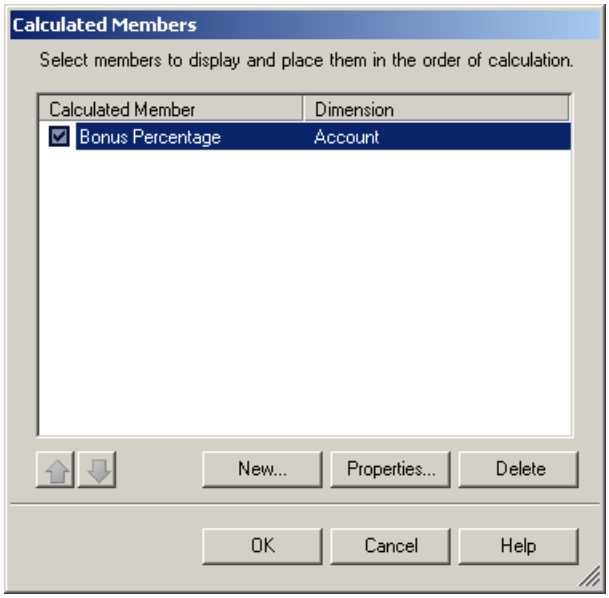
You can remove Excel-based calculated members either by disabling them or deleting them. The following steps illustrate how to remove an existing Excel-based calculated member:

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

The following window appears:



2 Highlight the member that you want to remove.



3 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.

Product	...	Product Total
Customer	...	Customer Total
Analysis	...	Budget
Currency	...	USD
Frequency	...	PTD
Time	...	2009

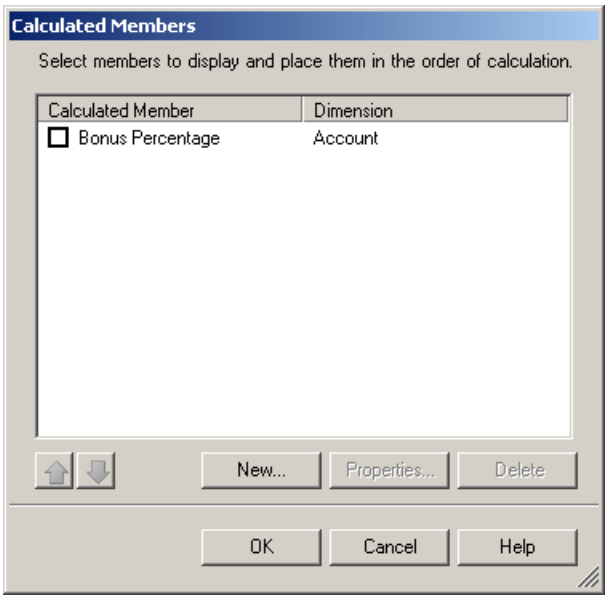
	Finance	Accounting	Corporate Legal	Payroll	Purchasing
R41000 (Salaries)	25,000.00	5,000.00	10,000.00	4,000.00	6,000.00
Estimated Bonus Expense	6,250.00	1,250.00	2,500.00	1,000.00	1,500.00

Editing an Excel-Based Calculated Member

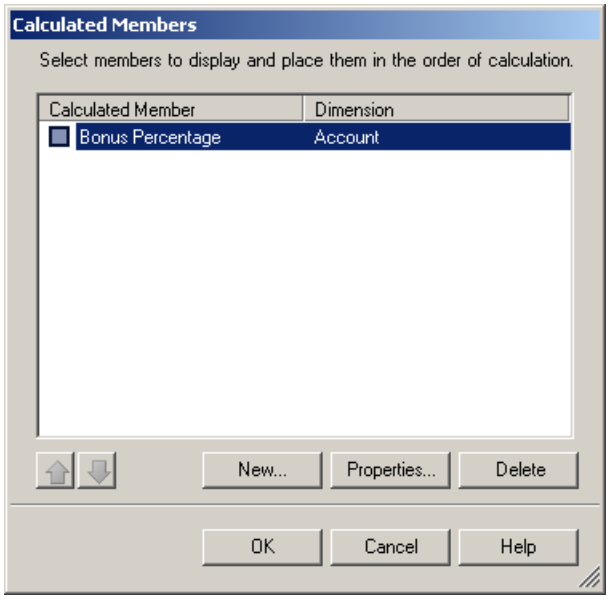
The following steps illustrate how to edit an existing Excel-based Calculated Member:

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

The following window appears:

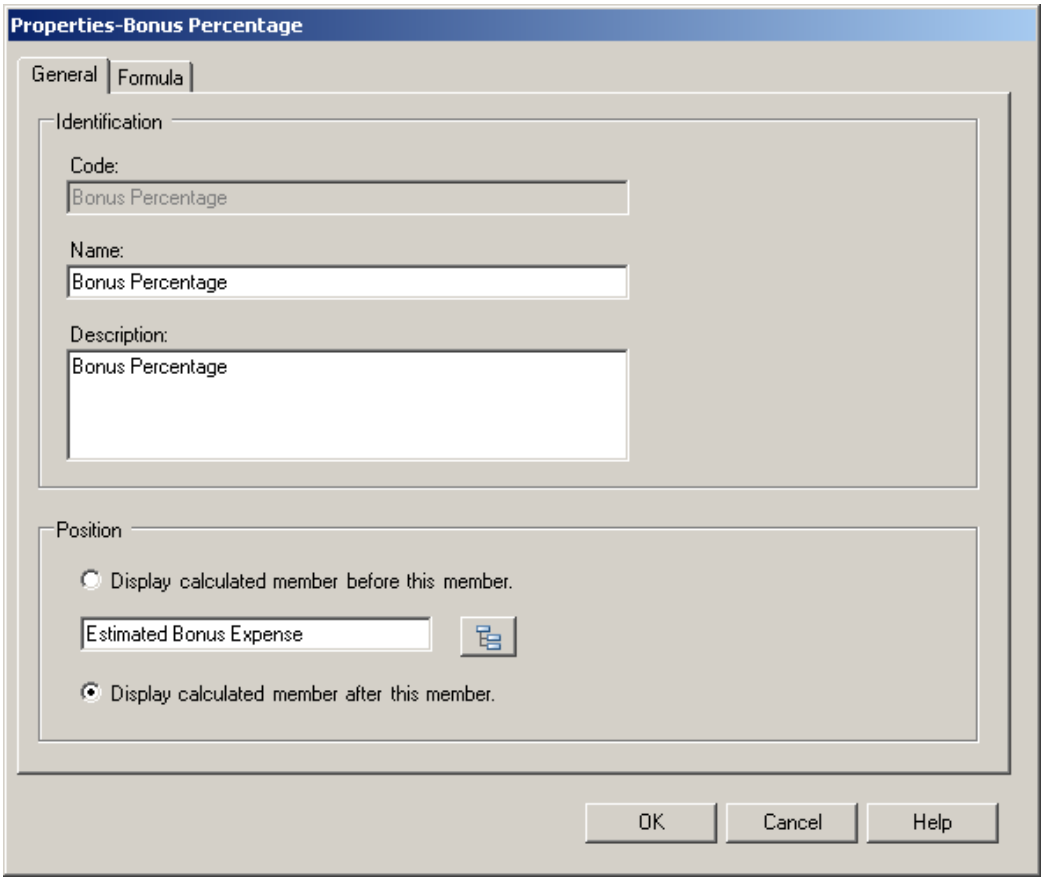


2 Highlight the member that you want to edit.

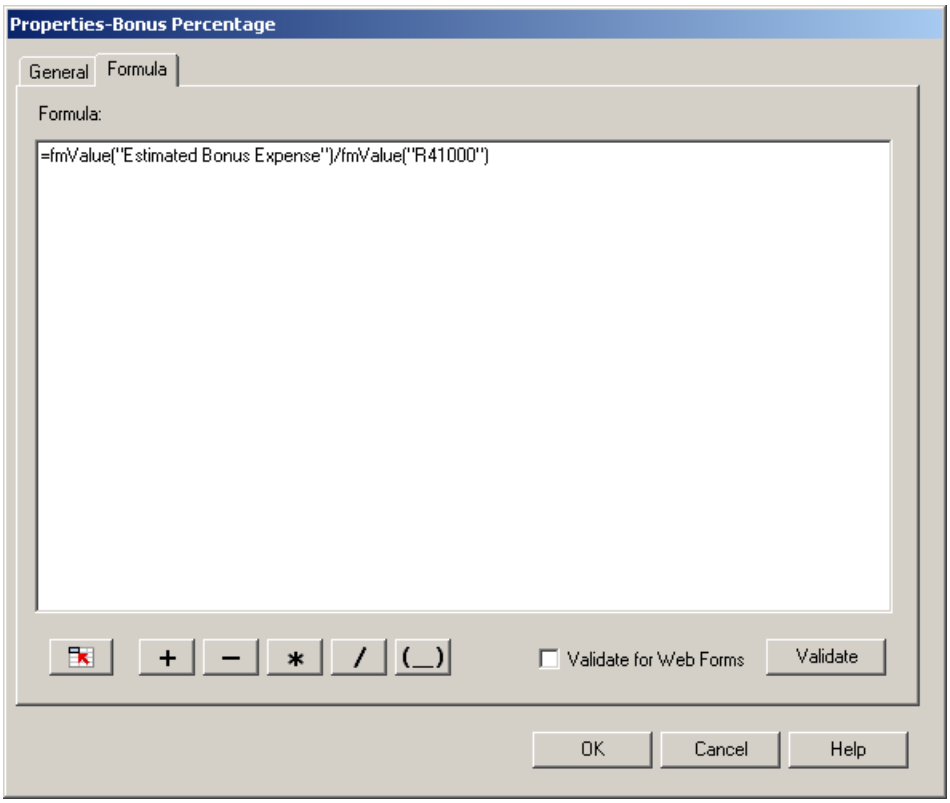


3 Click **`Properties`** to open the calculated member's Properties window.

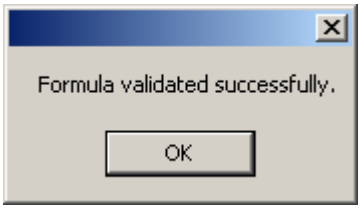
The default tab displayed is the **`General`** tab. This is where you modify the name, description, and position.



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

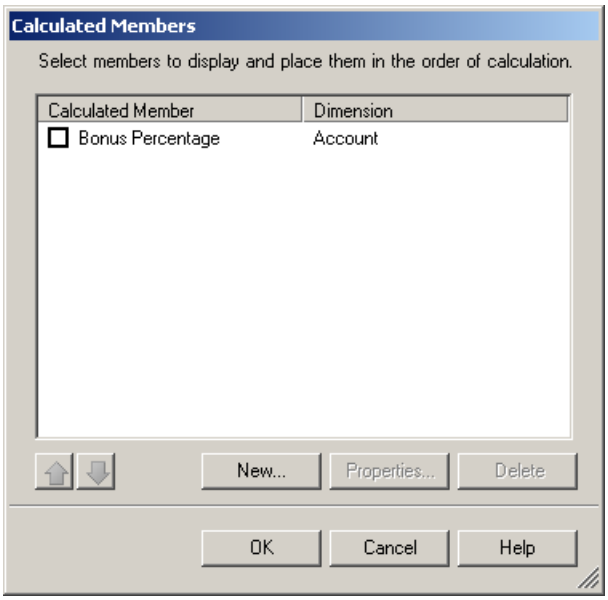


5 If the formula is valid, click **OK.**



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

7 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 Solutions** 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:

Product	Product Total
Customer	Customer Total
Analysis	Budget
Currency	USD
Frequency	PTD
Time	2009

	Finance	Accounting	Corporate Legal	Payroll	Purchasing
R41000 (Salaries)	25,000.00	5,000.00	10,000.00	4,000.00	6,000.00
Estimated Bonus Expense	6,250.00	1,250.00	2,500.00	1,000.00	1,500.00
Bonus Percentage	0.25	0.25	0.25	0.25	0.25

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

Product	Product Total
Customer	Customer Total
Analysis	Budget
Currency	USD
Frequency	PTD
Time	2009

	R41000 (Salaries)	Estimated Bonus Expense	Bonus Percentage
Finance	25,000.00	6,250.00	0.25
Accounting	5,000.00	1,250.00	0.25
Corporate Legal	10,000.00	2,500.00	0.25
Payroll	4,000.00	1,000.00	0.25
Purchasing	6,000.00	1,500.00	0.25

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

Product	Product Total
Customer	Customer Total
Analysis	Budget
Currency	USD
Frequency	PTD
Account	Bonus Percentage

	Jan 2009
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:

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

	Jan 2009			Q1 2009		
	R41000 (Salaries)	Estimated Bonus Expense	Bonus Percentage	R41000 (Salaries)	Estimated Bonus Expense	Bonus Percentage
Finance	25,000.00	6,250.00	0.25	25,000.00	6,250.00	0.25
Accounting	5,000.00	1,250.00	0.25	5,000.00	1,250.00	0.25
Corporate Legal	10,000.00	2,500.00	0.25	10,000.00	2,500.00	0.25
Payroll	4,000.00	1,000.00	0.25	4,000.00	1,000.00	0.25
Purchasing	6,000.00	1,500.00	0.25	6,000.00	1,500.00	0.25

Resolving Conflicts between Excel-Based Calculated Members

Introduction

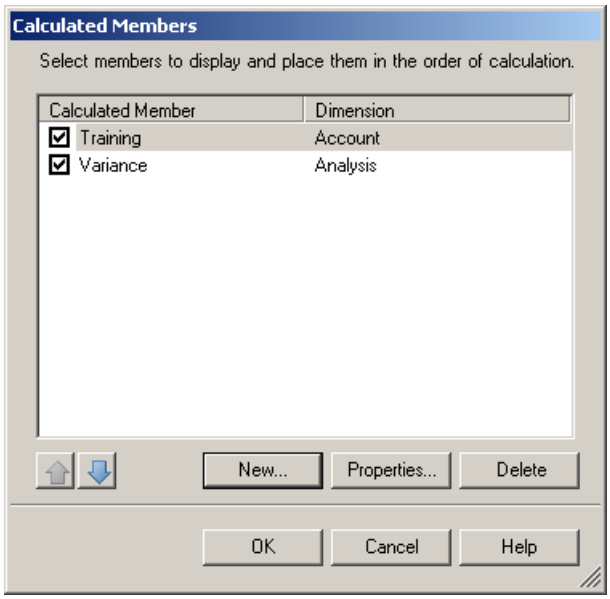
Excel-based calculated members can conflict only with other Excel-based calculated members. All server-side formula types (driver, modeling, and reporting) run before Excel-based calculated members. Therefore, in the event of a conflict involving two or more Excel-based calculated members, the members 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	Purchasing
Product	Product Total
Customer	Customer Total
Currency	USD
Time	Jan 2009
Frequency	PTD

	Actual	Budget	Variance
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	Purchasing
Product	Product Total
Customer	Customer Total
Currency	USD
Time	Jan 2009
Frequency	PTD

	Actual	Budget	Variance
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

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
Customer ...	Customer Total
Currency ...	USD
Time ...	Jan 2009
Frequency ...	PTD
Account ...	R63000 (Sales)

	Actual	Budget	Variance
Worldwide Operations	532,469.00	525,000.00	7,469.00
United States	420,397.00	415,000.00	5,397.00
Canada	79,389.00	75,000.00	4,389.00
Mexico	32,683.00	35,000.00	(2,317.00)

fmCode

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
Customer	Customer Total
Currency	USD
Time	Jan 2009
Frequency	PTD
Account	R63000 (Sales)

	Org Code	Actual	Budget	Variance
Worldwide Operations	0000	532,469.00	525,000.00	7,469.00
United States	0001	420,397.00	415,000.00	5,397.00
Sales	1000	420,397.00	415,000.00	5,397.00
Central	1100	107,237.00	110,000.00	(2,763.00)
Eastern	1400	151,411.00	145,000.00	6,411.00
Western	1700	161,749.00	160,000.00	1,749.00
Canada	0002	79,389.00	75,000.00	4,389.00
Mexico	0003	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

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 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
Customer	Customer Total
Currency	USD
Time	Jan 2009
Frequency	PTD
Account	R63000 (Sales)

	Functional Currency	Actual	Budget	Variance
Worldwide Operations	USD	532,469.00	525,000.00	7,469.00
United States	USD	420,397.00	415,000.00	5,397.00
Sales	USD	420,397.00	415,000.00	5,397.00
Canada	CAD	79,389.00	75,000.00	4,389.00
Canada Sales	CAD	79,389.00	75,000.00	4,389.00
Mexico	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
Time	...	Jan 2008
FREQUENCY	...	PTD

	Manager	Sales
Customer Total	Robinson	4,218,600.00
Partners	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

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	0001 (United States)
0.410000	0002 (Canada)
0.290000	0003 (Mexico)
0.330000	0100 (Administration)
0.330000	0110 (Corporate Communications)
0.330000	0120 (Facilities)
0.330000	0130 (Human Resources)
0.330000	0200 (Finance)
0.330000	0210 (Accounting)
0.330000	0220 (Corporate Legal)
0.330000	0230 (Payroll)
0.330000	0240 (Purchasing)
0.330000	0300 (Product Delivery)
0.330000	0400 (Telemarketing)
0.330000	0500 (Technology)
0.330000	0510 (Information Systems)
0.330000	0520 (Publications)
0.330000	0530 (Quality Assurance)

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 Rate =
fmRate("Tax Rate")

Product	Product Total
Analysis	Budget
Currency	USD
Customer	Customer Total
Time	Jul 2009
Frequency	PTD

	Net Income	Income Taxes	Est Tax Rate	Income Before Taxes
Worldwide Operations	82,080.00	41,920.00	0.00	124,000.00
United States	63,650.00	31,350.00	0.33	95,000.00
Sales	324,950.00	160,050.00	0.33	485,000.00
Administration	(67,000.00)	(33,000.00)	0.33	(100,000.00)
Finance	(50,250.00)	(24,750.00)	0.33	(75,000.00)
Product Delivery	(36,850.00)	(18,150.00)	0.33	(55,000.00)
Telemarketing	(40,200.00)	(19,800.00)	0.33	(60,000.00)
Technology	(67,000.00)	(33,000.00)	0.33	(100,000.00)
Canada	10,620.00	7,380.00	0.41	18,000.00
Mexico	7,810.00	3,190.00	0.29	11,000.00
Admin Mexico	(2,840.00)	(1,160.00)	0.29	(4,000.00)
HR Mexico	(1,420.00)	(580.00)	0.29	(2,000.00)
IT Support - Mexico	(2,130.00)	(870.00)	0.29	(3,000.00)
Mexico Sales	14,200.00	5,800.00	0.29	20,000.00

Since the driver rate was defined on the Internal Organization dimension, the Est Tax Rate Excel-based calculated member is positioned in the Account dimension on the columns. Had the member been placed on the same dimension as the dimension that the rates were qualified by, the resulting table would appear as displayed below:

Product	Product Total
Analysis	Budget
Currency	USD
Customer	Customer Total
Time	Jul 2009
FREQUENCY	PTD

	Net Income	Income Taxes	Income before Taxes
Est Tax Rate	################	##########	################
Worldwide Operations	82,080.00	41,920.00	124,000.00
United States	63,650.00	31,350.00	95,000.00
Sales	324,950.00	160,050.00	485,000.00
Administration	(67,000.00)	(33,000.00)	(100,000.00)
Finance	(50,250.00)	(24,750.00)	(75,000.00)
Product Delivery	(36,850.00)	(18,150.00)	(55,000.00)
Telemarketing	(40,200.00)	(19,800.00)	(60,000.00)
Technology	(67,000.00)	(33,000.00)	(100,000.00)
Canada	10,620.00	7,380.00	18,000.00
Mexico	7,810.00	3,190.00	11,000.00
Admin Mexico	(2,840.00)	(1,160.00)	(4,000.00)
HR Mexico	(1,420.00)	(580.00)	(2,000.00)
IT Support - Mexico	(2,130.00)	(870.00)	(3,000.00)
Mexico Sales	14,200.00	5,800.00	20,000.00

fmXRate

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","JUL2009")

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

	Jul 2009	Aug 2009	Sep 2009	Oct 2009	Nov 2009	Dec 2009
Period Avg Fx Rate - 1 USD per CAD	1.2599743	1.2618488	1.2623983	1.2618357	1.2597409	1.2600537
Net Income	(4,720.00)	(4,867.50)	(5,310.00)	(4,572.50)	(4,838.00)	(5,605.00)

fmCXRate

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", "JAN2008", "ACCOUNT", "E12000", "ORGANIZATION", "0002", "CUSTOM1 (Product)", "P00000", "CUSTOM2 (Customer)", "C00000", "FREQUENCY", "PTD", "ORGANIZATION_TRADER", "ALL", "SOURCE", "Total")

Organization ...	Canada
Product ...	Product Total
Customer ...	Customer Total
Analysis ...	Actual
Currency ...	CAD
Frequency ...	PTD

	Jan 2008	Feb 2008	Mar 2008
Hist Rate - Common Stock - 1 USD per CAD	1.235550	1.230825	1.231858
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 fmCode and fmProperty functions. 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	United States
Product	Product Total
Customer	Customer Total
Currency	USD
Time	Jan 2009
Frequency	PTD

	Actual	Budget	Variance
R16000 (Net Income)	57,029.00	77,750.00	(20,721.00)
R16500 (Income Taxes)	18,750.00	26,000.00	7,250.00
Income Before Taxes	75,779.00	103,750.00	(27,971.00)
R62000 (Gross Profit)	204,529.00	231,750.00	(27,221.00)
R63000 (Sales)	420,397.00	475,000.00	(54,603.00)
R64000 (Discounts & Allowances)	17,191.00	23,250.00	6,059.00
R67000 (Cost of Goods Sold)	198,677.00	220,000.00	21,323.00
R26000 (Operating Expenses)	128,750.00	128,000.00	(750.00)
Balance Sheet	0.00	(0.00)	(0.00)
A10000 (Assets)	278,504.80	278,750.00	245.20
L20000 (Liabilities)	116,475.80	96,000.00	20,475.80
E10000 (Stockholder's Equity)	162,029.00	182,750.00	(20,721.00)
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."

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

Example

Avg Fx Rate per 1 USD =

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

End of Period Fx Rate per 1 USD =

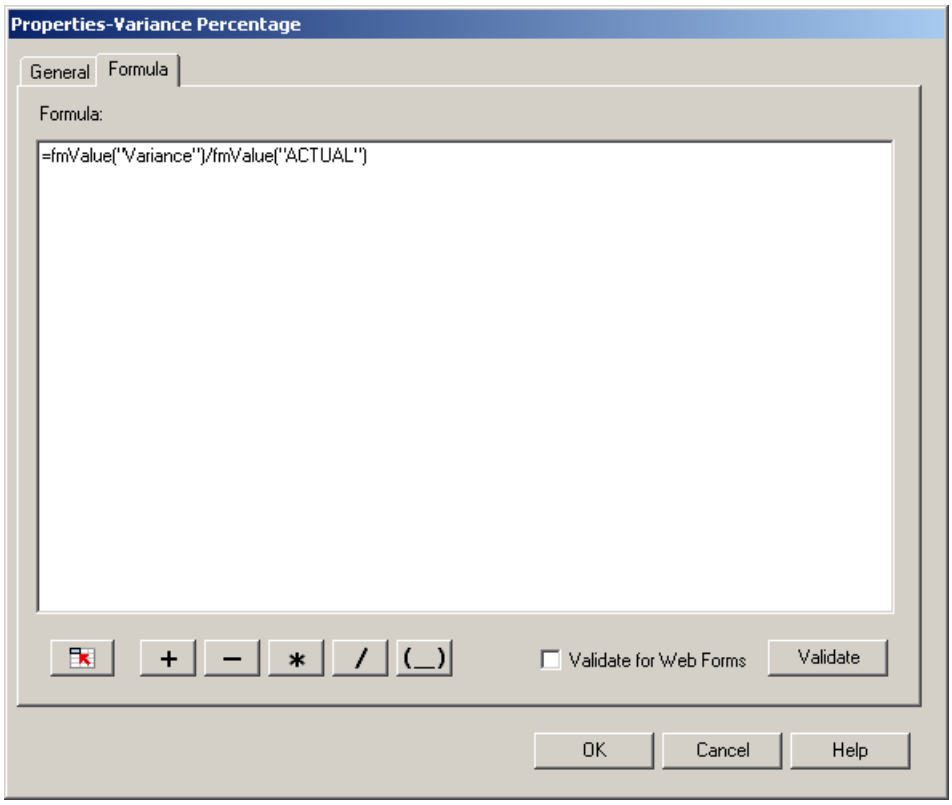
=fmXrate("PeriodClose","USD",fmProperty("ORGANIZATION","FunctionalCurrency"))

Product ...	Product Total
Customer ...	Customer Total
Analysis ...	Actual
Currency ...	USD
Time ...	Jan 2009
Frequency ...	PTD

	Worldwide Operations	United States	Canada	Mexico
Avg Fx Rate per 1 USD	1.00	1.00	1.26	13.37
End of Period Fx Rate per 1 USD	1.00	1.00	1.27	12.92
R16000 (Net Income)	169,101.00	57,029.00	79,389.00	32,683.00
Balance Sheet	0.00	0.00	0.00	(0.00)
A10000 (Assets)	456,103.02	278,504.80	130,299.22	47,299.00
L20000 (Liabilities)	202,067.69	116,475.80	70,975.89	14,616.00
E10000 (Stockholder's Equity)	254,035.33	162,029.00	59,323.33	32,683.00

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:

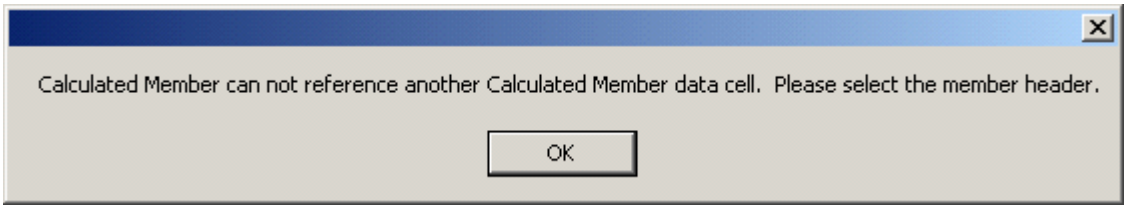


The resulting read-only table appears as follows:

Product	Product Total
Customer	Customer Total
Currency	USD
Time	Jan 2009
Frequency	PTD
Account	Net Income

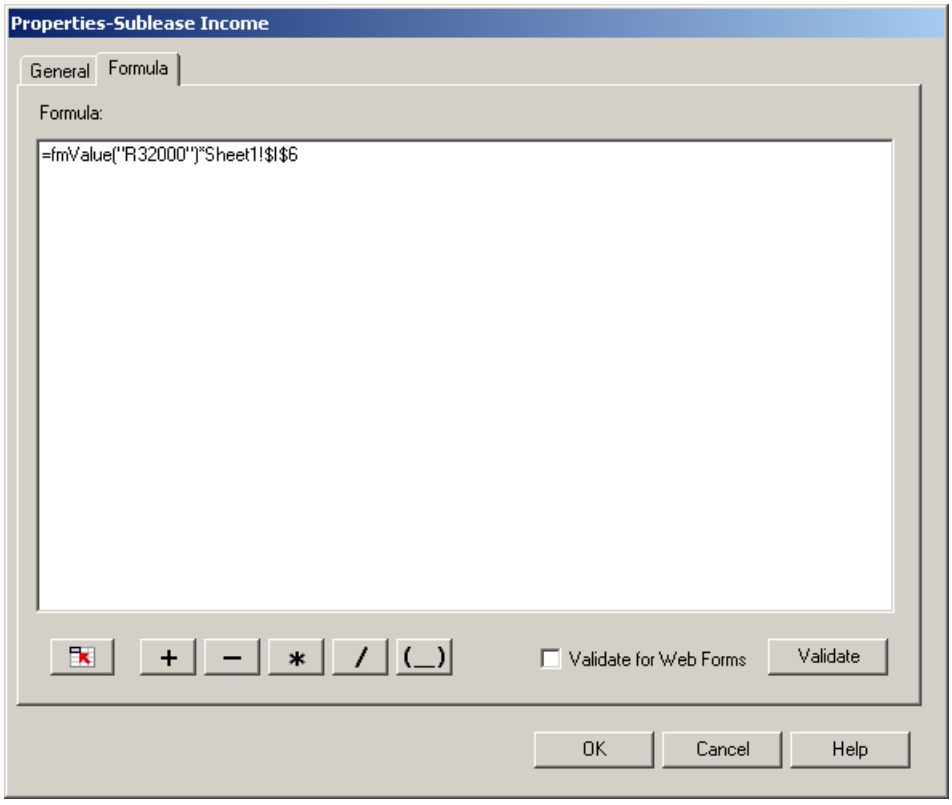
	Actual	Budget	Variance	Variance Percentage
Worldwide Operations	532,469.00	525,000.00	7,469.00	1.4%
United States	420,397.00	415,000.00	5,397.00	1.3%
Canada	79,389.00	75,000.00	4,389.00	5.5%
Mexico	32,683.00	35,000.00	(2,317.00)	-7.1%

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:



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.



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
Customer ...	Customer Total
Product ...	Product Total
Currency ...	USD
Frequency ...	PTD
Analysis ...	Budget

-0.35

	Jan 2008	Feb 2008	Mar 2008	Q1 2008
R31000 (Facilities)	39,750.00	39,750.00	39,750.00	119,250.00
R32000 (Rent)	25,000.00	25,000.00	25,000.00	75,000.00
Est. Sublease Income (Sublease Income)	(8,750.00)	(8,750.00)	(8,750.00)	(26,250.00)
R33000 (Electric)	5,000.00	5,000.00	5,000.00	15,000.00
R34000 (Water)	4,500.00	4,500.00	4,500.00	13,500.00
R35000 (Repairs & Maintenance)	2,500.00	2,500.00	2,500.00	7,500.00
R36000 (Telecom)	1,500.00	1,500.00	1,500.00	4,500.00
R37000 (I/S Expenses)	750.00	750.00	750.00	2,250.00
R38000 (Other Facilities Expenses)	500.00	500.00	500.00	1,500.00

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.

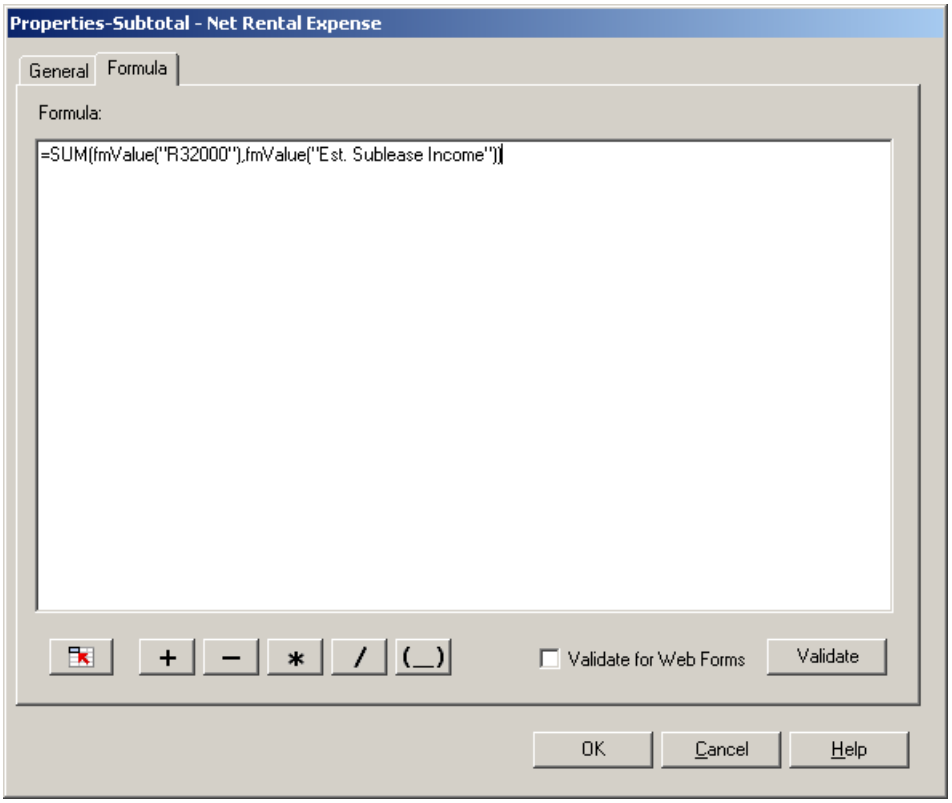
Using Any Excel Function or Valid Excel Expression

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
Customer ...	Customer Total
Product ...	Product Total
Currency ...	USD
Frequency ...	PTD
Analysis ...	Budget

	Jan 2008	Feb 2008	Mar 2008	Q1 2008
R31000 (Facilities)	39,750.00	39,750.00	39,750.00	119,250.00
Subtotal - Net Rental Expense	16,250.00	16,250.00	16,250.00	48,750.00
R32000 (Rent)	25,000.00	25,000.00	25,000.00	75,000.00
Est. Sublease Income (Sublease Income)	(8,750.00)	(8,750.00)	(8,750.00)	(26,250.00)
R33000 (Electric)	5,000.00	5,000.00	5,000.00	15,000.00
R34000 (Water)	4,500.00	4,500.00	4,500.00	13,500.00
R35000 (Repairs & Maintenance)	2,500.00	2,500.00	2,500.00	7,500.00
R36000 (Telecom)	1,500.00	1,500.00	1,500.00	4,500.00
R37000 (I/S Expenses)	750.00	750.00	750.00	2,250.00
R38000 (Other Facilities Expenses)	500.00	500.00	500.00	1,500.00

Excel-Based Calculated Members and Display Styles

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

Model Properties has debit accounts with a debit balance set to positive and credit accounts with credit balances as negative numbers.

On the table, Excel Properties has debit accounts with a debit balance as Default and credit accounts with credit balances as Default.

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

Organization	Worldwide Operations
Product	Product Total
Customer	Customer Total
Currency	USD
Time	Jan 2009
Frequency	PTD

	Actual	Budget	Variance
Net Income	(532,443.00)	(524,850.00)	7,593.00
Income Taxes	18,750.00	26,000.00	7,250.00
Income Before Taxes	(551,193.00)	(550,850.00)	343.00
Gross Profit	(679,943.00)	(678,850.00)	1,093.00
Sales	(895,811.00)	(922,100.00)	(26,289.00)
Discounts & Allowances	17,191.00	23,250.00	6,059.00
Cost of Goods Sold	198,677.00	220,000.00	21,323.00
Operating Expenses	128,750.00	128,000.00	(750.00)

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:

=$D$11-$C$11

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

Model Properties has debit accounts with a debit balance set to positive and credit accounts with credit balances as negative numbers.

On the table, Excel Properties has 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	Worldwide Operations
Product	Product Total
Customer	Customer Total
Currency	USD
Time	Jan 2009
Frequency	PTD

	Actual	Budget	Variance
Net Income	532,443.00	524,850.00	7,593.00
Income Taxes	18,750.00	26,000.00	7,250.00
Income Before Taxes	551,193.00	550,850.00	343.00
Gross Profit	679,943.00	678,850.00	1,093.00
Sales	895,811.00	922,100.00	(26,289.00)
Discounts & Allowances	17,191.00	23,250.00	6,059.00
Cost of Goods Sold	198,677.00	220,000.00	21,323.00
Operating Expenses	128,750.00	128,000.00	(750.00)

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",$C$11-$D$11,$D$11-$C$11)

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

Model Properties has debit accounts with a debit balance set to positive and credit accounts with credit balances as negative numbers.

On the table, Excel Properties has 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	Worldwide Operations
Product	Product Total
Customer	Customer Total
Currency	USD
Time	Jan 2009
Frequency	PTD

	FRCAST	Variance
Net Income	0.00	7,593.00
Income Taxes	0.00	7,250.00
Income Before Taxes	0.00	343.00
Gross Profit	0.00	1,093.00
Sales	0.00	(26,289.00)
Discounts & Allowances	0.00	6,059.00
Cost of Goods Sold	0.00	21,323.00
Operating Expenses	0.00	(750.00)

Even when the members referenced in the Excel-based calculated member are not on the table, the Excel 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("Model"," ACCOUNT", "R16000", "ANALYSIS",
"ACTUAL", "ORGANIZATION", "0000", "CUSTOM1 (Product)", "P00000",
"CUSTOM2 (Customer)", "C00000", "CURRENCY", "USD", "TIME", "JAN2009",
"FREQUENCY", "PTD") * TableCreditNeg("NewTable0") /
TableScale("NewTable0"))-(CDAGet("Model", "ACCOUNT", "R16000",
"ANALYSIS", "BUDGET", "ORGANIZATION", "0000", "CUSTOM1 (Product)",
"P00000", "CUSTOM2 (Customer)", "C00000", "CURRENCY", "USD", "TIME",
"JAN2009", "FREQUENCY", "PTD") * TableCreditNeg("NewTable0") /
TableScale("NewTable0")),(CDAGet("Model", "ACCOUNT", "R16000",
"ANALYSIS", "BUDGET", "ORGANIZATION", "0000", "CUSTOM1 (Product)",
"P00000", "CUSTOM2 (Customer)", "C00000", "CURRENCY", "USD", "TIME",
"JAN2009", "FREQUENCY", "PTD") * TableCreditNeg("NewTable0") /
TableScale("NewTable0"))-(CDAGet("Model", "ACCOUNT", "R16000",
"ANALYSIS", "ACTUAL", "ORGANIZATION", "0000", "CUSTOM1 (Product)",
"P00000", "CUSTOM2 (Customer)", "C00000", "CURRENCY", "USD", "TIME",
"JAN2009", "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. The requested value is retrieved.

```
Beginning Bal - Equipment =
fmValue("A26000","TIME",-1)
```

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

	Oct 2008	Nov 2008	Dec 2008
Ending Bal. Equipment	57,832.00	57,832.00	83,477.00
Additions - Equipment	55,276.00	55,276.00	80,921.00
Disposals - Equipment	(5,692.00)	(5,692.00)	(5,692.00)
Transfers & W/O - Equipment	8,248.00	8,248.00	8,248.00
Beginning Bal - Equipment	42,589.00	57,832.00	57,832.00

Since Sep 2008 is not on the table but a valid member in the cycle and model, the resulting value of 42,589 was retrieved with the following formula:

```
=(CDAGet("Model", "ACCOUNT", "A26000", "TIME", "SEP2008",
"ORGANIZATION", "0000", "CUSTOM1 (Product)", "P00000", "CUSTOM2
(Customer)", "C00000", "CURRENCY", "USD", "ANALYSIS", "ACTUAL",
"FREQUENCY", "PTD", "ORGANIZATION_TRADER", "ALL", "SOURCE",
"Total") * TableDebitPos("NewTable0") / TableScale("NewTable0"))
```

Due to the fact that Oct 2008 is on the table, the formula for Beginning Bal - Equipment for Nov 2008 resolves to:

```
=$C$11
```

If a member is not included in the model but is in the cycle, an Excel-based calculated member resolves to a value of zero. This more typically occurs with members of the Time dimension. In the following example, the Cycle's starting Time member is Jan 2008, and the Model's starting Time Member is Feb 2008. Assume that the Jan 2008 Ending Bal.Equipment is $38,741.

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

	Feb 2008	Mar 2008	Apr 2008
Ending Bal. Equipment	38,741.00	38,741.00	38,741.00
Additions - Equipment	38,741.00	38,741.00	38,741.00
Disposals - Equipment	0.00	0.00	0.00
Transfers & W/O - Equipment	0.00	0.00	0.00
Beginning Bal - Equipment	0.00	38,741.00	38,741.00

In this example, Beginning Bal - Equipment for Feb 2008 is 0 due to the Time member start date on the Model. In Feb 2008, the Excel-based calculated member resolves to a formula expression of =0.

Similarly, if the Model's and Cycle's starting periods are both Jan 2008, the resulting value for Jan 2008 Beginning Bal - Equipment would be zero. The formula expression for this crossing would be =0.

If a member is not included on the table and is not in both the Model and Cycle, the resulting value is 0.

Excel-Based Calculated Members on the Web

All Excel-based calculated member functions are supported on the Web, whether in a Financial form or an Operational Planning form. Calculated members that resolve to CDA formulas are retrieved at the time the form is opened. FmValue results that resolve to cell references are updated through the following actions:

- Expanding
- Collapsing
- Pivoting

- Slicing
- Clicking Refresh

Differences between Calculated Members in Excel and on the Web

Excel behavior and Web behavior differ in displaying Excel-based calculated members in the following scenarios:

Scenario 1: Time Offsets

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

Excel The resulting value is displayed.

Web A gray, non-writeable cell 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.

Web A gray, non-writeable cell 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.

Web The calculated member is not 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

Default Members

Explicit Members and Implicit (Default) Members

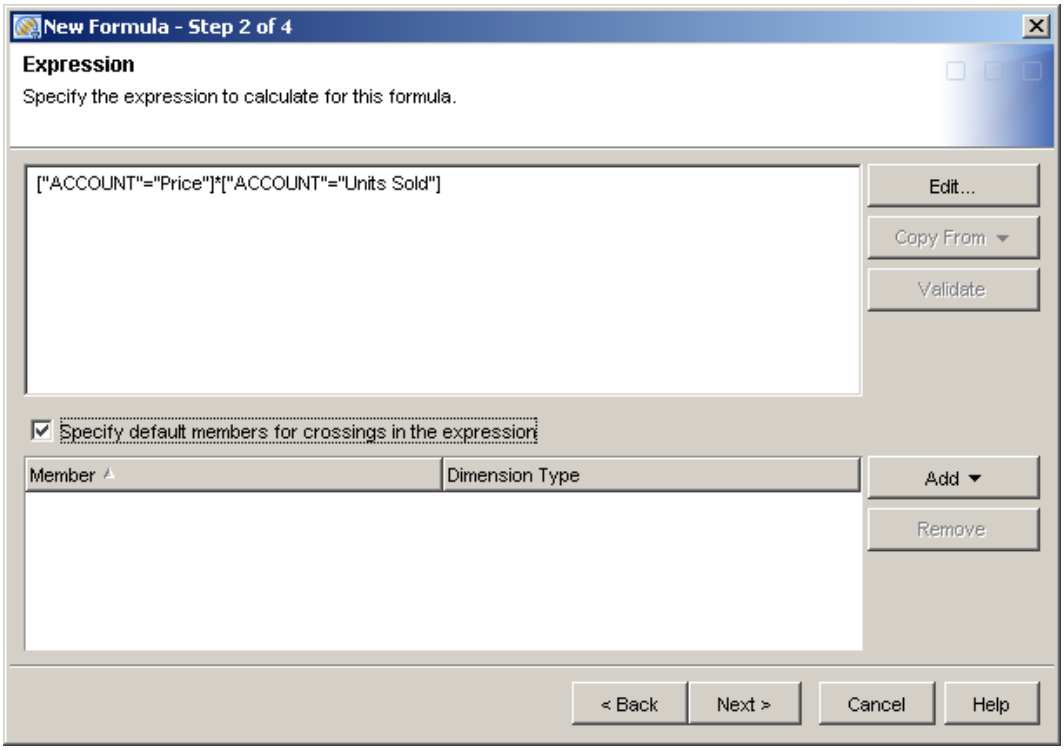
All SAS Financial Management formulas can contain two types of default members: *explicit members* and *implicit members*.

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

Default members are considered 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, default members offer an easy alternative to assigning members to each formula input in the formula expression.

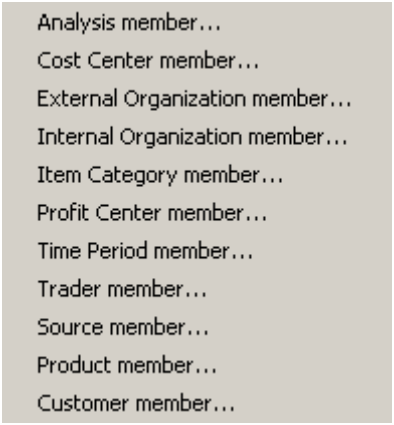
You can select only one member *per dimension* as a default member.

1 Use the **`Specify default members for crossings in the expression`** check box in the Formula wizard to select default members on an expression-by-expression basis.



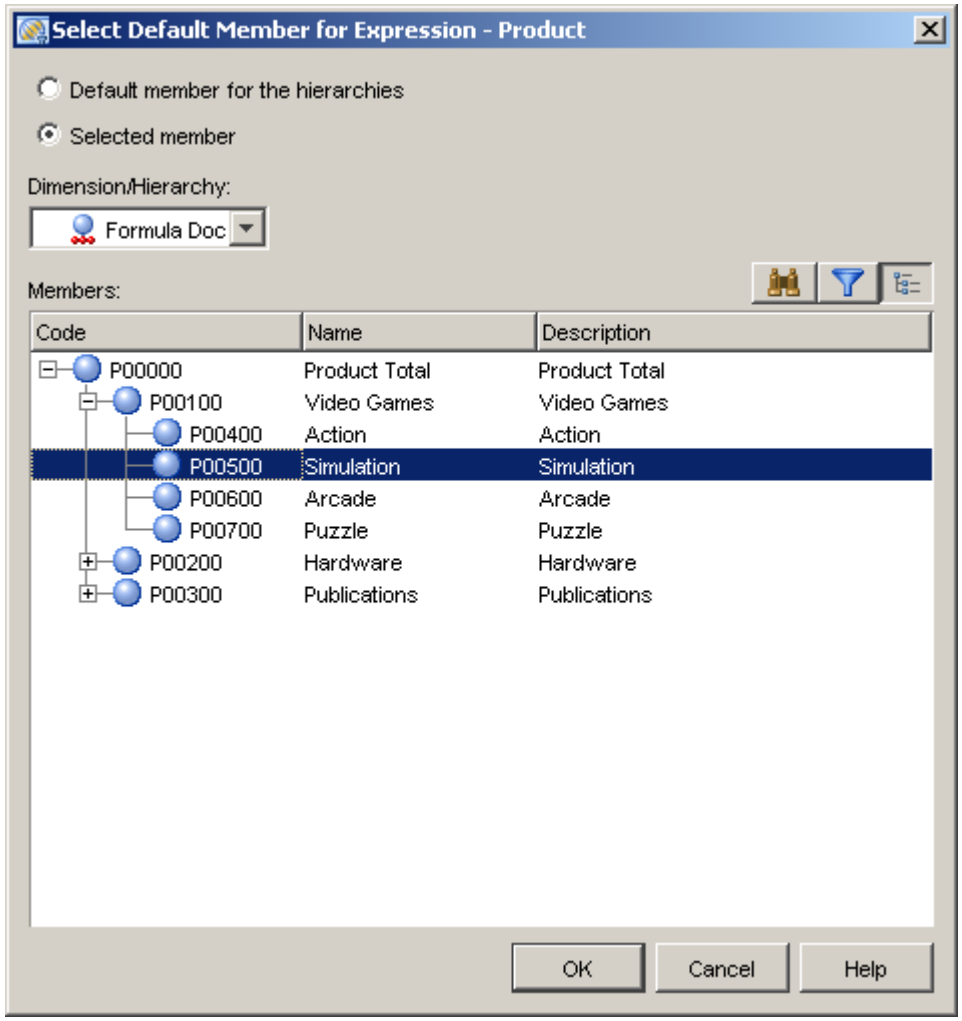
2 Click **Add**.

The list of available dimensions is displayed.

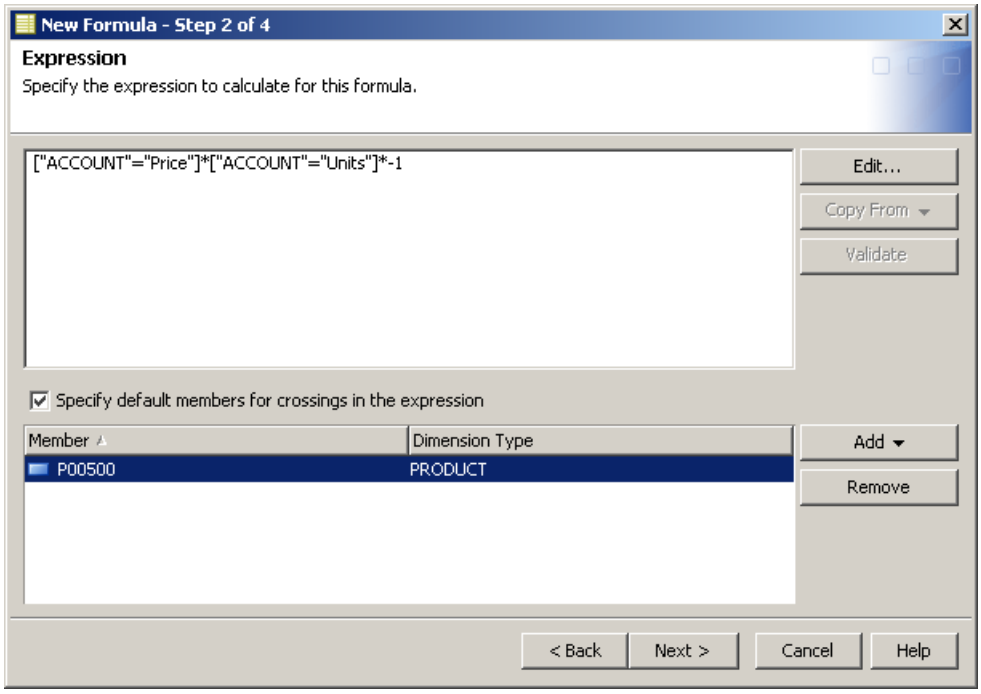


When you select the desired dimension, its hierarchies and members are displayed.

3 Select a member and click **OK**.

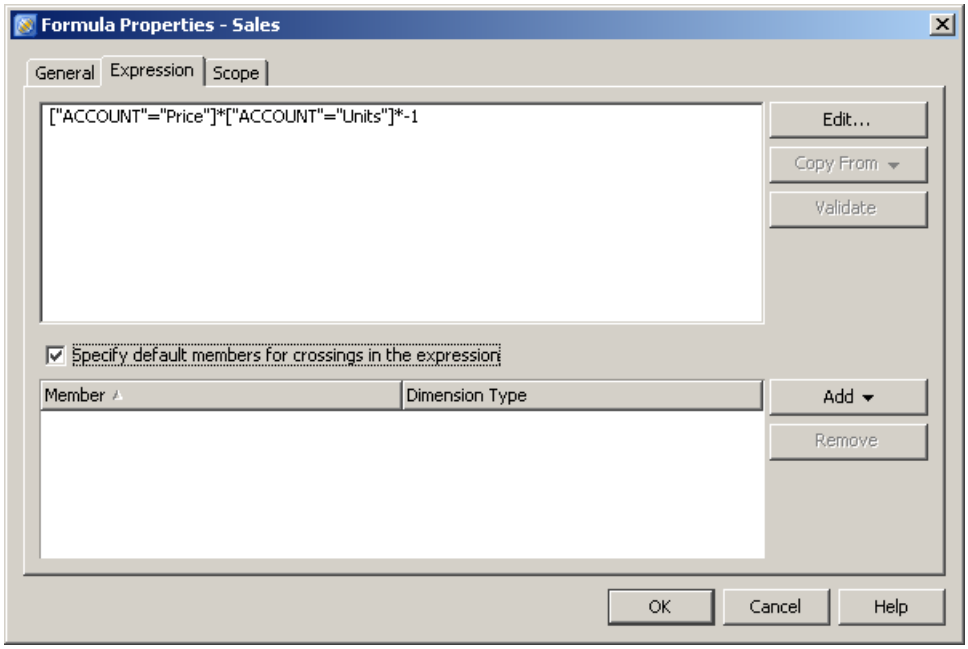


The results of the default member selection for a given dimension are displayed. You can continue to select additional default members for different dimensions, modify current selections, and remove current selections.



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

4 Select the **`Specify default members for crossings in the expression`** check box to activate the default member option.



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

Formula

Sales = ["ACCOUNT"="PRICE"]*["ACCOUNT" ="UNITS"]*-1

Default Members

Product dimension: P00500 - Simulation

Adding product member P00500 as a default member has the same effect as if the formula expression were written as follows:

Sales = ["ACCOUNT"="PRICE"]["PRODUCT"="P00500"]*["ACCOUNT" ="UNITS"] ["PRODUCT"="P00500"]*-1

Facts

The following data entry table displays the facts that were entered for Price and Units for the leaf members under Product member P00100 (Video Games).

Organization	Western
Customer	Civillian
Analysis	Budget
Currency	USD
Frequency	PTD
TIME	Jan 2008
Source	Form Data
Trader	EXT

	P00100 (Video Games)	P00100.vc (Video Games)	P00400 (Action)	P00500 (Simulation)	P00600 (Arcade)	P00700 (Puzzle)
Price		0.00	15.00	20.00	10.00	25.00
Units	500.00	0.00	100.00	50.00	200.00	150.00

Results

In the following table, results for Sales for all leaf members under Product member P00100 (Video Games) are (1,000.00). This is computed by multiplying the Price for P00500 (Simulation) by the Units of P00500 (Simulation): (20 x 50)* -1 = (1,000.00).

Organization	Western
Customer	Civillian
Analysis	Budget
Currency	USD
Frequency	PTD
TIME	Jan 2008
Source	Form Data
Trader	EXT

	P00100 (Video Games)	P00100.vc (Video Games)	P00400 (Action)	P00500 (Simulation)	P00600 (Arcade)	P00700 (Puzzle)
Price		0.00	15.00	20.00	10.00	25.00
Units	500.00	0.00	100.00	50.00	200.00	150.00
Sales	(5,000.00)	(1,000.00)	(1,000.00)	(1,000.00)	(1,000.00)	(1,000.00)

Since the Sales account is a modeling formula, the Sales account result for product P00100 (Video Games) is (5,000.00). This is the sum of all the subordinate leaf members. Note that the formula runs at the virtual child of P00100 (Video Games) as well. This is the intended behavior and should be considered when using default members.

Explicit versus Implicit: Who Wins?

The previous section provides an example of a formula expression with an implicit reference to Product member P00500. 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"="P00500"]["ACCOUNT"="PRICE"]*["PRODUCT"="P00500"]["ACCOUNT"="UNITS"]* -1

In some scenarios, you might use a default 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 P00600, while the Default member selection implicitly states Product member P00500.

Formula

Sales = ["PRODUCT"="P00600"]["ACCOUNT"="PRICE"]*["PRODUCT"="P00600"]["ACCOUNT"="UNITS"]* -1

Default Members

Product dimension: P00500 - Simulation

Facts

The following data-entry table displays the facts that were entered for Price and Units for the leaf members under Product member P00100 (Video Games).

Organization ...	Western
Customer ...	Civillian
Analysis ...	Budget
Currency ...	USD
Frequency ...	PTD
TIME ...	Jan 2008
Source ...	Form Data
Trader ...	EXT

	P00100 (Video Games)	P00100.vc (Video Games)	P00400 (Action)	P00500 (Simulation)	P00600 (Arcade)	P00700 (Puzzle)
Price		0.00	15.00	20.00	10.00	25.00
Units	500.00	0.00	100.00	50.00	200.00	150.00

Results

Organization ...	Western
Customer ...	Civillian
Analysis ...	Budget
Currency ...	USD
Frequency ...	PTD
TIME ...	Jan 2008
Source ...	Form Data
Trader ...	EXT

	P00100 (Video Games)	P00100.vc (Video Games)	P00400 (Action)	P00500 (Simulation)	P00600 (Arcade)	P00700 (Puzzle)
Price		0.00	15.00	20.00	10.00	25.00
Units	500.00	0.00	100.00	50.00	200.00	150.00
Sales	(10,000.00)	(2,000.00)	(2,000.00)	(2,000.00)	(2,000.00)	(2,000.00)

In the preceding table, results for Sales for all leaf members under P00100 (Video Games) are (2,000.00). This is computed by multiplying the Price for P00600 (Arcade) by the Units of P00600 (Arcade): (10 x 200)* -1 = (2,000.00).

In the following scenario, the formula expression is modified slightly so that the product P00600 is explicitly stated only with the Units account. The default member reference and the facts remain unchanged.

Formula

Sales = ["ACCOUNT"="PRICE"]*["PRODUCT"="P00600"] ["ACCOUNT"="UNITS"]* -1

Default Members

Product dimension: P00500 - Simulation

Results

Organization ... Western
Customer ... Civillian
Analysis ... Budget
Currency ... USD
Frequency ... PTD
TIME ... Jan 2008
Source ... Form Data
Trader ... EXT

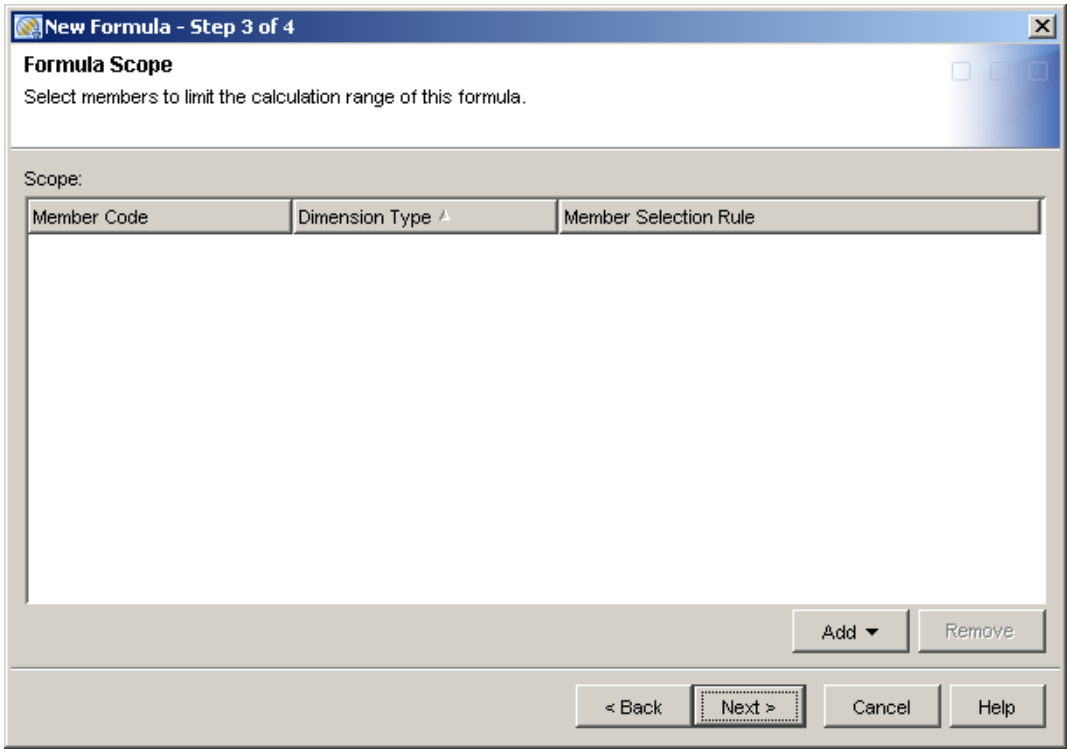
	P00100 (Video Games)	P00100.vc (Video Games)	P00400 (Action)	P00500 (Simulation)	P00600 (Arcade)	P00700 (Puzzle)
Price		0.00	15.00	20.00	10.00	25.00
Units	500.00	0.00	100.00	50.00	200.00	150.00
Sales	(20,000.00)	(4,000.00)	(4,000.00)	(4,000.00)	(4,000.00)	(4,000.00)

In the preceding table, results for Sales for all leaf members under P00100 (Video Games) are (4,000.00). This is computed by multiplying the Price for P00500 (Simulation) by the Units of P00600 (Arcade): (20 * 200)* -1 = (4,000.00).

Formula Scope

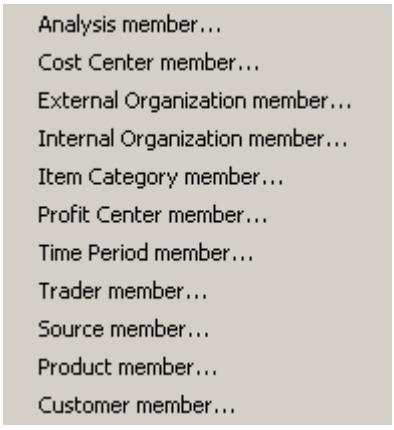
Formula scope is an optional means of restricting where a formula runs. It can be applied on an expression-by-expression basis and is available for all server-side formulas (reporting, driver, and modeling formulas). From a performance standpoint, formula scope is most effective when it is used with modeling formulas to limit the number of crossings where a formula runs.

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:



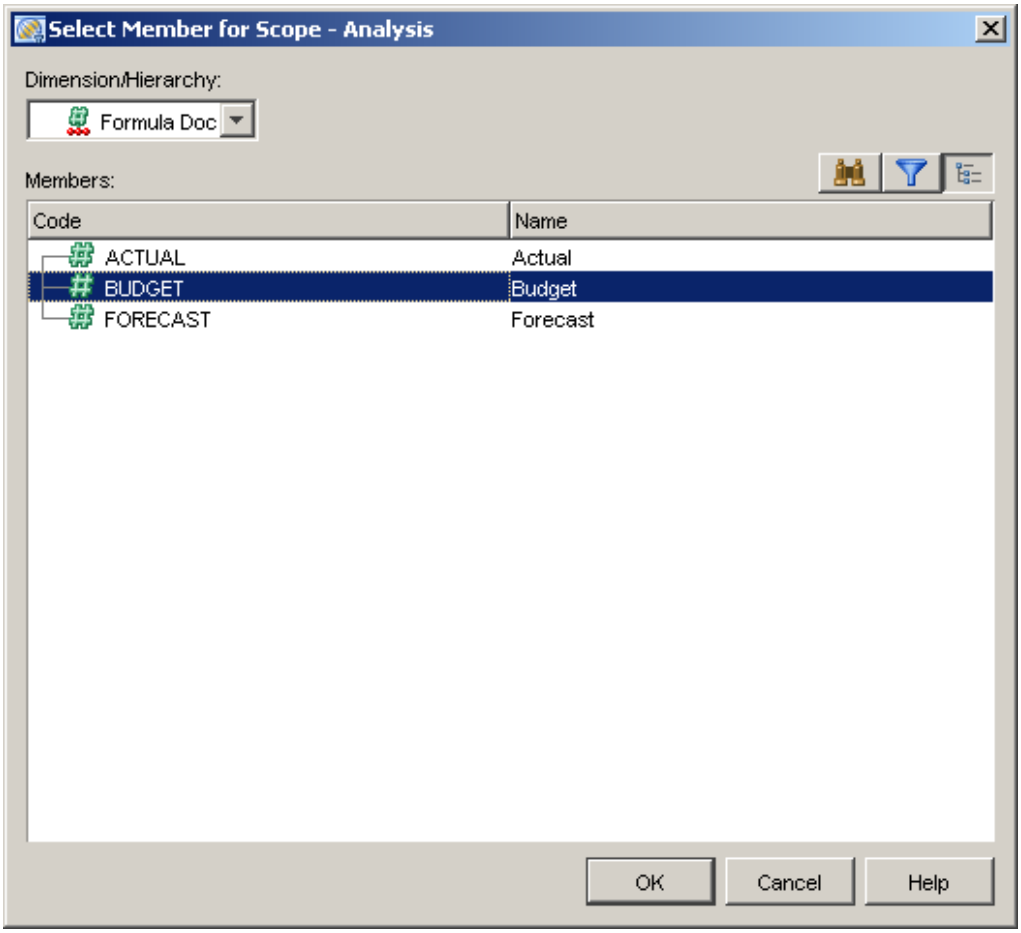
1 Click **Add**.

A list of available dimensions is displayed.



When you select a dimension, its hierarchies and members are displayed. Multiple-member selection is available.

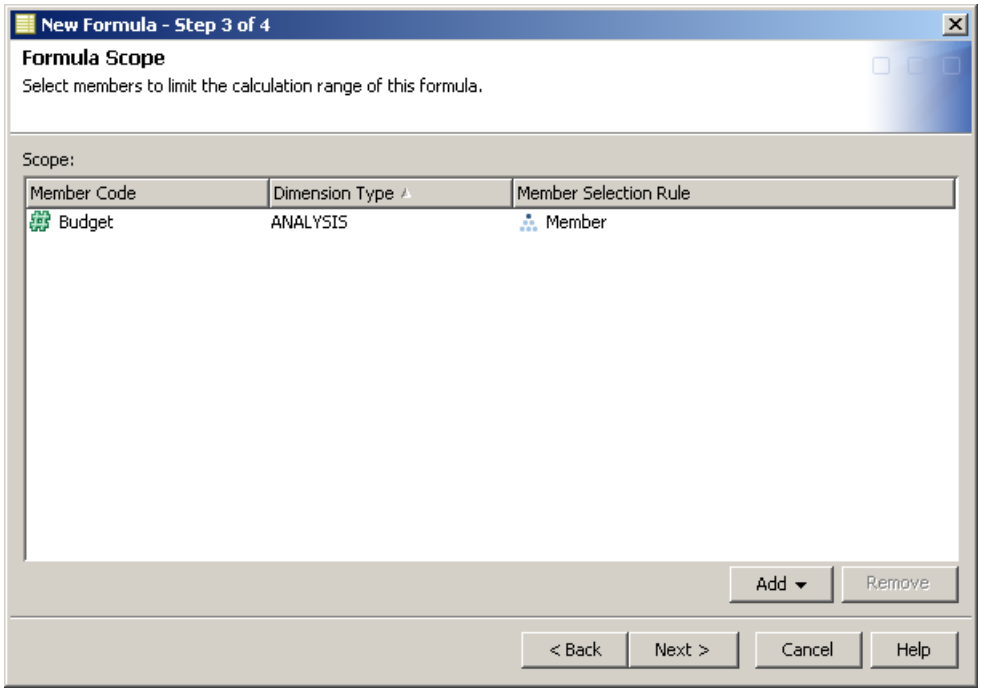
2 Select the required members and click **OK**.



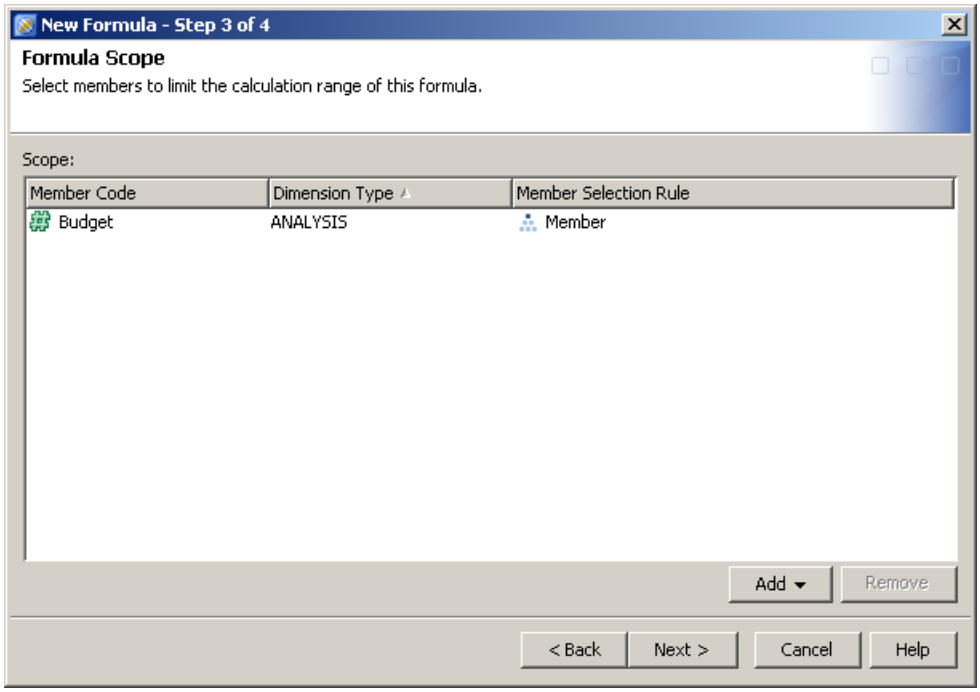
The results of the formula scope member selection for a given dimension are displayed. You can continue to do one or more of the following tasks:

- select members for different dimensions
- modify current selections
- remove current selections

3 When the formula scope is completely defined, click **Next** to continue through the Formula wizard.



For existing calculated members, formula scope is available through **Member Properties > Formulas** under the **Scope** tab.



To illustrate how formula scope works, consider the following driver formula:

Formula

Sales = ["ACCOUNT"="PRICE"]*["ACCOUNT"="UNITS"]* -1

Formula Scope

["ANALYSIS"="Budget"]

Facts and Results

The following data-entry table displays these facts:

- facts that were entered for Price and Units for the Budget member of the Analysis dimension
- facts that were entered for Sales for the Actual member of the Analysis dimension

Because the Budget member was selected for formula scope, only the crossings that contain the account member Sales and the Analysis member Budget are gray (calculated). Crossings that contain the account member Sales and the Analysis member Actual yellow (available for data entry). All crossings for the Variance member of the Analysis dimension are gray because Variance is a calculated member.

ORGANIZATION	Eastern U.S. Operations
Product	Arcade
FREQUENCY	PTD
Customer	HAL
Currency	USD
ORGANIZATION_TRADER	EXT
Source	Form Data

	Oct 2008			Nov 2008			Dec 2008		
	Actual	Budget	Variance	Actual	Budget	Variance	Actual	Budget	Variance
Price	0.00	7.50	7.50	0.00	7.50	7.50	0.00	7.50	7.50
Units	0.00	100.00	100.00	0.00	120.00	120.00	0.00	150.00	150.00
Sales	(850.00)	(750.00)	100.00	(925.00)	(900.00)	25.00	(1,100.00)	(1,125.00)	(25.00)

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.

There are three member selection rules:

- **Member** is available for reporting formulas and for any type of formula that is scoped on an Analysis member.
- **Leaf descendants of Member if any, otherwise Member** is available for modeling formulas and driver formulas.
- **Member and its descendants** is available for reporting formulas.

Here are two scenarios to illustrate how member selection rules are determined.

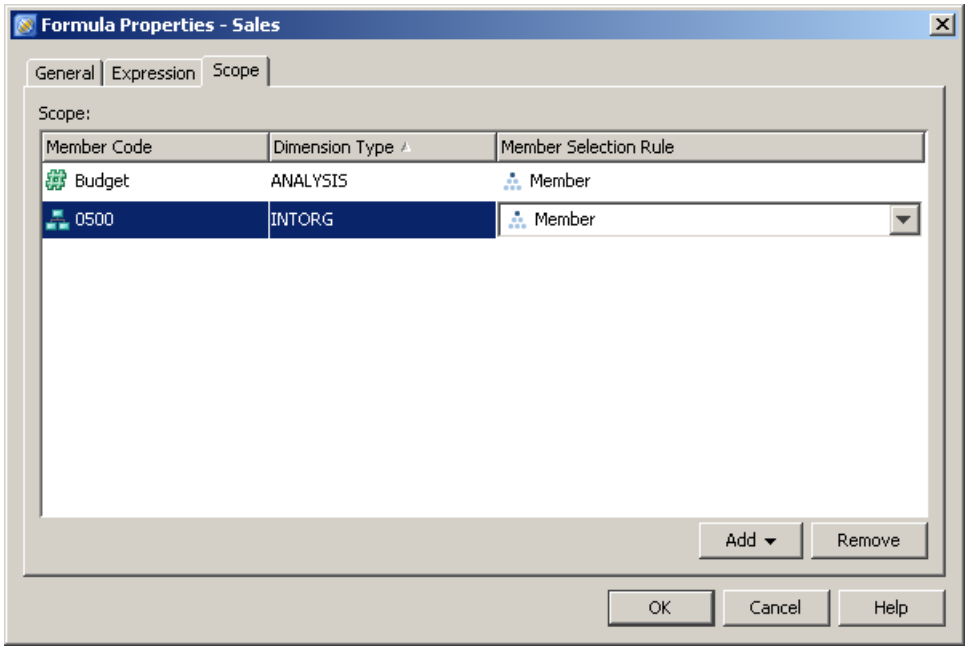
Scenario 1

Reporting formula

Formula Scope

["ANALYSIS"="BUDGET"]
["INTORG"= "0500"]

The **Scope** tab appears as follows:



Because the Analysis dimension has only flat hierarchies, the only selection rule available for an Analysis member is **Member**. Because IntOrg member 0500 is a roll-up member and the formula is a reporting formula, you can select one of two member selection rules: **Member and its descendants** or **Member**.

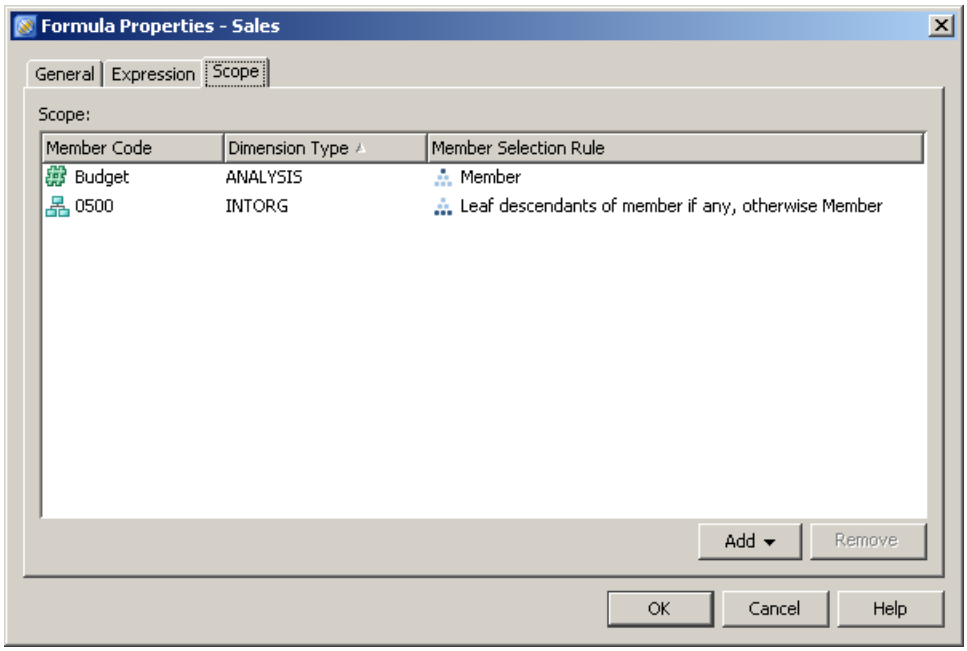
Scenario 2

Driver formula or modeling formula

Formula Scope

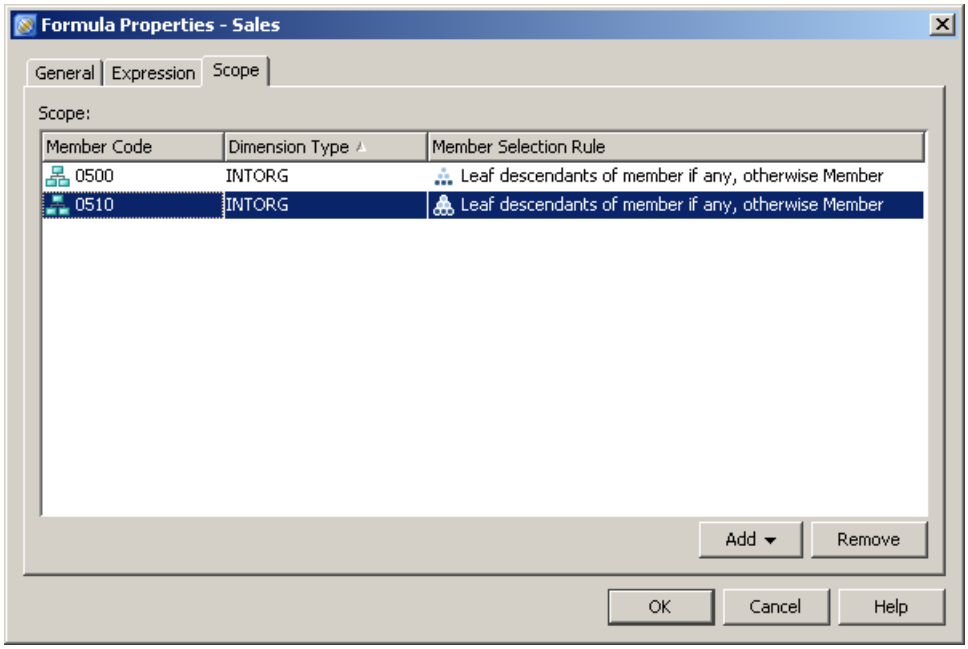
["ANALYSIS"="BUDGET"]
["INTORG"= "0500"]

The **Scope** tab appears as follows:



As in Scenario 1, the Analysis dimension has only flat hierarchies, so the only selection rule available for an Analysis member is **Member**. IntOrg member 0500 is a roll-up member, and the formula is either a modeling formula or a driver formula (which run only at all-leaf crossings). This means that the only available member selection rule is **Leaf descendants of member, if any, otherwise Member**.

When the IntOrg member is changed to a leaf member, the member selection rule provided is still **Leaf descendants of member, if any, otherwise Member**, as shown here:



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:

Uses of the SUBSTR Function

Example: Bonus Expense = NESTIF

```
(SUBSTR(CURRENT("INTORG"),2,1)= "2", ["ACCOUNT"="R41000"]*.25,
(SUBSTR(CURRENT("INTORG"),2,1)= "1", ["ACCOUNT"="R41000"]*.20,
(SUBSTR(CURRENT("INTORG"),2,1)= "5", ["ACCOUNT"="R41000"]*.20,
1=1, ["ACCOUNT"="R41000"]*.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.

Member-Based Uses of the IF and NESTIF Functions

Example: Discounts = NESTIF

(["PRODUCT"="P00100"],["ACCOUNT"="Sales"]*-0.20,
["PRODUCT"="P00200"],["ACCOUNT"="Sales"]*-0.25,
["PRODUCT"="P00300"],["ACCOUNT"="Sales"]*-0.30)

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"="P00100"],["ACCOUNT"="Sales"]*-0.20,
["PRODUCT"="P00200"],["ACCOUNT"="Sales"]*-0.25,
["PRODUCT"="P00300"],["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: P00100
Expression: ["ACCOUNT"="Sales"]*-0.20
Scope: ["PRODUCT"="P00100"]

Formula 2

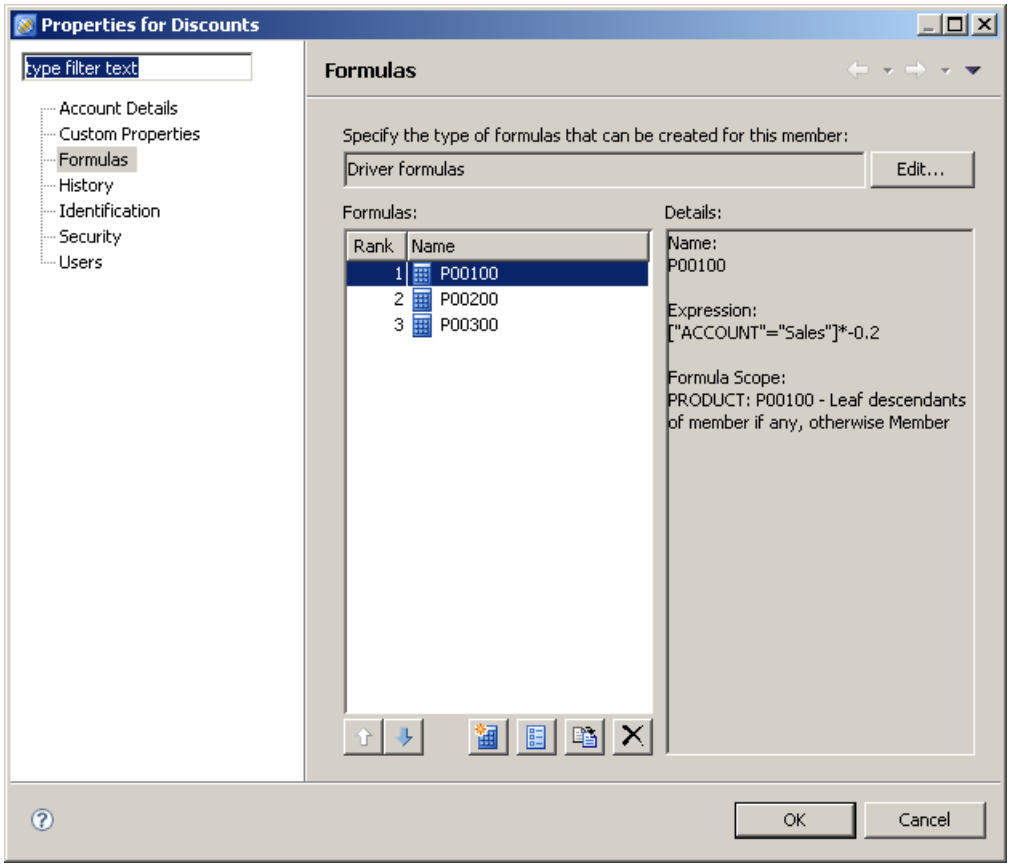
Name: P00200
Expression: ["ACCOUNT"="Sales"]*-0.25
Scope: ["PRODUCT"="P00200"]

Formula 3

Name: P00300
Expression: ["ACCOUNT"="Sales"]*-0.30
Scope: ["PRODUCT"="P00300"]

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 default 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:

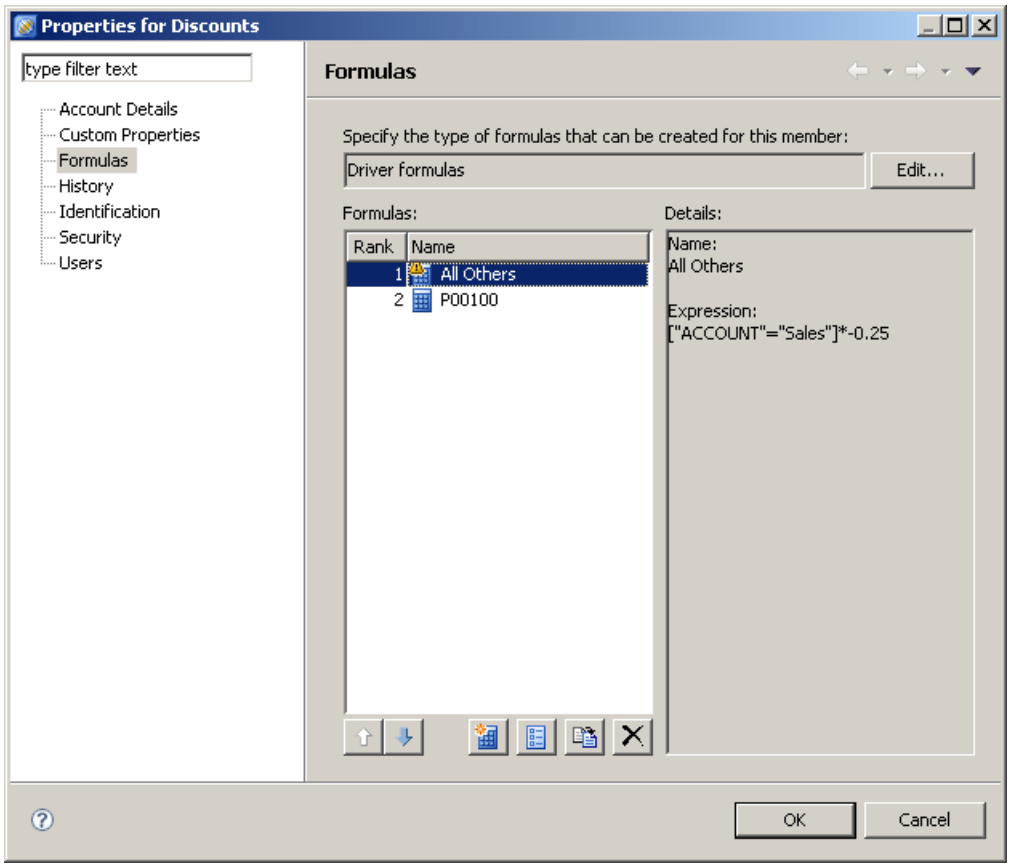
Formula 1

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

Formula 2

Name: P00100
Expression: ["ACCOUNT"="Sales"]*-0.20
Scope: ["PRODUCT"="P00100"]

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



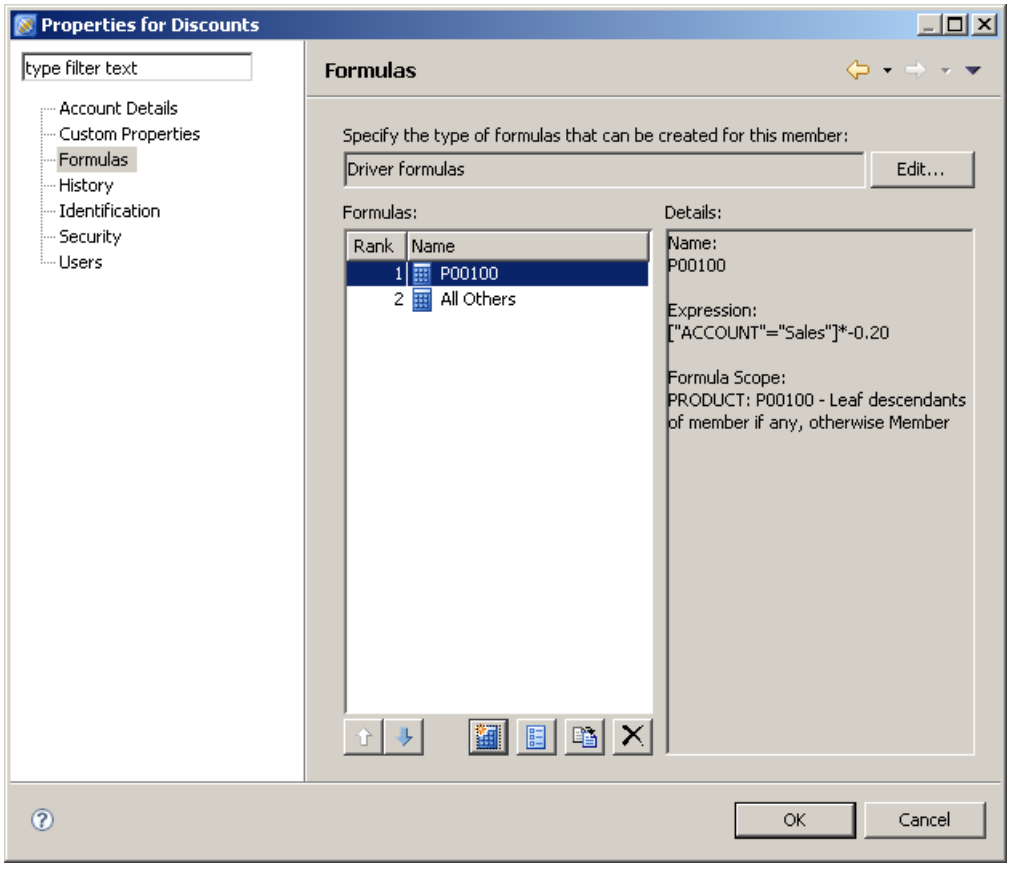
Results

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 ...	Eastern
Customer ...	Westco
Analysis ...	Budget
Currency ...	USD
Frequency ...	PTD
TIME ...	Q1 2008

	P00000 (Product Total)	P00100 (Video Games)	P00200 (Hardware)	P00300 (Publications)
Sales	(22,125.00)	(6,750.00)	(11,175.00)	(4,200.00)
Discounts	5,531.25	1,687.50	2,793.75	1,050.00
% Discount	0.25	0.25	0.25	0.25

When the formulas are rearranged so that the P00100 formula is ranked first, the warning message is not displayed.



Results

The following table displays the results when the P00100 formula is ranked first. This results in Discounts of 20% for all leaf descendants of P00100 and 25% for all other Product members.

Organization ...	Eastern
Customer ...	Westco
Analysis ...	Budget
Currency ...	USD
Frequency ...	PTD
TIME ...	Q1 2008

	P00000 (Product Total)	P00100 (Video Games)	P00200 (Hardware)	P00300 (Publications)
Sales	(22,125.00)	(6,750.00)	(11,175.00)	(4,200.00)
Discounts	5,193.75	1,350.00	2,793.75	1,050.00
% Discount	0.23	0.20	0.25	0.25

CHAPTER 7

Calculated Members Restrictions

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** and under **Details** when publishing an Operational Planning form set. 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** and under **Details** when publishing an Operational Planning form set. An invalid driver formula is displayed as a gray, non-writeable cell with a value of zero. 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** and under **Details** when publishing an Operational Planning form set. For both formula types, the result is a gray, non-writeable cell with a value of zero. 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.

R40000 (Staff Expenses)
R41000 (Salaries)
R42000 (Benefits)
R44000 (Travel)
R50000 (Other Selling Expenses)
R62000 (Gross Profit)

Driver Formula

R40000 = ["ACCOUNT" = "R62000"]* -0.2

Results

The following table displays the results. Because account R40000 (Staff Expenses) is a roll-up member, the formula is ignored and the leaf values are summed.

ORGANIZATION	Western U.S. Operations
Customer	Customer Total (vc)
Product	Product Total (vc)
Analysis	Budget
Currency	USD
FREQUENCY	PTD

	Jan 2009
R62000 (Gross Profit)	(100,000.00)
R40000 (Staff Expenses)	38,000.00
R41000 (Salaries)	30,000.00
R42000 (Benefits)	3,000.00
R43000 (Bonuses)	1,500.00
R44000 (Travel)	2,500.00
R50000 (Other Selling Expenses)	1,000.00

For Jan 2008, instead of the formula for member R40000 (Staff Expenses) executing as (100,000.00) * -0.2 to yield 20,000.00, the result is computed as 30,000.00 + 3,000.00 + 1,500.00 + 2,500.00 + 1,000.00 for a result of 38,000.00.

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, and under Details when publishing an OP form set:

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

Time Offsets: Formulas That Run Out of Bounds

Description

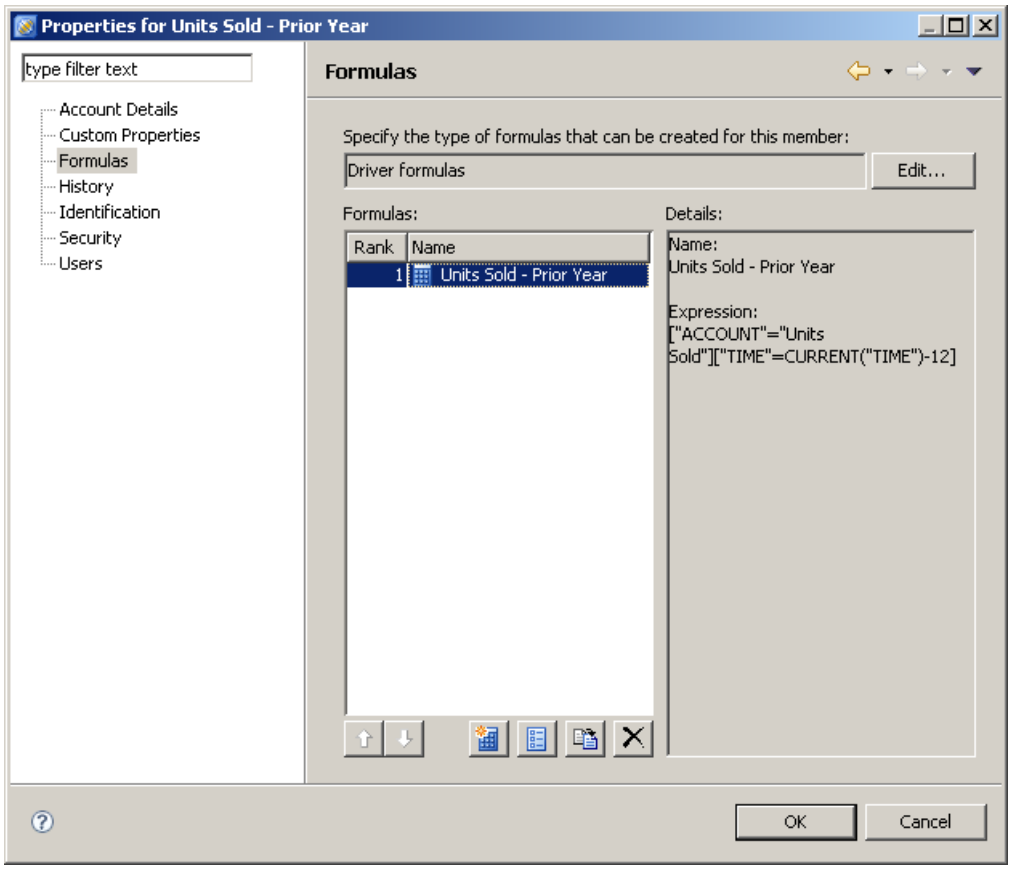
A formula referencing a time offset such as (["TIME"=CURRENT("TIME")-*x*] results in crossings where the formula runs are "out of bounds." This happens, because the formula is seeking inputs from crossings that do not exist. Crossings where formulas run out of bounds are deemed not applicable to the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

Affected Formula Types

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

Example

In the following example, Units Sold - Prior Year is 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 2008 is the first time member in the Time hierarchy. Based on the formula expression, results for time members from Jan 2008 through Dec 2008 return a zero. This is because their formula inputs (for example, Jan 2007 and Feb 2007) do not exist and are "out of bounds."

ORGANIZATION	Mexico Sales
Product	Flight Stick
Customer	SW Mart
Analysis	Budget
Currency	MXN
FREQUENCY	PTD

	Jan 2008	Feb 2008	Mar 2008	Q1 2008	Jan 2009	Feb 2009	Mar 2009	Q1 2009
Units Sold	50.00	75.00	100.00	100.00	125.00	150.00	175.00	175.00
Units Sold - Prior Year	0.00	0.00	0.00	0.00	50.00	75.00	100.00	100.00

Warning Message

No warning is provided.

Balance Account Calculated Members Scoped to Time

Description

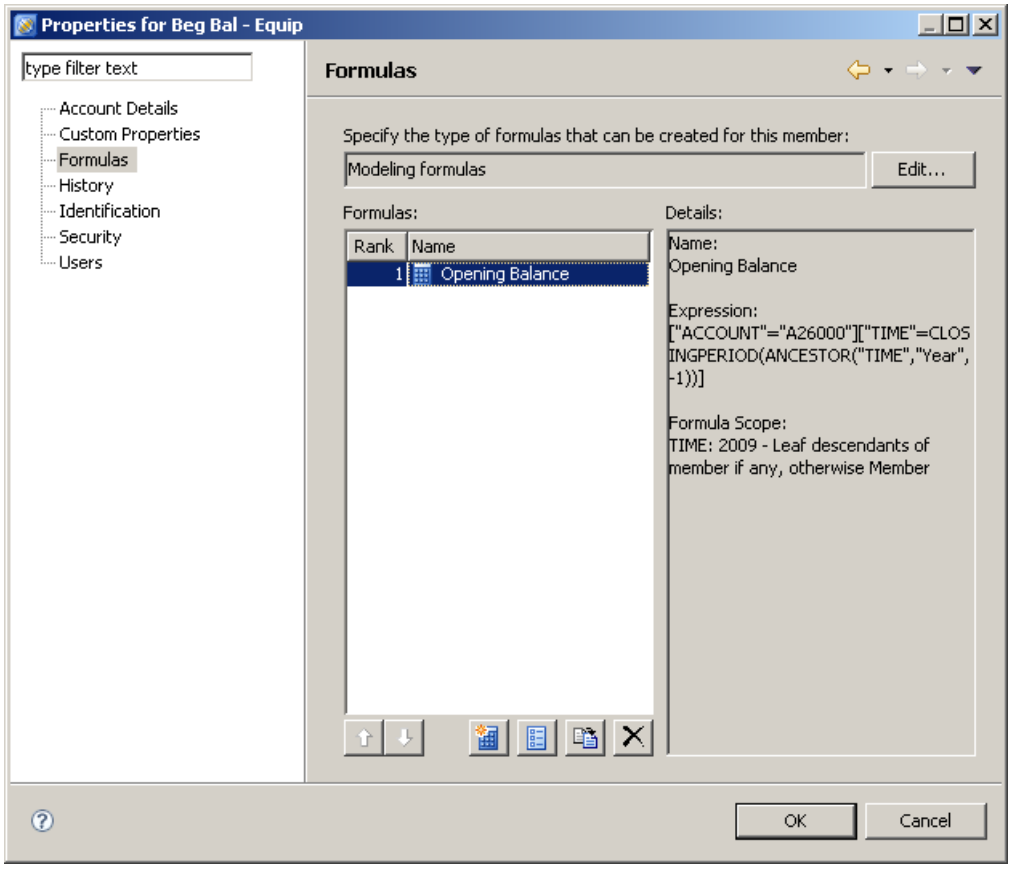
A calculated member on a balance account type scoped on the Time dimension is considered invalid. If the calculated member is assigned as a Driver, the result is a zero for the time periods in which it is scoped to and in any future time periods. If the calculated member is assigned as a Modeling formula, the result is a red cell (NaN) for the time periods in which it is scoped to 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 Beginning Balance - Equipment Account member is scoped to all leaf descendents of the Time member 2009. As a result, any Time periods in the formula scope as well as future time members return red cells (NaNs).

ORGANIZATION	Worldwide Operations
Product	Product Total
Customer	Customer Total
Currency	USD
FREQUENCY	PTD

	Actual	Budget			
	Dec 2008	Jan 2009	Feb 2009	Mar 2009	Q1 2009
Ending Balance, Equipment	450,000.00				
Beginning Balance - Equipment	0.00				
Additions, Equipment	450,000.00	10,000.00	20,000.00	30,000.00	30,000.00
Disposals, Equipment	0.00	0.00	0.00	0.00	0.00
Transfers & Writeoffs - Equipment	0.00	0.00	0.00	0.00	0.00

Warning Messages

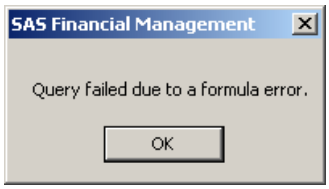
SAS Financial Management Studio

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**:**

```
A modeling or driver formula on a balance account cannot be scoped in
time.
```

Excel

For a Modeling formula, the following Cell Information is provided:



Referencing a Dimension and/or Member Not in the Model

Description

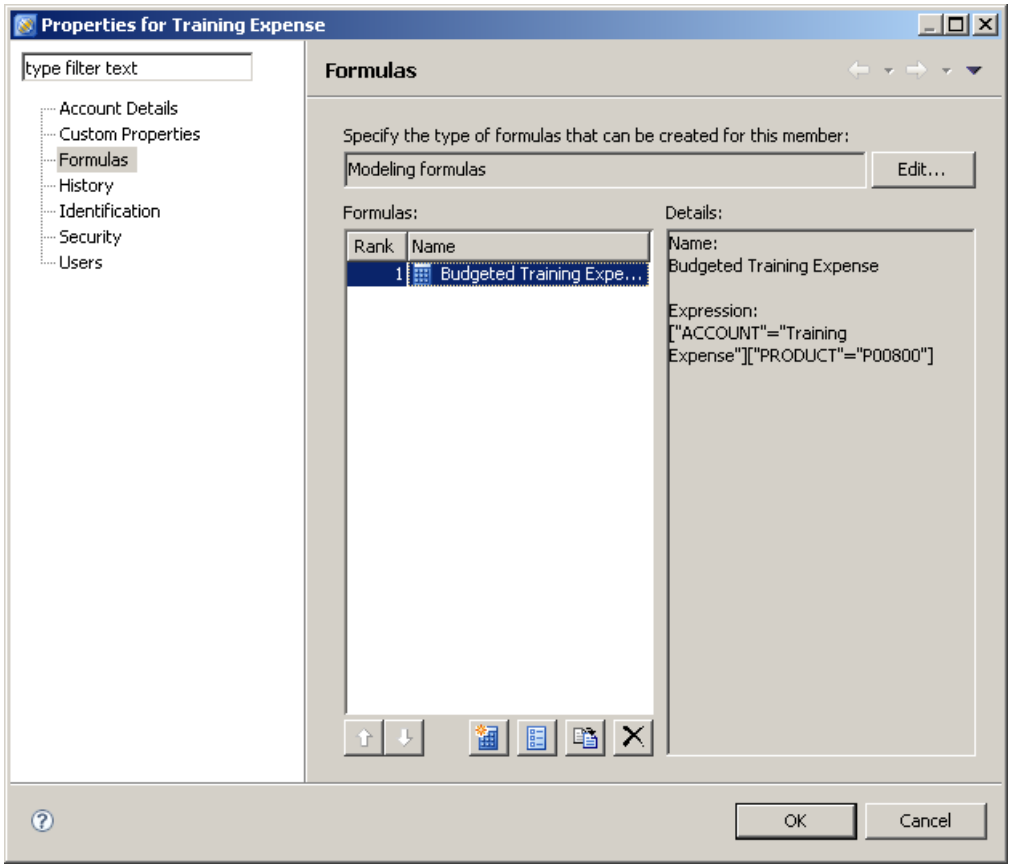
A calculated member that references a dimension and member that is not in the model is invalid. A calculated member that references a member that is not in the hierarchy is also invalid. If assigned as a Driver, the result is a zero. If the calculated member is assigned as a Modeling or Reporting formula,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

Affected Formula Types

- Driver
- Modeling
- Reporting

Example

In the following example, Account member Training Expense is a Modeling formula that references a Product member not in the hierarchy.



Results

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

ORGANIZATION	...	Worldwide Operations
Product	...	Product Total
Customer	...	Customer Total
Analysis	...	Budget
Currency	...	USD
FREQUENCY	...	PTD

	Oct 2009	Nov 2009	Dec 2009	Q4 2009
Training Expense				

Warning Messages

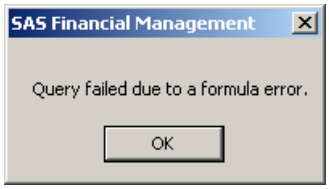
SAS Financial Management Studio

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

```
Formula references a member P00800 that is not in the model.
```

Excel

For a Modeling or Reporting formula, the following Cell Information is provided:



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

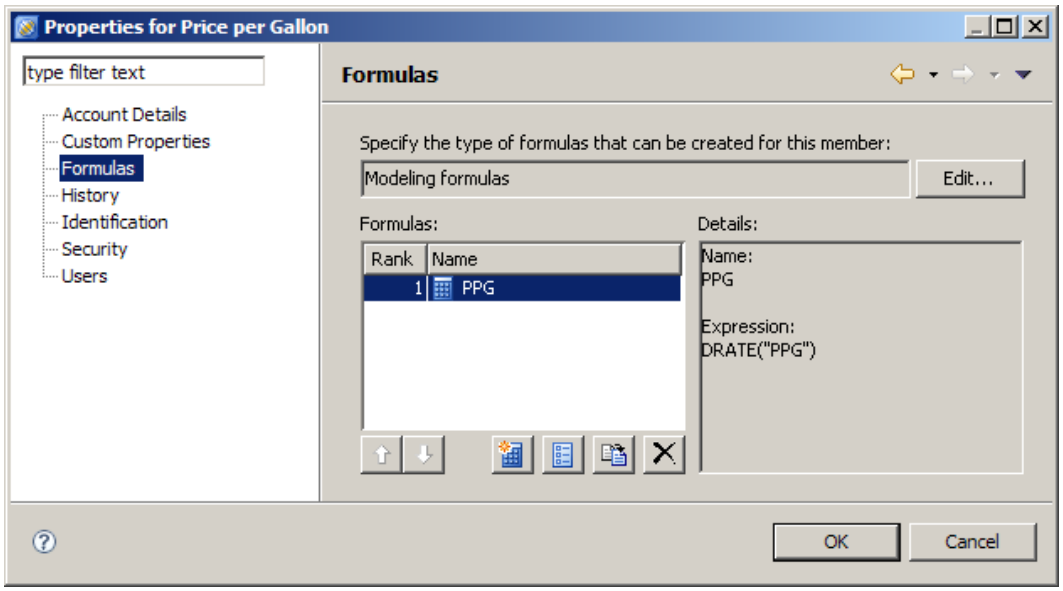
If assigned as either a Driver or Modeling formula, the displayed result is a gray, non-writeable cell with a value of zero.

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.

Analysis	Forecast
Frequency	PTD
Trader	External
Cost Center	DOD
Currency	USD
ORGANIZATION	Corporate Legal
Product	Action
Source	Form Data
	JAN2009
Price per Gallon	0.00

Warning Messages

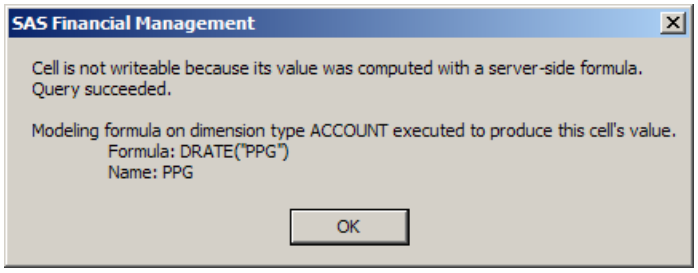
SAS Financial Management Studio

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

```
This constant formula expression is not supported as a modeling or driver
formula.
```

Excel

For a Modeling or Driver formula, the following Cell Information is provided:



Circular References

Description

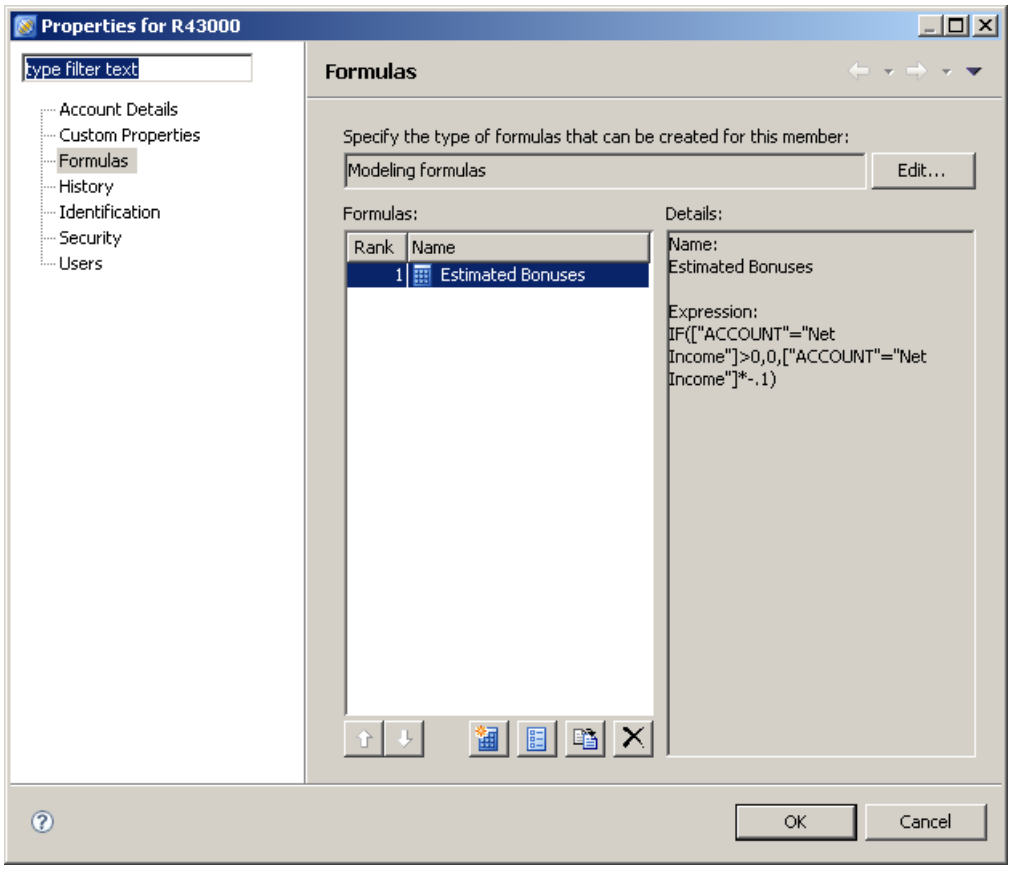
Calculated members with formula expressions containing circular references are invalid.

Affected Formula Types

- Driver
- Modeling
- Reporting

Example

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



Results

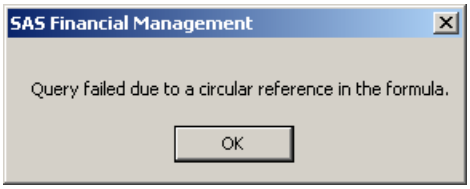
The following table displays the results of the circular reference in the Bonus account member. The circular reference affects all account members that it rolls up to.

ORGANIZATION ...	Central U.S. Operations
Product ...	Product Total
Customer ...	Customer Total
Analysis ...	Budget
Currency ...	USD
FREQUENCY ...	PTD

	Jan 2009
Net Income	
Income Taxes	0.00
Income before Taxes	
Gross Profit	0.00
Operating Expense	
Administrative Expenses	0.00
Marketing Expenses	0.00
Selling Expenses	
Staff Expenses	
Salaries	100,000.00
Benefits	30,000.00
Bonuses	
Travel	10,000.00
Other Selling Expenses	5,000.00
Other Operating Expenses	2,000.00

Warning Message for Excel

For a Modeling or Reporting formula, the following Cell Information is provided:



Dividing by Zero

Description

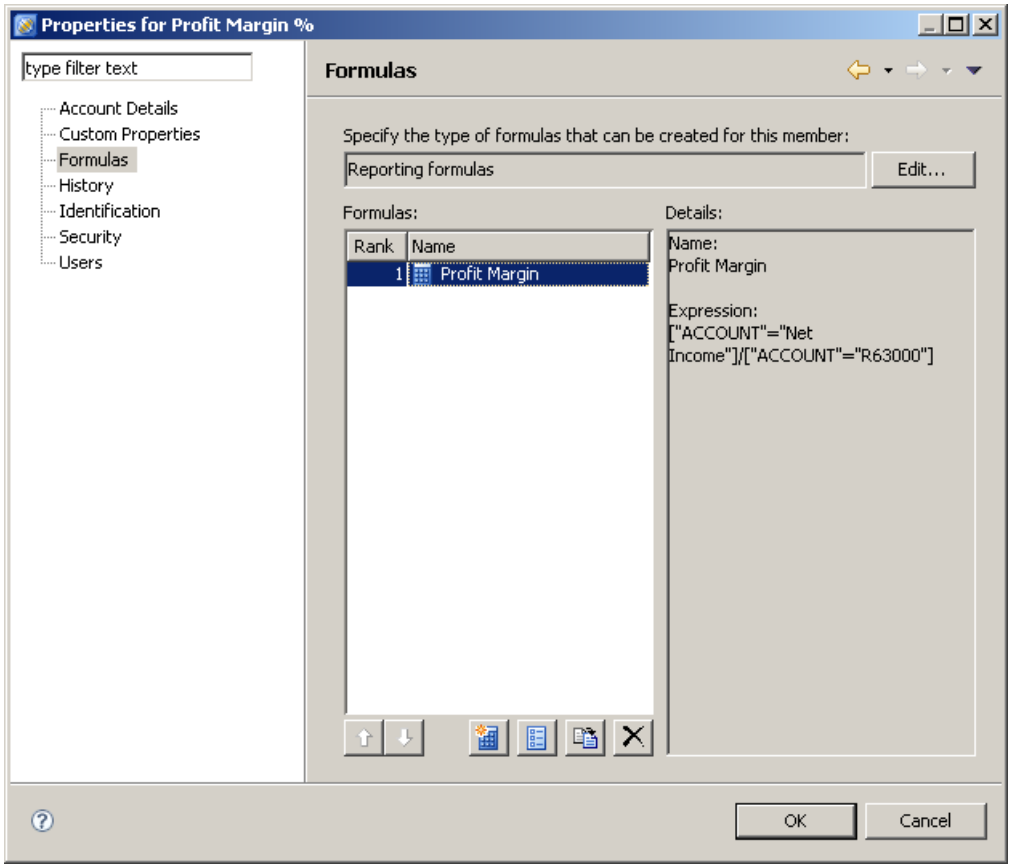
A calculated member where the divisor returns a value of zero is invalid.

Affected Formula Types

- Driver
- Modeling
- Reporting

Example

In the following example, Profit Margin % is calculated by dividing Net Income by Sales.



Results

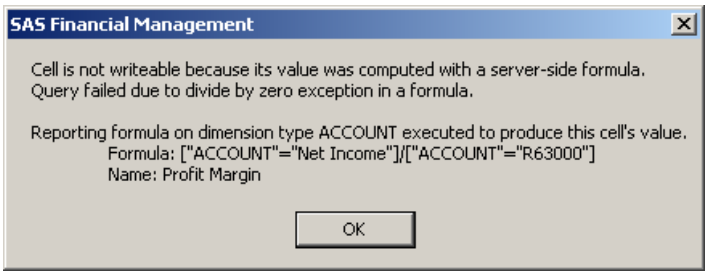
In the following table, Sales results have not been entered yet. Therefore, the formula currently is being divided by zero pending Sales account data input.

ORGANIZATION ...	Eastern U.S. Operations
Customer ...	Customer Total (vc)
Product ...	Product Total (vc)
Analysis ...	Budget
Currency ...	USD
FREQUENCY ...	PTD

	Jan 2009
Net Income	82,500.00
R57000 (Income Taxes)	0.00
R58000 (Income before Taxes)	82,500.00
R62000 (Gross Profit)	0.00
R63000 (Net Sales)	0.00
R64000 (Cost of Sales)	0.00
R26000 (Operating Expense)	82,500.00
Profit Margin %	

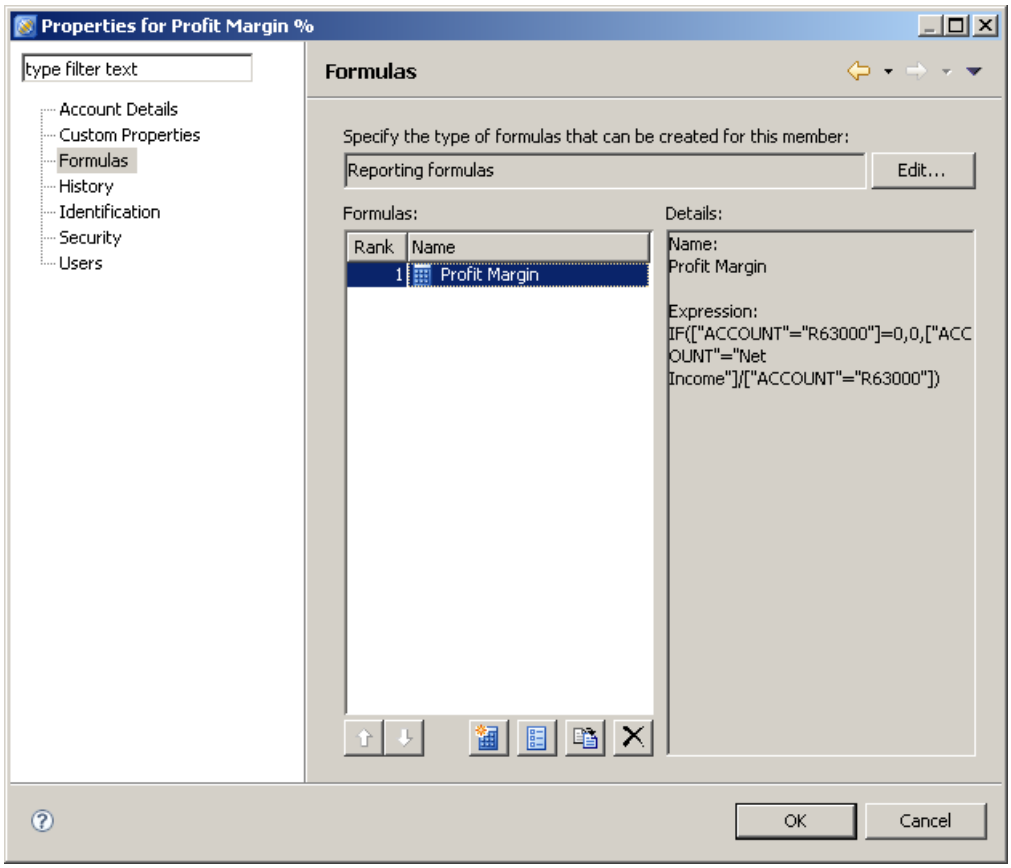
Warning Message for Excel

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 zero, the formula expression is no longer considered invalid.

The resulting table is as follows:

ORGANIZATION ...	Eastern U.S. Operations
Customer ...	Customer Total (vc)
Product ...	Product Total (vc)
Analysis ...	Budget
Currency ...	USD
FREQUENCY ...	PTD

	Jan 2009
Net Income (Net Income)	82,500.00
R57000 (Income Taxes)	0.00
R58000 (Income before Taxes)	82,500.00
R62000 (Gross Profit)	0.00
R63000 (Sales)	0.00
R64000 (Cost of Sales)	0.00
R26000 (Operating Expense)	82,500.00
Profit Margin % (Profit Margin %)	0.00

CHAPTER 8

Statistical Account Behavior

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

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:

Customer	Radio City
Analysis	Budget
Currency	USD
Frequency	PTD
TIME	Jan 2008
Account	Price
Source	Form Data
Trader	EXT

	How to use Simulator	Arcade Secrets	Puzzle Tricks
Dallas	15.00	20.00	10.00
San Antonio	16.00	18.00	9.00
Chicago	17.00	19.00	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).

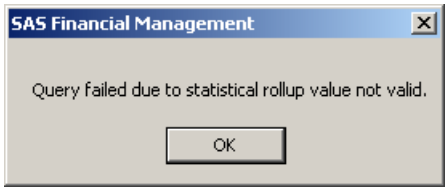
Customer	Radio City
Analysis	Budget
Currency	USD
Frequency	PTD
TIME	Jan 2008
Account	Price
Source	Form Data
Trader	EXT

	Publications	How to use Simulator	Arcade Secrets	Puzzle Tricks
Central				
Dallas		15.00	20.00	10.00
San Antonio		16.00	18.00	9.00
Chicago		17.00	19.00	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 “How to use Simulator”, “Arcade Secrets”, and “Puzzle Tricks” are leaf members. Organization members “Dallas”, “San Antonio”, and “Chicago” 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 “Publications” and Organization member “Central” are both roll-up members and therefore 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 shown in the previous section.

Organization	Dallas
Customer	Radio City
Analysis	Budget
Currency	USD
Frequency	PTD
Account	Price
Source	Form Data
Trader	EXT

	Jan 2008	Feb 2008	Mar 2008	Q1 2008
Publications				
How to use Simulator	15.00	15.50	16.00	
Arcade Secrets	20.00	20.00	20.50	
Puzzle Tricks	10.00	10.00	10.00	

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:

Organization	Dallas
Customer	Radio City
Product	Arcade Secrets
Analysis	Budget
Currency	USD
Account	Price
Source	Form Data
Trader	EXT

	Jan 2008	Feb 2008	Mar 2008	Q1 2008
PTD	20.00	20.00	20.50	
LTD	20.00	20.00	20.50	
YTD	20.00	20.00	20.50	
QTD	20.00	20.00	20.50	
MTD	20.00	20.00	20.50	
PA	20.00	20.00	20.50	
LA	20.00	20.00	20.50	
YA	20.00	20.00	20.50	
QA	20.00	20.00	20.50	
MA	20.00	20.00	20.50	

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	Eastern
Customer	Buy Best
Product	Action
Analysis	Budget
Currency	USD
Frequency	PTD
Source	Form Data
Trader	EXT

	Jan 2008	Feb 2008	Mar 2008	Q1 2008
Gross Margin	(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 Margin Percentage	0.40	0.43	0.41	0.42

In the table above, the yellow cells represent crossings in which all dimension members are at a leaf level.

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 Margin % = ["ACCOUNT"="Gross Margin"]/["ACCOUNT"="Sales"]

Gross Margin % is a Statistical account. Gross Margin and Sales are Revenue accounts.

Facts

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

Organization …	Eastern
Customer …	Buy Best
Product …	Action
Analysis …	Budget
Currency …	USD
Frequency …	PTD

	Jan 2008	Feb 2008	Mar 2008	Q1 2008
Gross Margin	(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 and Organization, then the results at roll-up members for the Gross Margin % account are displayed. Statistical accounts that are calculated members *do* display values at roll-up points, as shown here:

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

	Jan 2008	Feb 2008	Mar 2008	Q1 2008
Gross Margin	(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 Margin Percentage	0.40	0.43	0.41	0.42

Frequency Computation

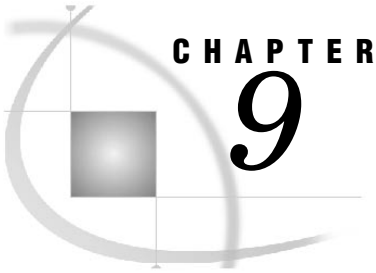
Statistical accounts that are calculated members also participate in frequency computation. To illustrate, consider the following table:

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

	Jan 2008	Feb 2008	Mar 2008	Q1 2008
PTD	0.40	0.43	0.41	0.42
LTD	0.40	0.42	0.42	0.42
YTD	0.40	0.42	0.42	0.42
QTD	0.40	0.42	0.42	0.42
MTD	0.40	0.43	0.41	
PA	0.40	0.43	0.41	0.42
LA	0.40	0.42	0.42	0.42
YA	0.40	0.42	0.42	0.42
QA	0.40	0.42	0.42	0.42
MA	0.40	0.43	0.41	

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, you might want to consider using a Statistical Balance account.



CHAPTER 9

Commonly Used Functions

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
Internal Organization	Worldwide Operations

	2008	Jan 2009	Feb 2009
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 & Writeoffs - 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 2008, which is the last leaf period of the prior year.

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

Analysis	Budget				
FREQUENCY	PTD				
Internal Organization	Worldwide Operations				
	Nov 2008	Dec 2008	2008	Jan 2009	Feb 2009
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.

Syntax

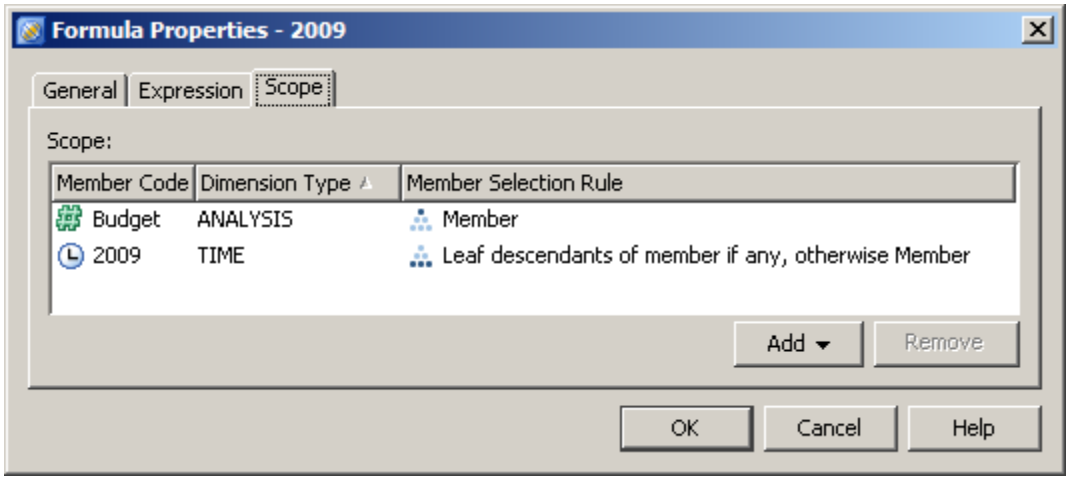
CURRENT(<dimensionTypeCode>)

Example

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 2009.

FREQUENCY	PTD			
Internal Organization	Worldwide Operations			
	Actual		Budget	
	Jan 2007	Feb 2008	Jan 2009	Feb 2009
Units Sold	8.00	9.00	8.80	9.90

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 Taxes =
IF(["ACCOUNT"="Income Before Taxes"]<0,["ACCOUNT"="Income Before Taxes"]*
DRATE("TaxRate2009")*-1,0)

In this formula, the DRATE function works on the Income Taxes 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 Income Taxes will be zero.

The DRATE for TaxRate2009 is displayed as the last row on this report by using an Excel calculated member with the following syntax:

=fmRate("TaxRate2009")

Analysis	Budget		
FREQUENCY	PTD		
Time Period	Jan 2009		
	Central	Eastern	Western
Net Income	(12.60)	5.00	(18.30)
Income Taxes	7.40	0.00	11.70
Income before Taxes	(20.00)	5.00	(30.00)
TaxRate2009	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 Period	Dec 2008		
	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

Variance to Budget =

NESTIF(PROPERTY("ACCOUNT","AccountType")="Expense",["ANALYSIS"="Budget"]-["ANALYSIS"="Actual"],PROPERTY("ACCOUNT","AccountType")="Revenue",["ANALYSIS"="Actual"]-["ANALYSIS"="Budget"])

This NESTIF statement works with the PROPERTY function on an Analysis member. The desired outcome is to return values in the Variance to Budget analysis member based on Account Type.

FREQUENCY	PTD
Internal Organization	Worldwide Operations
Time Period	Dec 2008

		Actual	Budget	Variance to Budget
Net Income	Revenue	32.00	29.00	3.00
Income Taxes	Expense	17.00	13.00	(4.00)
Income before Taxes	Revenue	49.00	42.00	7.00
Gross Profit	Revenue	120.00	110.00	10.00
Operating Expense	Expense	71.00	68.00	(3.00)
Administrative Expenses	Expense	0.00	14.00	14.00
Marketing Expenses	Expense	15.00	0.00	(15.00)
Selling Expenses	Expense	21.00	23.00	2.00
Staff Expenses	Expense	21.00	23.00	2.00
Other Operating Expenses	Expense	35.00	31.00	(4.00)

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
Internal Organization	Worldwide Operations
FREQUENCY	PTD

	Jan 2008	Feb 2008	Mar 2008
	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
Internal Organization	Central U.S. Operations
Time Period	Jan 2009

	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>)

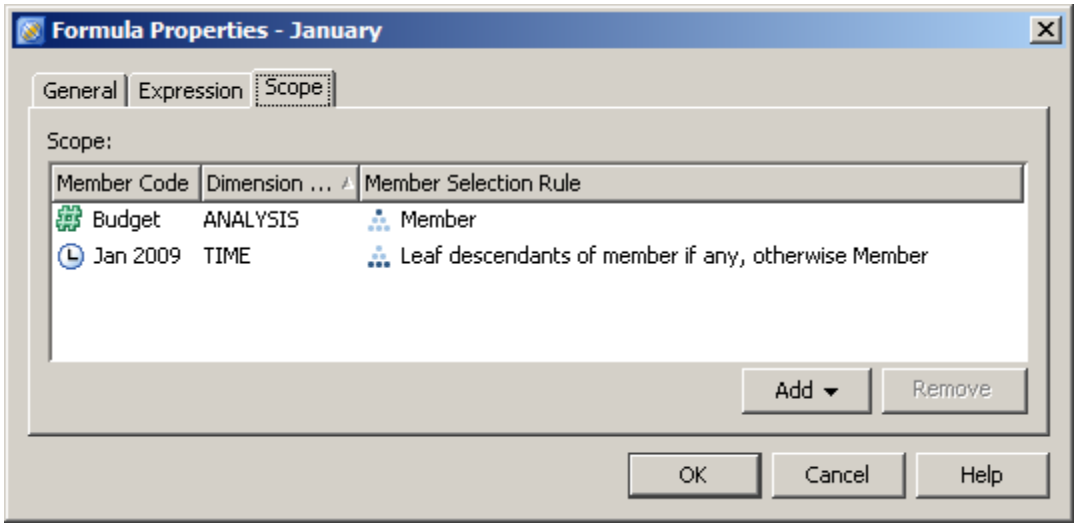
Example

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.

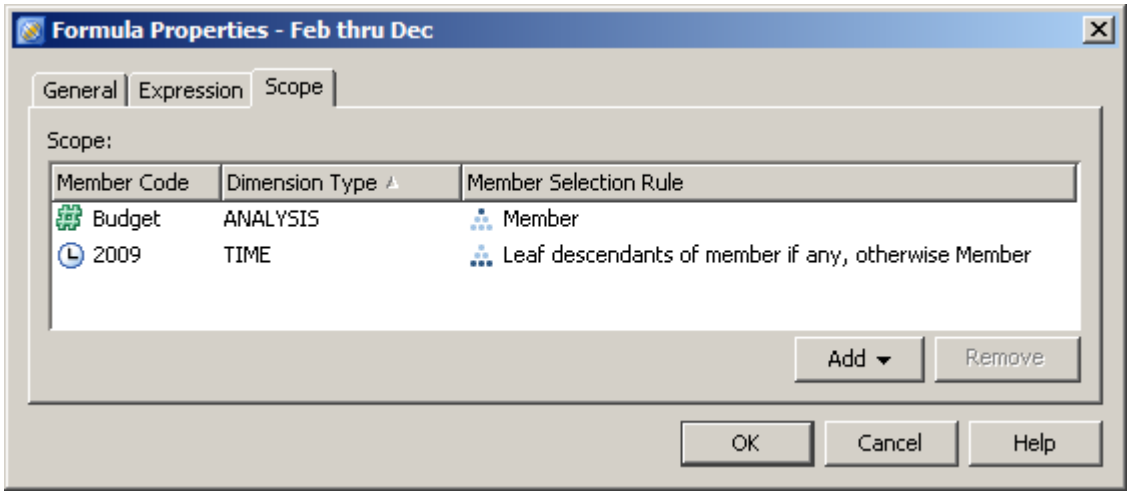


The expression ranked second is named “Feb thru Dec”.

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 2009, 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
Internal Organization Worldwide Operations

	Actual	Budget		
	Dec 2008	Dec 2008	Jan 2009	Feb 2009
Total Salaries	100.00	80.00	105.00	115.00
Salaries	100.00	80.00	0.00	0.00
Previous Period Salaries	0.00	0.00	100.00	105.00
Change 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.

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

Variance to Forecast =

IF((PROPERTY("ACCOUNT","AccountType")="Asset")

OR

(PROPERTY("ACCOUNT","AccountType")="Liability") ,

["ANALYSIS"="Actual"]-["ANALYSIS"="Forecast"],

["ANALYSIS"="Forecast"]-["ANALYSIS"="Actual"])

In this formula, the PROPERTY function works with the IF function on an Analysis member. The desired outcome is to calculate values in a Variance column based on the Account Type and Balance Type.

FREQUENCY	PTD
Internal Organization	Western U.S. Operations
Time Period	Jan 2009

			Actual	Forecast	Variance to Forecast	
Assets	Asset	Debit	22.00	22.50	(0.50)	Unfavorable
Liabilities	Liability	Credit	(8.00)	(6.00)	(2.00)	Unfavorable
Equity	Equity	Credit	(14.00)	(16.50)	(2.50)	Unfavorable
Net Sales	Revenue	Credit	(12.00)	(9.00)	3.00	Favorable
Operating Expense	Expense	Debit	10.00	8.00	(2.00)	Unfavorable

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)

Analysis	...	Budget		
Currency	...	USD		
Time Period	...	Jan 2008		

	Sales	Central U.S. Operations	Eastern U.S. Operations	Western U.S. Operations
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	9,223,372,036,854,780,000.00	

When a formula that uses the ROUND function encounters a value divided by zero, the result is the largest long value in Java (9223372036854780000). (See the Revenue per Employee formula crossing with Eastern U.S. Operations.) When a formula encounters a zero divided by zero, a red cell is displayed (NaN) with Cell Information. The information states that the query failed due to an unknown reason. (See the Revenue per Employee formula crossing with Western U.S. Operations.)

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)

Analysis	Budget
Currency	USD
Time Period	Jan 2008

	Sales	Central U.S. Operations	Eastern U.S. Operations	Western U.S. Operations
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)

Analysis	Budget
Currency	USD
Time Period	Jan 2008

	Sales	Central U.S. Operations	Eastern U.S. Operations	Western U.S. Operations
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"="JAN2005"]:["TIME"="JUN2005"])

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>)

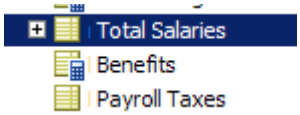
Example

Salaries and Benefits =

SUM(["ACCOUNT"="Total Salaries"]:["ACCOUNT"="Payroll Taxes"])

In this formula, the SUM function is used in the Salaries and Benefits account member. The desired outcome is to sum the accounts from Total Salaries through Payroll Taxes.

The relevant range of the Account hierarchy is shown below.



Analysis	Forecast
FREQUENCY	PTD
Internal Organization	Worldwide Operations

	Jan 2009
Total Salaries	6.00
Benefits	2.00
Payroll Taxes	1.00
Salaries and Benefits	9.00

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

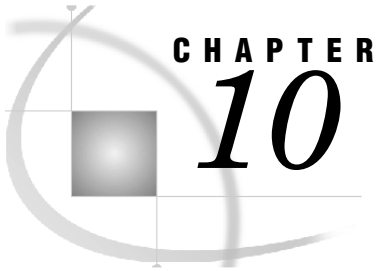
See PARENT.

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 */`**

To illustrate:

```
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 */
```



CHAPTER 10

Interactions with Load and Delete Data Options

Options for Loading Data

Introduction

There are three 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 operational data to this cycle

Load new data to this cycle Option

This method is available for both Financial and Operational Planning cycles. Use this option to load data that is stored in the Detail Data Store. 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

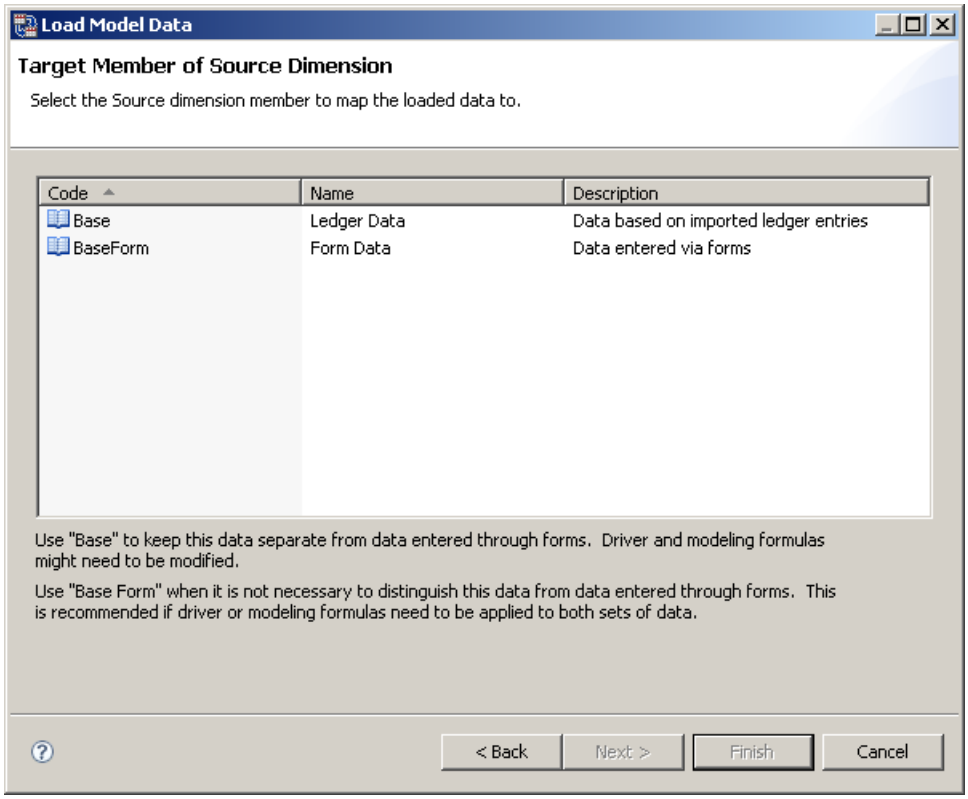
Available only for Financial cycles, this option is used to load data stored in the SDM. In addition to specifying the Time and Analysis dimension, the data load can be subset to a select group of members for the IntOrg, Account, and Custom dimensions. Loading can be either to the Base or BaseForm Source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

Load operational data to this cycle Option

Available only for Financial cycles, this option is used to load data from an Operational cycle to a Financial cycle. The Account dimension in the Operational cycle must be the same as the Account dimension used in the Financial cycle. The load is subset by the Time and Analysis dimension members specified. Data loaded from an Operational Planning cycle to a Financial Cycle is always loaded to the BaseForm Source member. Member mapping is optional for Time and Analysis. The mapping is done systematically for other dimensions when necessary. When Operational Planning dimensions contain larger, more detailed dimension hierarchies than the Financial Cycle's dimensions, systematic mapping occurs during the load process.

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 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 DDS 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"]*-1

Formula Scope

["ANALYSIS"="Actual"]

["ANALYSIS"="Budget"]

["ANALYSIS"="Forecast"]

Facts

For the Forecast Analysis member, facts from the Jul 2008 through Dec 2008 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 U.S. Operations
Product	Arcade
Customer	HAL
Analysis	Budget
Currency	USD
FREQUENCY	PTD
ORGANIZATION_TRADER	EXT
Source	Form Data

	Jul 2008	Aug 2008	Sep 2008	Q3 2008
Price	10.00	10.00	10.00	
Units	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 are modified via Data Entry to the following values:

- Jul 2008: 125
- Aug 2008: 150
- Sep 2008: 175

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

ORGANIZATION	Eastern U.S. Operations
Analysis	Forecast
Product	Arcade
FREQUENCY	PTD
Customer	HAL
Currency	USD
ORGANIZATION_TRADER	EXT
Source	Total

	Jul 2008	Aug 2008	Sep 2008	Q3 2008
Units	125.00	150.00	175.00	450.00

Results

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

Jul 2008: 10.00 * 125.00 * -1 = (1,250.00)

Aug 2008: 10.00 * 150.00 * -1 = (1,500.00)

Sep 2008: 10.00 * 175.00 * -1 = (1,750.00)

Q3 2008: (1,250.00) + (1,500.00) + (1,750.00) = (4,500.00)

The read-only table displays the following results:

ORGANIZATION	Eastern U.S. Operations
Product	Arcade
Customer	HAL
Analysis	Forecast
Currency	USD
FREQUENCY	PTD

	Jul 2008	Aug 2008	Sep 2008	Q3 2008
Price				
Units	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 account through the Form are automatically stored in the BaseForm Source member.

The contributing data records for the Units account for the Total Source member for Jul 2008 show two things:

- that a value of 100 is in the Base Source member
- a net value of 25 is in the BaseForm source member

TIME	SOURCE	PRODUCT	Custome	OBJECT_TYPE	OBJECT	Value	USD converted
Jul 2008	Base	P00600	C00500	dataload	28000	100.00	100.00
Jul 2008	BaseForm	P00600	C00500	formset	Copy of Formula Doc Formset	(100.00)	(100.00)
Jul 2008	BaseForm	P00600	C00500	formset	Copy of Formula Doc Formset	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 Jul 2008 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 Account member were stored and the values for Jul 2008 Sales are calculated:

ORGANIZATION	Eastern U.S. Operations
Product	Arcade
Customer	HAL
Analysis	Forecast
Currency	USD
FREQUENCY	PTD
ORGANIZATION_TRADER	EXT
Time	Jul 2008

	Total	TotalBeforeElim	TotalBeforeAdj	BaseForm	TotalAfterImport	Base
Price				0.00		10.00
Units	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:

Jul 2008: (10.00 * 100.00 * -1) + (0.00 * 25.00 * –1) = (1,000.00)

Aug 2008: (10.00 * 125.00 * -1) + (0.00 * 25.00 * –1) = (1,250.00)

Sep 2008: (10.00 * 150.00 * -1) + (0.00 * 25.00 * –1) = (1,500.00)

Q3 2008: (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"]+["SOURCE"="BaseForm"]["ACCOUNT"=
"Units"])*-1
```

Formula Scope

["SOURCE"="BaseForm"]

The results for Jul 2008 based on the revised formula expression are as follows:

ORGANIZATION	Eastern U.S. Operations
Product	Arcade
Customer	HAL
Analysis	Forecast
Currency	USD
FREQUENCY	PTD
ORGANIZATION_TRADER	EXT
Time	Jul 2008

	Total	TotalBeforeElim	TotalBeforeAdj	BaseForm	TotalAfterImport	Base
Price				0.00		10.00
Units	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:

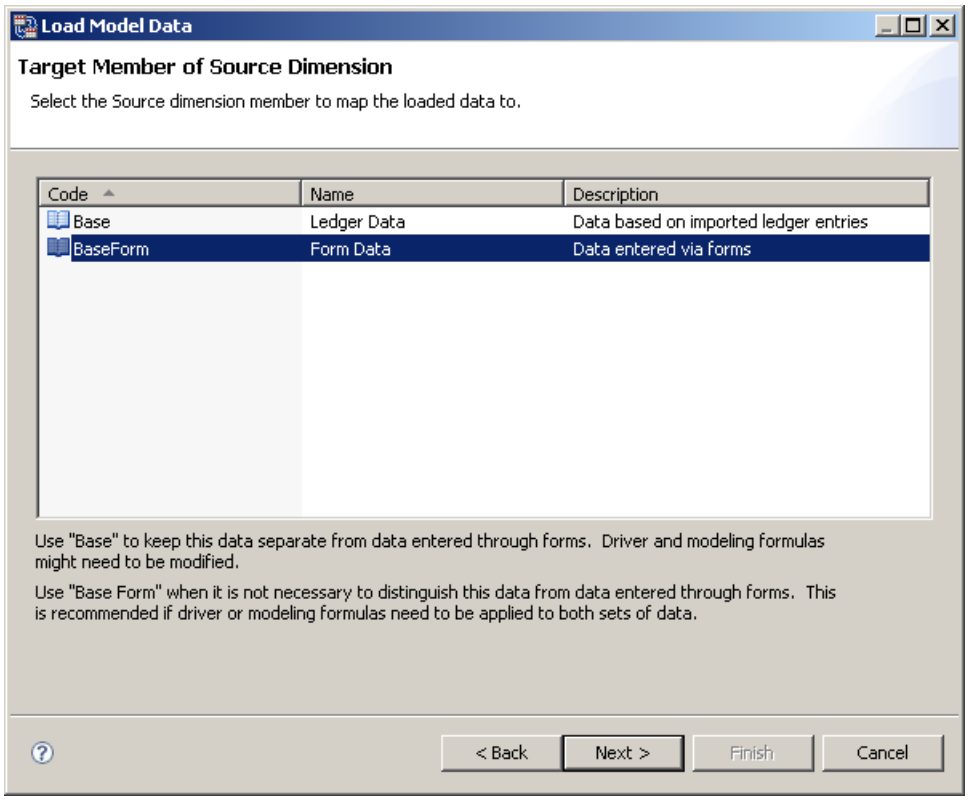
ORGANIZATION	Eastern U.S. Operations
Product	Arcade
Customer	HAL
Analysis	Forecast
Currency	USD
FREQUENCY	PTD
ORGANIZATION_TRADER	EXT
Time	Jul 2008

	Total	CTA	Elim	TotalBeforeElim	Adj	TotalBeforeAdj	BaseForm	TotalAfterImport	BaseJourn	Base
Price		0.00	0.00				0.00		0.00	10.00
Units	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 the Detail Data Store or a model separate from data that is entered through Forms.

Scenario 2: Selecting BaseForm Source Member

If you do not have to keep data that is loaded from the Detail Data Store or a model separate from data enter 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"]* –1

Formula Scope

["ANALYSIS"="Actual"]

["ANALYSIS"="Budget"]

["ANALYSIS"="Forecast"]

Facts

For the Forecast Analysis member, facts from the Jul 2008 through Dec 2008 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 U.S. Operations
Product	Puzzle
Customer	Buy Best
Analysis	Budget
Currency	USD
FREQUENCY	PTD
ORGANIZATION_TRADER	EXT
Source	Form Data

	Jul 2008	Aug 2008	Sep 2008	Q3 2008
Price	15.00	15.00	15.00	
Units	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:

- Jul 2008: 100
- Aug 2008: 200
- Sep 2008: 300

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

ORGANIZATION	Eastern U.S. Operations
Product	Puzzle
Customer	Buy Best
Analysis	Forecast
Currency	USD
FREQUENCY	PTD
ORGANIZATION_TRADER	EXT
Source	Form Data

	Jul 2008	Aug 2008	Sep 2008	Q3 2008
Units	100.00	200.00	300.00	600.00

Results

The expected results for Sales are computed as follows:

Jul 2008: 15.00 * 100.00 * –1 = (1,500.00)

Aug 2008: 15.00 * 200.00 * –1 = (3,000.00)

Sep 2008: 15.00 * 300.00 * –1 = (4,500.00)

Q3 2008: (1,500.00) + (3,000.00) + (4,500.00) = (9,000.00)

The read-only table displays the following results:

ORGANIZATION	...	Eastern U.S. Operations
Product	...	Puzzle
Customer	...	Buy Best
Analysis	...	Forecast
Currency	...	USD
FREQUENCY	...	PTD
ORGANIZATION_TRADER	...	EXT
Source	...	Form Data

	Jul 2008	Aug 2008	Sep 2008	Q3 2008
Price	15.00	15.00	15.00	
Units	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 based on the fact that all 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 account for Jul 2008 are as follows:

TIME	SOURCE	PRODUCT	Custome	OBJECT_TYPE	OBJECT	Value	USD converted
Jul 2008	BaseForm	P00700	C00800	dataload	29100	200.00	200.00
Jul 2008	BaseForm	P00700	C00800	formset	Copy of Formula Doc Formset	(200.00)	(200.00)
Jul 2008	BaseForm	P00700	C00800	formset	Copy of Formula Doc Formset	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, you should consider the following series of questions before you load data from a model:

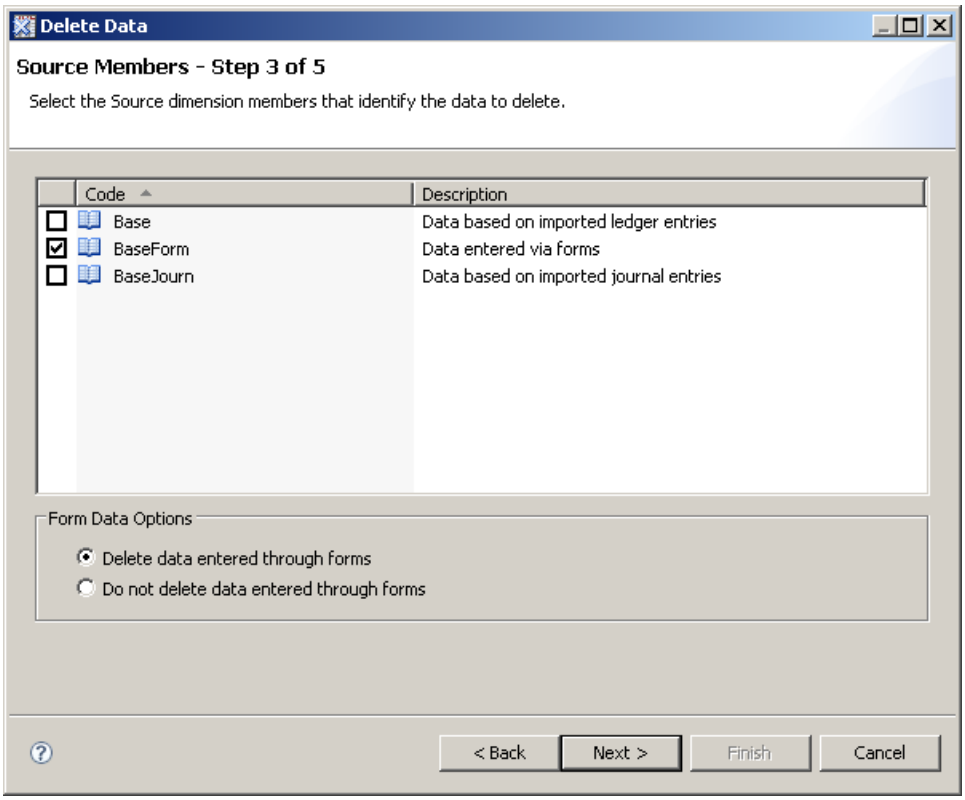
- Does the model contain driver formulas? If it does, continue to the next question. If not, proceed to load the data.
- Will the data be loaded to the Base member or the BaseForm member of the Source dimension? If to Base, continue to the next question. If to BaseForm, proceed to load data.
- Are any 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

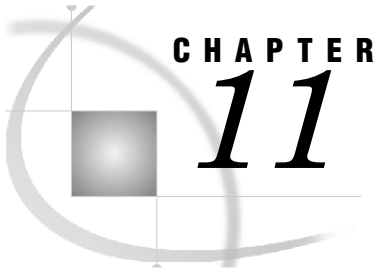
BaseForm member of the Source dimension. This selection allows users to modify the results loaded within the same Source member. In contrast, driver formula facts loaded to the Base member cannot be modified in the Base Source member.

Deleting Data and Driver Facts

Delete data from this cycle is available for both Financial and Operational cycles. Similar to the load wizards, data can be deleted by a subset of members for the Time, Analysis, IntOrg, Trader, Account, and Custom dimensions. For deletion of Financial 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.



For Operational Planning, since all facts are loaded to the BaseForm Source member, the selection of Source members is not applicable.



CHAPTER 11

Using Driver Formulas in Financial Cycles

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, including system filters.

Note that *all* driver formulas in the hierarchy are executed, regardless of whether they are included on the form.

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.

Note: It is unnecessary to enter a value for the formula input in order to trigger execution of the formula. It is sufficient to enter a value at *any* crossing that matches the formula output crossing in all dimensions except Time and Account.

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:

- 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 default 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. However, it is not necessary for the account that is the driver formula member to be in the data-entry table. Every other dimension member of a driver formula output crossing must be on the data-entry table, but the driver formula member itself need not be.
- 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.
- To render a result at the time of data entry, driver formula input crossings must match driver formula result crossings for all dimensions. The only exceptions are the Time dimension (for relative time addressing) and the Account dimension. This means that an input crossing that is not identical to the output crossing does not trigger a driver formula at the time of data entry. This applies to formulas that either explicitly state dimensions other than Time and/or Account in the expression or reference them via Default Members.

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
- driver formula expressions referencing dimensions other than Account and/or Time
- 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 System Filters to restrict the number of crossings required to run driver formulas.

Note: *All* driver formulas in the hierarchy are executed, regardless of whether they are included on the form.

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.

Relative Time References

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, but can write only to members that are included in the form.

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

This chapter describes factors that affect the performance of driver formulas and modeling formulas. As described earlier, there are significant differences between driver formulas and modeling formulas that concern how often a formula runs and at how many crossings.

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 192,000 writable crossings.

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. There are 50 driver formulas in this model. When a value is entered into a form for a single crossing, this triggers all driver formulas in this model for the crossing for which the value was entered. In this example, the number of accounts increases from 30 (in the form set) to 50 (the number of driver formulas in the model). The driver formulas will be executed for all writable time periods on the form (12 in this example). All other dimensions are limited to the single member for which a value was entered. In this example, entering a value in a form triggers the execution of driver formulas for a total of 600 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 192,000 crossings: the number of writable crossings for one form (12 Time periods * 20 Cost Centers) times the number of driver formulas (50) times the number of target members (16).

Dimension	Create Form Template (writable crossings for one Form below)	Driver execution per Entered Value	Run Driver Formulas for Formset	
Account	30	50	50	
Time	12	12	12	
Intorg	1	1	16	
Cost Center	20	1	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	600	192,000	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:

- number of driver formulas in the Account hierarchy
- number of writable Time periods on the form

The following factors affect the number of crossings where a driver formula runs when you select the **`Run driver formulas for this form set`** option:

- number of driver formulas in the Account hierarchy
- number of writable crossings in the form set for all other dimensions

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 System Filters 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	600
Use "Run driver formulas for this form set" action	N/A	192,000

Managing Formulas for Performance

There are several factors that can affect the query and execution time for Modeling and Driver formulas. Formula performance is largely attributed to the following:

- Model Size
- Formula Scope
- Number of formula expression inputs
- Use of the SUBSTR function
- Ability to use Distributive Optimization

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.

Information about model size can be found in the ODCS Diagnostic Servlet. In the following example, the model contains over 161 trillion crossings, computed as 83*21*6*45*45*9189*118*7.

Home - Cache - Cube - Dispatcher - History - Log - MDX - Log Out

163012 - ChemicalCoModel - ChemicalCoModel | Display

163012:ChemicalCoModel

facts - formulas - xrates - prates

DimType	Role	DimTypeIntId	Dim	DimIntId	Hier	DefMember	Member Count	asOf
0/1 ACCOUNT	1 - ACCOUNT	-1	46803 ACCOUNT	-46803	30515 ACCOUNT_HIER	57/286774 A10000	83	20091209 10:50:01:000
1/13 SOURCE	14 - SOURCE	-13	5 SOURCE	-5	4 SOURCE	20/500 Total	21	20091209 10:50:01:000
2/2 ANALYSIS	3 - ANALYSIS	-2	64029 ANALYSIS	-64029	42275 Formula Doc	1/34675 Budget	6	20091209 10:50:01:000
3/6 INTORG	7 - INTORG	-6	46804 ORGANIZATION	-46804	30512 Formula Doc	41/286775 0000	45	20091209 10:50:01:000
4/10 TRADER	12 - TRADER	-6	46805 ORGANIZATION_TRADER	-46804	30512 Formula Doc	44/25838 ALL	45	20091209 10:50:01:000
5/11177 PRODUCT	99 - Other	-11177	432963 EVENT	-432963	288967 Event	9186/2839878 Total Events	9189	20091209 10:50:01:000
6/9 TIME	11 - TIME	-9	46824 TIME	-46824	30537 TIME_MR	12/287492 YR1997	118	20091209 10:25:54:000
7/4 CURRENCY	5 - CURRENCY	-4	2 CURRENCY	-2	2 CURRENCY	0/8366 USD	7	20091209 10:50:01:000
8/15 FREQUENCY	18 - FREQUENCY	-15	7 FREQUENCY	-7	6 FREQUENCY	0/544 PTD	19	20091209 10:50:01:000
9/14 XRATE_TYPE	17 - XRATE_TYPE	-14	6 XRATE_TYPE	-6	5 XRATE_TYPE	5/526 PeriodOpen	8	20091209 10:50:01:000
10/12 RATE_TYPE	15 - RATE_TYPE	-12	4 RATE_TYPE	-4	3 RATE_TYPE	-1/-1 Unknown	13	20091209 10:50:01:000
11/11 TO_CURRENCY	16 - TO_CURRENCY	-4	3 TO_CURRENCY	-2	2 CURRENCY	-1/-1 Unknown	7	20091209 10:50:01:000

Total Size of QueryCube with cardinalities 83,21,6,45,45,9189,118,7 = 160739259729300

In computing total number of crossings, note that the Frequency dimension is not included in the calculation.

In an effort to reduce model size, the use of Custom Properties is recommended as an alternative to 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 EVENT custom dimension a Custom Property of the INTORG dimension results in an adjusted cube size of approximately 17 billion crossings, computed as 83*21*6*45*45*118*7. That's a reduction of more than 161 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 original ChemicalCo model size of 161 trillion crossings. Scoping for an individual modeling formula on the Source, Trader, and Analysis dimensions reduces the number of crossings from 161 trillion to approximately 341 million! Information from the Cube view of the ODCS Diagnostic Servlet is displayed below:

Formula for VCubeId 163012:

Id	Type	TargetDimType	TargetMember	Text	CellCount	ScopeParser
342423	Leaf	0: ACCOUNT	72: S1000	["ACCOUNT"='R27000']+['ACCOUNT'='R51000']+ ["ACCOUNT"='R39000']	341555130	FormulaScopeParser: {[TRADER].[EXT].leaves} * {[SOURCE].[BaseForm].leaves} * {[ANALYSIS].[Budget].leaves}

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" 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.

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 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 default 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.

System Filters

The number of cells for which the "Run driver formulas for this form set" action executes can be reduced dramatically by System Filters. The "Source" dimension in the System Filters should be used as the Target Hierarchy for the form set. The following table demonstrates the impact.

INTORG is the Target Hierarchy for the Formset
INTORG is the Source dimension in System Filters; Location is the Target dimension
System Filters are defined such that each INTORG member is linked to four Location members

Dimension	Form's Writable Crossings without System Filters Enabled	Form's Writable Crossings with System Filters 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 in Account hierarchy	50	50
Total Cell Executions per Form	132,000,000	264,000
Number of Forms	500	500
Total Number of Cells Executed	66,000,000,000	132,000,000

Reduction of >99%

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 is 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

Using Formulas in Operational Planning Cycles

Supported Formula Types

Operational Planning supports two formula types:

- driver formulas
- Microsoft Excel-based calculated member formulas

In order to be valid for Operational Planning, driver formulas cannot contain references to any dimensions other than Account and/or Time. This applies to formula expressions that are either explicitly stated or assigned by using Default Members in the Formula Editor.

Note: These restrictions apply only to the formula expression. They do not apply to Formula scope.

In addition to Driver formulas, all Excel-based calculated member functions are valid with Operational Planning.

Using Drivers in Operational Planning

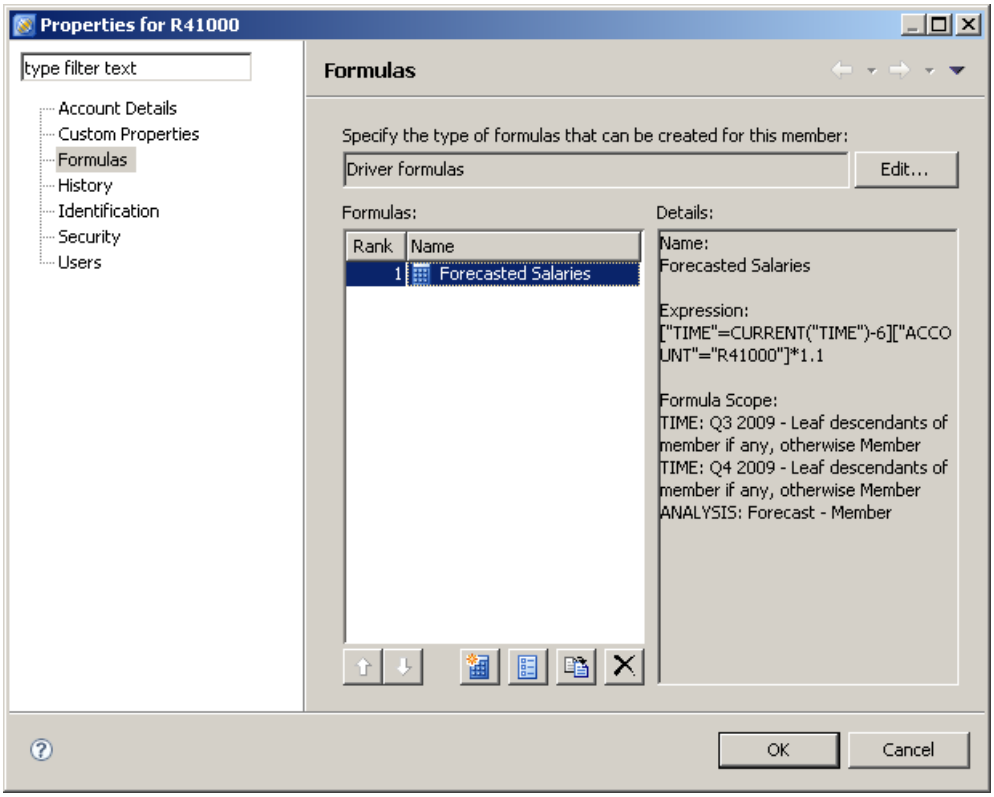
Introduction

The following examples illustrate driver formulas that are valid in Operational Planning.

Example 1: Forecasted Salaries

The following formula looks at prior period results for salaries and increases the amounts by ten percent for the periods of July through December 2009 to estimate a forecasted salary.

R41000 (Salaries) =



The results in an Operational Planning data-entry form are as follows:

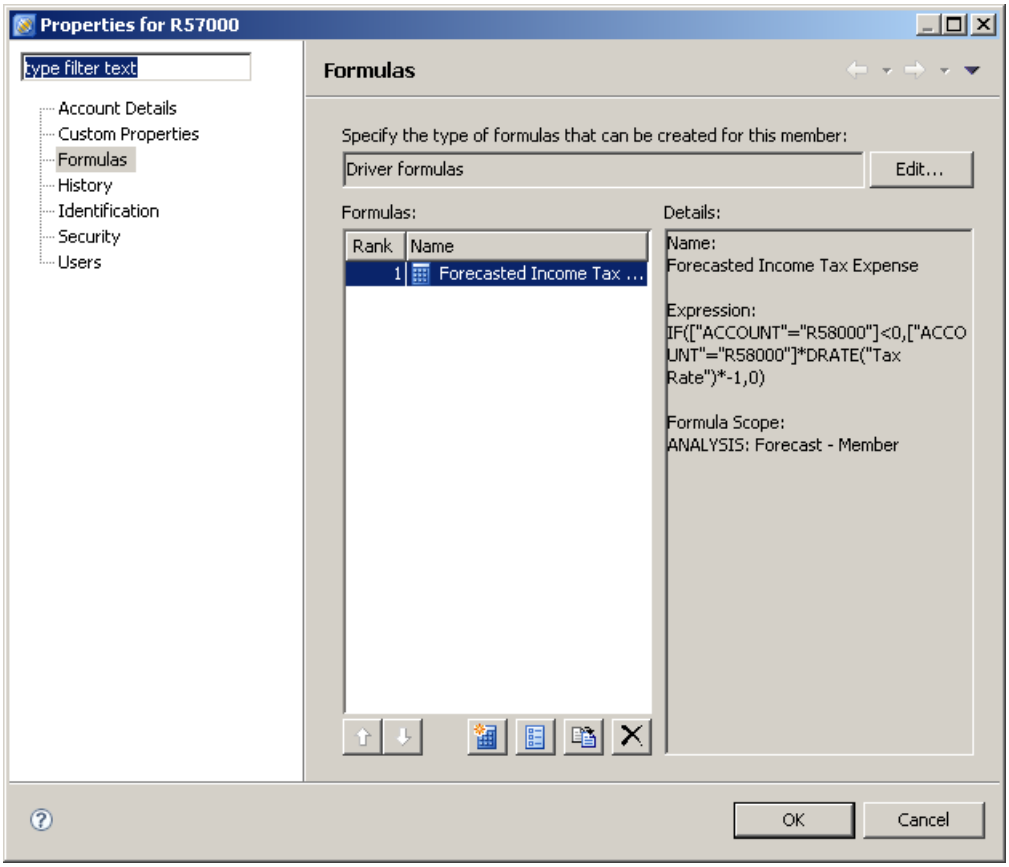
ORGANIZATION	Eastern U.S. Operations
Product	Product Total (vc)
Customer	Customer Total (vc)
Analysis	Forecast
Currency	USD
FREQUENCY	PTD

	Jan 2009	Feb 2009	Mar 2009	Q1 2009	Q2 2009	Jul 2009	Aug 2009	Sep 2009	Q3 2009
Salaries	10,000.00	10,000.00	12,500.00	32,500.00	0.00	11,000.00	11,000.00	13,750.00	35,750.00

Example 2: Forecasted Income Tax Expense

The intent of this formula is to calculate Income Tax expense only when Income before Taxes results in a profit. In the event of a loss, forecasted income tax expense returns a zero. When Income before Taxes is a profit, the formula expression calculates the applicable tax rate based on the IntOrg member per the Rate table in FM Studio, multiplies that by the Income before Taxes value to project forecasted income tax expense.

R57000 (Income Taxes) =



The results in an Operational Planning data-entry form are as follows:

Product	Simulation
Analysis	Forecast
Currency	USD
Customer	Customer Total (vc)
FREQUENCY	PTD
Time	Jul 2009

	Net Income	Income Taxes	Income before Taxes
Worldwide Operations	1,240,470.00	486,030.00	1,726,500.00
United States	986,400.00	383,600.00	1,370,000.00
Canada	163,200.00	76,800.00	240,000.00
Mexico	90,870.00	25,630.00	116,500.00

Using Excel-Based Calculated Members in Operational Planning

Introduction

The following examples illustrate common Excel-based calculated members that can be used in Operational Planning.

Example 1: Variance

This formula compares Actual to Budgeted values and returns the variance.

=fmValue("Budget")-fmValue("Actual")

After entering inputs and selecting either **`Refresh`** or **`Submit`**, the results in an Operational Planning data-entry form are as follows:

Product	Product Total (vc)
Account	Net Sales
Currency	USD
Customer	Customer Total (vc)
FREQUENCY	PTD
Time	Jul 2009

	Actual	Budget	Variance
Worldwide Operations	3,764,546.00	3,675,000.00	89,546.00
United States	2,905,696.00	2,800,000.00	105,696.00
Canada	648,964.00	675,000.00	(26,036.00)
Mexico	209,886.00	200,000.00	9,886.00

Example 2: Estimated Tax Rate

This formula returns the various tax rates as defined in the Tax Rate table for Driver Rates.

= fmRate("Tax Rate")

After entering inputs and selecting either **`Refresh`** or **`Submit`**, the results in an Operational Planning data-entry form are as follows:

Product	Simulation
Analysis	Forecast
Currency	USD
Customer	Customer Total (vc)
FREQUENCY	PTD
Time	Jul 2009

	Net Income	Income Taxes	Est Tax Rate	Income before Taxes
Worldwide Operations	1,240,470.00	486,030.00	0.28	1,726,500.00
United States	986,400.00	383,600.00	0.28	1,370,000.00
Canada	163,200.00	76,800.00	0.32	240,000.00
Mexico	90,870.00	25,630.00	0.22	116,500.00

Unsupported Formula Types and Formula Expressions

Introduction

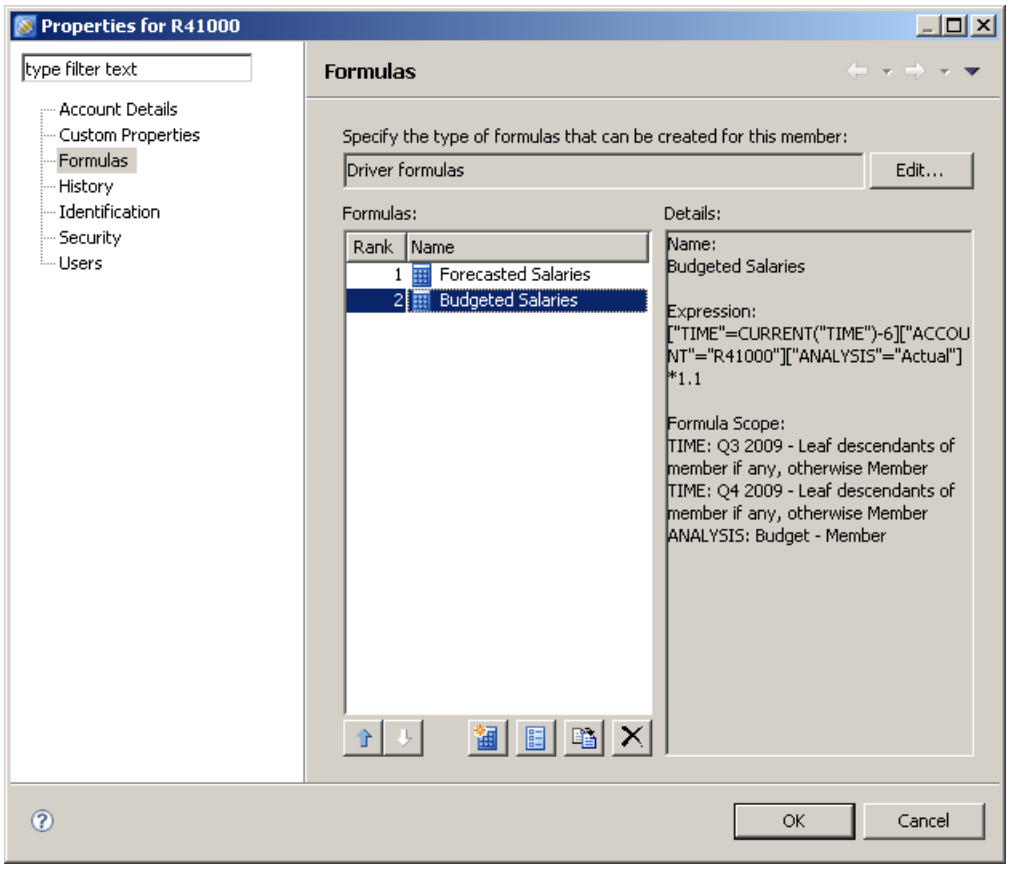
Operational Planning does not support Modeling or Reporting formula types. In addition to these two formula types, driver formulas that reference dimension members other than Time and/or Account are not supported.

The crossings related to unsupported formula types and expressions are rendered in Operational Planning as gray, non-writeable cells with a value of zero. To illustrate, consider the following examples.

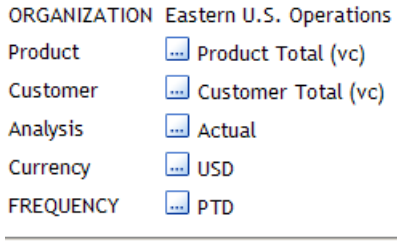
Example 1: Budgeted Salaries

The following is an example of an unsupported driver formula expression. This formula is similar to the previous example provided above. However, it includes a reference to an Analysis dimension member in the expression text.

R41000 (Salaries) =

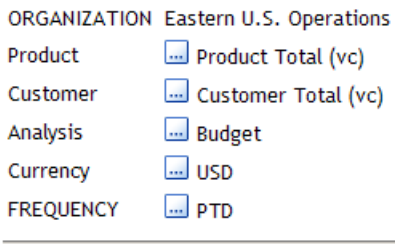


Assume the following actual data values for Q1'09:



	Jan 2009	Feb 2009	Mar 2009	Q1 2009	Q2 2009	Jul 2009	Aug 2009	Sep 2009	Q3 2009
Salaries	9,487.22	9,487.22	11,859.03	30,833.47	0.00	0.00	0.00	0.00	0.00

Slicing to the Budget Analysis member displays the Time members for Q3 2009 with gray, non-writeable cells and zero values. This is due to the fact that the Driver formula expression contains references to a dimension that is not supported for Operational Planning, and is therefore ignored.

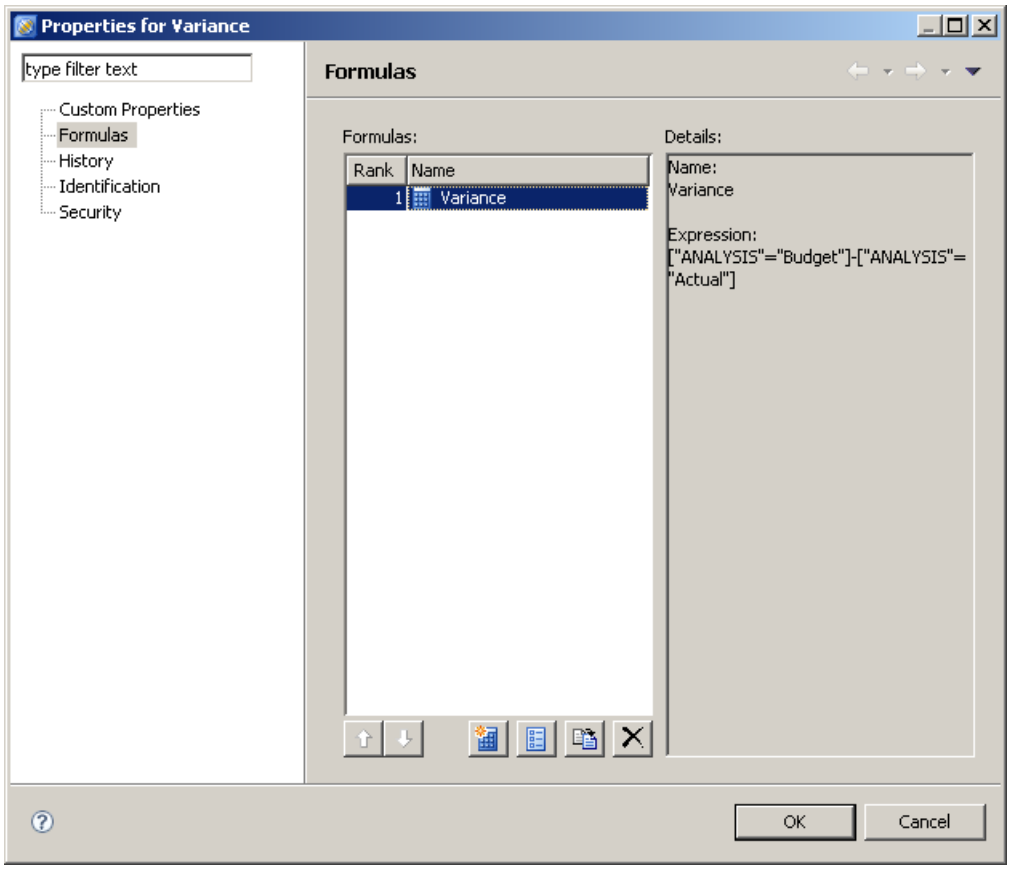


	Q1 2009	Q2 2009	Jul 2009	Aug 2009	Sep 2009	Q3 2009
Salaries	0.00	0.00	0.00	0.00	0.00	0.00

Example 2: Variance

This is an example of an unsupported formula type. The Variance Analysis member is defined as a Reporting formula and is therefore not supported in Operational Planning.

Variance =



After entering inputs and selecting either **`Refresh`** or **`Submit`**, note that the results for the Variance Analysis member are displayed as gray, non-writeable cells. Each gray cell has a value of zero in an Operational Planning data-entry form.

Product	Product Total (vc)
Account	Net Sales
Currency	USD
Customer	Customer Total (vc)
FREQUENCY	PTD
Time	Jul 2009

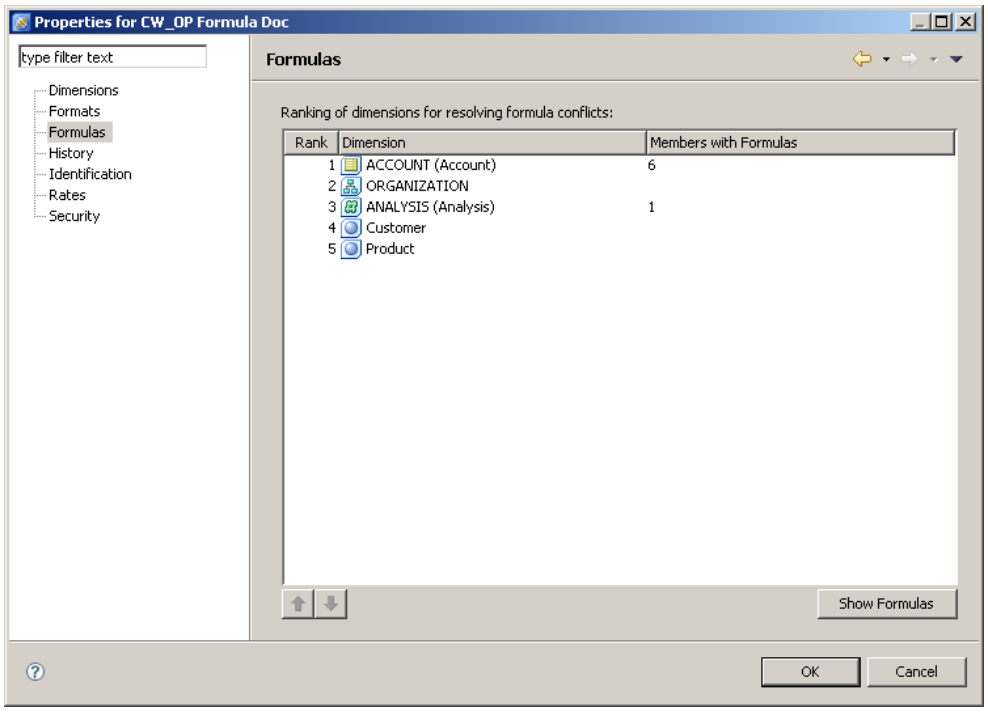
	Actual	Budget	Variance
Worldwide Operations	3,764,546.00	3,675,000.00	0.00
United States	2,905,696.00	2,800,000.00	0.00
Canada	648,964.00	675,000.00	0.00
Mexico	209,886.00	200,000.00	0.00

Reviewing Formula Warnings for Operational Planning

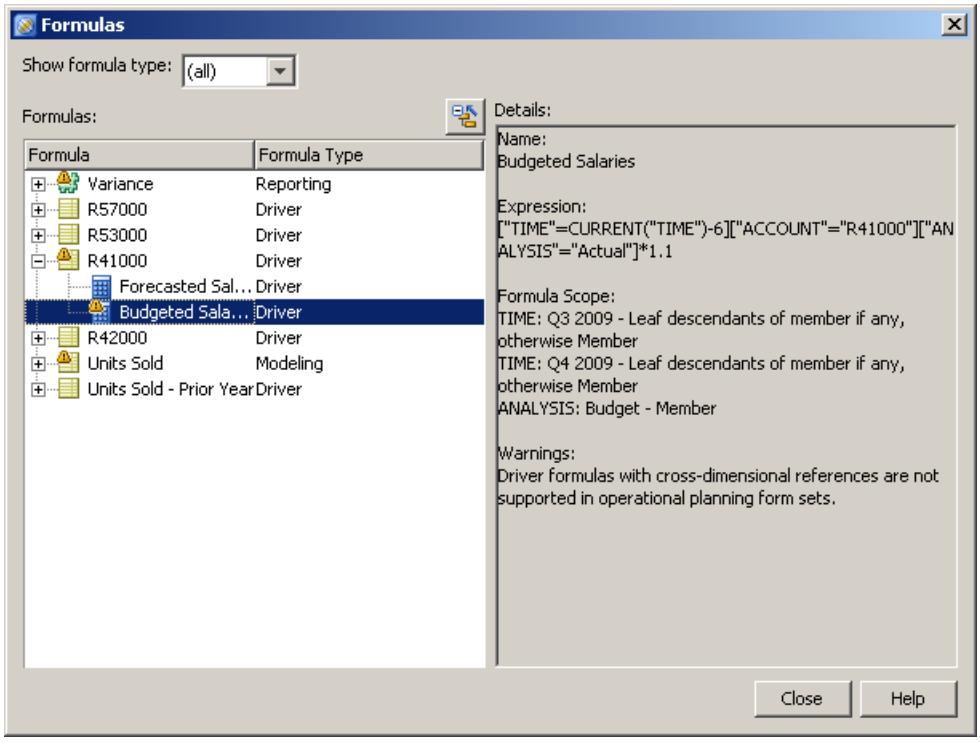
There are two locations where administrators can review and check for any formula warnings:

- Formulas page on the Model Properties dialog box
- Details dialog box offered at the time an Operational Planning form set is published

1 To check for formula warnings in the Models workspace, open the Models Properties dialog box and choose the Formulas page.

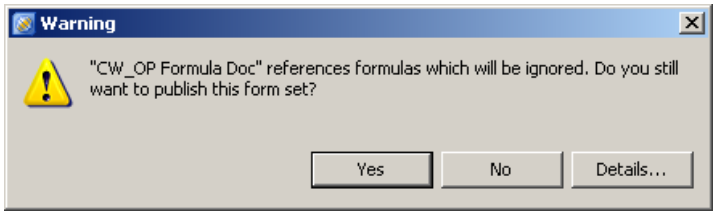


2 Click **Show Formulas**.

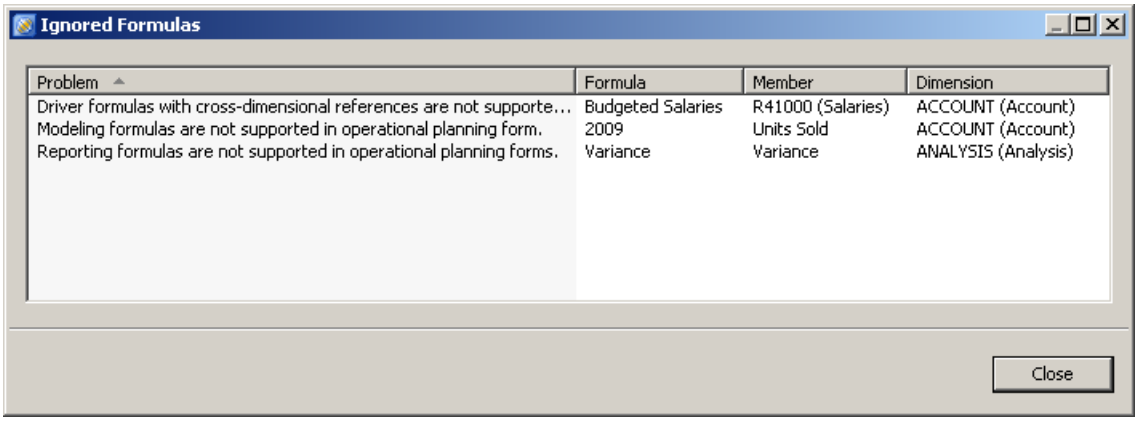


All calculated members for the selected hierarchies and as-of dates are displayed and can be sorted based on type. Each calculated member can be expanded to display the expressions associated with each member. Any formula expressions and calculated members in warning are marked and information is provided for each marked expression and member.

All information provided in the Model Property dialog box is also available at the time an Operational Planning form set is published. To illustrate, a form set associated with the Model noted above renders the following information at Publish:



Clicking on **`Details`** provides the following:



Selecting **No** on the Warning dialog will cancel the Publish. Selecting **Yes** will continue the Publish wizard to the next window.

Triggering Execution of a Driver Formula for Operational Planning

Introduction

There are two actions that trigger the execution of driver formulas in Operational Planning:

- entering data into a form
- publishing an Operational Planning form set

Driver Formulas That Are Triggered by Data Entry on a Form

The following provides the basic criteria for executing a driver formula at writeback for Operational Planning:

- The driver formula is not ignored. That is, it does not reference any dimension members other than Account or Time.
- The driver formula is in the hierarchy referenced in Model Properties.
- The input value shares the same crossings as the driver formula. The one exception is the Time dimension.

Driver Formulas That Are Triggered at Publish

The following provides the basic criteria for executing a driver formula at **`Publish`** for Operational Planning:

- The driver formula is not ignored. That is, it does not reference any dimension members other than Account or Time.
- The driver formula is in the hierarchy referenced in Model Properties.

Based on their design and limited range of execution, driver formulas provide an efficient way to calculate values in a SAS Financial Management query.

Driver Formulas and Changes to Exchange Rates, Rates, and Writeable Analysis Members

Since Models associated with Operational Planning form sets are locked upon Publish, driver formula execution is insulated against changes to driver formulas. Such changes include the addition or deletion of driver formulas and changes to the formula expression. There are, however, a few scenarios that require the Process Administrator to reset and republish in order to execute driver formulas for all crossings:

- changes to exchange rates that impact the computation for driver formulas
- changes to rates that impact the computation for driver formulas
- additional Analysis members selected for the writeable analysis members

If any of these scenarios include driver formulas that have been impacted, the form set should be reset and republished to ensure accurate driver formula results.

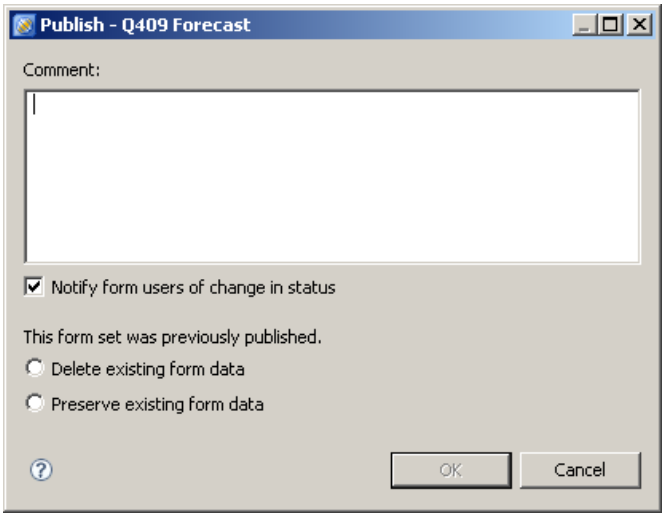
Deleting Facts and Driver Formula Results for Operational Planning

Introduction

Based on the form set state, there are three options for deleting Operational Planning facts and driver facts:

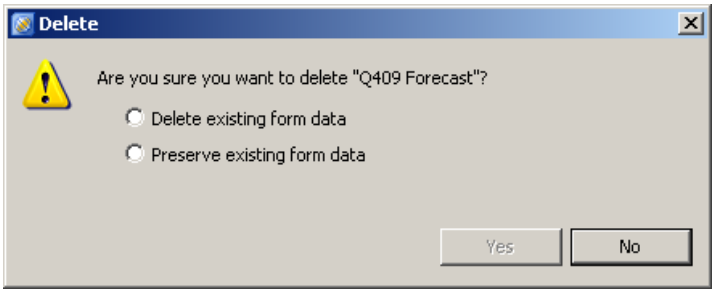
Facts Are Associated with an Incomplete Form Set

Republishing the form set offers an option to delete or preserve facts before publishing. This includes driver facts associated with the form set. This delete applies only to data entered in the form and driver formula results. It does not apply to data loaded via SAS Data Integration Studio.



Facts Are Associated with a Complete Form Set: Form Set Delete

Deleting a completed form set offers the option to delete or preserve facts before deleting. This includes driver facts associated with the complete form set. This delete applies only to data entered in the form as well as driver formula results. It does not apply to data loaded via SAS Data Integration Studio.



Facts Are Associated with a Complete Form Set: Delete Data from This Cycle Option

Deleting data through the **`Delete data from this cycle`** action in the **`Periods`** workspace will delete both facts and driver formula facts based on the specified crossings. This delete applies to data entered in a completed form set, driver formula results, and data loaded via SAS Data Integration Studio.

CHAPTER 14

Troubleshooting

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 Excel Add-in
- Web-based data-entry forms

Troubleshooting in SAS Financial Management Studio

Financial and Operational Planning Cycles: Reviewing Formula Validation within a Model

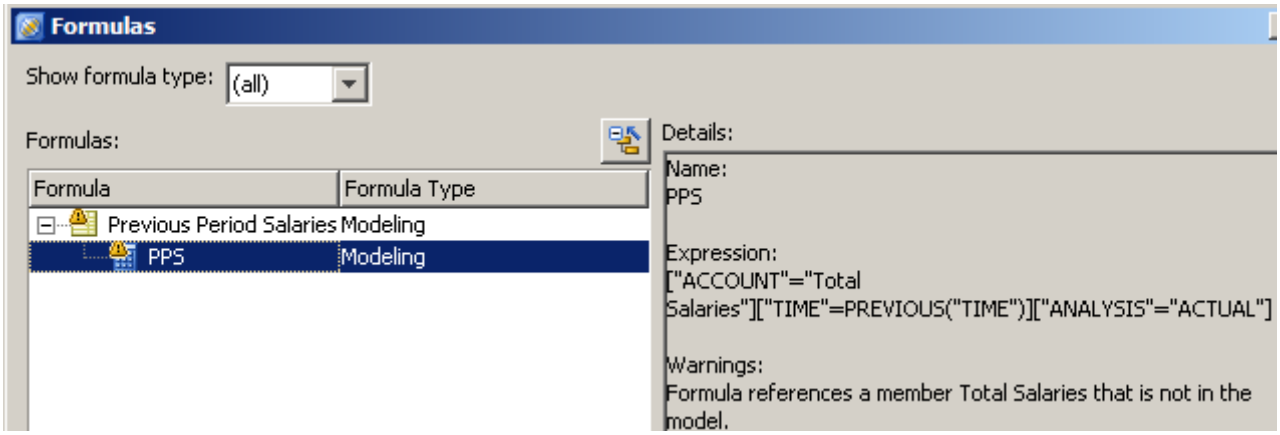
Formula syntax validation is performed when a formula expression is created on a calculated member in the Dimensions workspace in SAS Financial Management Studio. The validation ensures that the dimensions and members referenced in the formula expression exist and that the expression is syntactically 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:

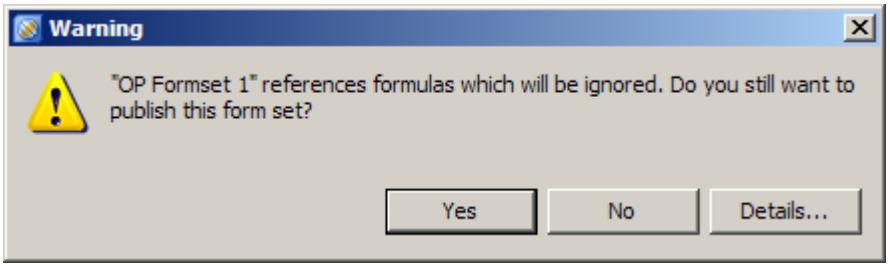


To minimize the number of invalid formula expressions displayed in a table, it is recommended that you resolve all Model formula warnings before viewing formula results in Excel or on the Web.

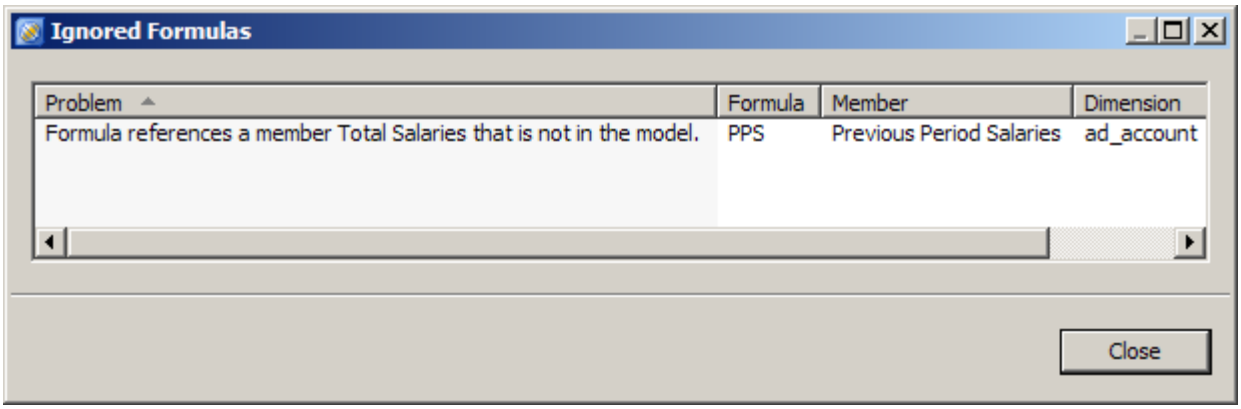
Operational Planning Cycles: Reviewing Formula Validation at Form Set Publish

The Publish action (with only Operational Planning form sets) generates a warning message if the Operational Planning model hierarchies include any of the following calculated members:

- calculated members assigned as Reporting formula types
- calculated members assigned as Modeling formula types
- calculated members assigned as Driver formulas that reference dimension members in the formula expression other than Account and Time members
- any invalid calculated members identified in Chapter 7, "Calculated Member Restrictions"



Selecting **Details** provides the list of Ignored Calculated members and gives specific information as to why each member is ignored.



Note that the list of Ignored formulas is a comprehensive list of all Ignored formulas based on the selected dimension hierarchies in the model. The result is comprehensive whether the calculated member is actually included in the form template or not. This listing is the same information that is available in Model Properties on the **`Formulas`** tab.

If the user continues with the **`Publish this form set`** action by selecting **Yes**, any crossings associated with these Ignored calculated members included in the form set template are displayed on the Web as follows:

Ignored Calculated Members	Appearance
Calculated members assigned as Reporting formula types	Gray, non-writeable cells with values of zero
Calculated members assigned as Modeling formula types	Gray, non-writeable cells with values of zero
Calculated members assigned as Driver formulas that reference Dimension members in the formula expression other than Account and Time members	Gray, non-writeable cells with values of zero
Any invalid calculated members identified in Chapter 7, “Calculated Member Restrictions”	Gray, non-writeable cells with values of zero

For formula warnings detected at Publish, resolve any invalid Driver formula warnings prior to Publish to ensure that the formulas will execute as desired.

Troubleshooting in Excel and on the Web

In some cases, formula results in the context of an Excel or Web-based table 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 on the Web.

Financial Cycle: 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, Default 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 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"]

	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 (MP3 player versus Cell phone) resulting in a value of zero for Account C.

	TOTAL	MP3 player	Cell phone
Account A	3.00	3.00	0.00
Account B	4.00	0.00	4.00
Account C	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	ANALYS	IRR	IT	TIME	RA	SOURCE	C	OBJECT_TYP	OBJECT	Value	USD converted
Previous Period Salaries	BUDGET	USD	SA	FEB2000	EXT	BaseForm	A	dataload	6724	3.60	3.60
Previous Period Salaries	BUDGET	USD	SA	FEB2000	EXT	BaseForm	A	formset	Salary Expense	9.00	9.00
Previous Period Salaries	BUDGET	USD	SA	FEB2000	EXT	BaseForm	A	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.

Web Data-Entry Forms

Whether a Financial or Operational Planning cycle, **Cell Information** and **Contributing Data** actions are not available in a Web data-entry form. To view information for a NaN displayed on the Web, hover over the NaN. A floating dialog box appears with information regarding the NaN.

For invalid formula issues in a Web data-entry form for Financial Cycles, the Excel template can be used to verify formula syntax for a specific cell through Cell Information. Note that in an Operational Planning Excel template, queries are not performed and all values are displayed as zero. Therefore, no NaNs are displayed.

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

Your Turn

We welcome your feedback.

- If you have comments about this book, please send them to **`yourturn@sas.com`**. Include the full title and page numbers (if applicable).
- If you have comments about the software, please send them to **`suggest@sas.com`**.