SAS Publishing



SAS® Information Map Studio 3.1: Tips and Techniques

The Power to Know

The correct bibliographic citation for this manual is as follows: SAS Institute Inc. 2006. SAS® Information Map Studio 3.1: Tips and Techniques. Cary, NC: SAS Institute Inc.

SAS[®] Information Map Studio 3.1: Tips and Techniques

Copyright © 2006, SAS Institute Inc., Cary, NC, USA

ISBN-13: 978-1-59994-004-5 ISBN-10: 1-59994-004-3

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

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

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

U.S. Government Restricted Rights Notice. Use, duplication, or disclosure of the software by the government is subject to restrictions as set forth in FAR 52.227–19 Commercial Computer Software-Restricted Rights (June 1987).

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

1st printing, March 2006

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/pubs** or call 1-800-727-3228.

 SAS° and all other SAS Institute Inc. product or service names are registered trademarks or trademarks of SAS Institute Inc. in the USA and other countries. $^{\circ}$ indicates USA registration.

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

Contents

Chapter 1 \triangle About This Book 1

How to Use This Book 1 Intended Audience 1 Online Help 1

Chapter 2 riangle Tips for Setting Up Your Environment 3

Make User-Defined Formats Available 4 Make SAS Programs Available as SAS Stored Processes for Information Maps 6 Restrict Access to Specified Rows by Using a SAS Stored Process 8 Make Detail Data Available for Drill-Through 11 Make the Column Labels of Drill-Through Tables Available 19 Display Detail Data for a Large Cube 22 Increase the Java Heap Size for SAS Information Map Studio 25

Chapter 3 riangle Tips for Creating Information Maps 27

Create a Filter with a Measure, Category, or Column 27 Use a SAS Stored Process with an Information Map 40 Use SAS Stored Process Input Parameters with an Information Map 43 Use a Table Alias 44 Create Conditional SQL Code 45 Create Data Items from Hierarchies 47

Glossary 49

Index 55

CHAPTER1About This Book

How to Use This Book 1 Intended Audience 1 Online Help 1

How to Use This Book

This book documents tips and best practices for using SAS Information Map Studio 3.1. Each tip begins with a summary of the tip and then provides either background information about the tip or a detailed description of the technique for implementing the tip.

1

Intended Audience

This book assumes that the reader has a working understanding of SAS Information Map Studio and SAS Information Maps. It is also useful for the reader to have some knowledge of related SAS business intelligence products such as SAS Management Console and SAS Data Integration Studio.

Online Help

This book is a companion to the online Help for SAS Information Map Studio. The Help describes all of the windows and dialog boxes in SAS Information Map Studio, and it describes the main tasks that you can perform with the software. Use any of the following methods to get Help for SAS Information Map Studio:

Ose any of the following methods to get help for SAS information wap Stud

- \Box From the menu bar, select **Help** \blacktriangleright **SAS Information Map Studio**.
- □ Click **Help** (when available) in the application windows and dialog boxes.
- $\hfill\square$ Press F1 in most application windows and dialog boxes.

Tips for Setting Up Your Environment

Make User-Defined Formats Available 4 Tip 4 Technique 4 Use the LIBRARY Libref 4 Use the FMTSEARCH System Option 4 Make SAS Programs Available as SAS Stored Processes for Information Maps 6 Tip 6 Technique 6 Restrict Access to Specified Rows by Using a SAS Stored Process 8 Tip 8 Technique 8 Background 8 Example Scenario: Description 8 Example Scenario: The Basic Query 9 Example Scenario: The Subquery 9 Example Scenario: The Completed Query 9 Example Scenario: The Stored Process 10 Make Detail Data Available for Drill-Through 11 Tip 11 Technique 11 Make Detail Data Available to a Cube 11 Make Detail Data Available to an OLAP Server 13 Make Detail Data Available to an Information Map 17 Make the Column Labels of Drill-Through Tables Available 19 Tip 19 Technique 19 Display Detail Data for a Large Cube 22 Tip 22 Technique 22 Increase the Java Heap Size for SAS Information Map Studio 25 Tip 25 Technique 25

Make User-Defined Formats Available

Tip

You have two options for making user-defined formats available to SAS Information Map Studio. One option is to place a specifically named format catalog in the directory that is assigned to the LIBRARY libref. The other option is to specify the format catalog in the FMTSEARCH system option in your SAS configuration file.

Technique

Use the LIBRARY Libref

Use the LIBRARY libref to make your user-defined formats available by completing the following steps:

- 1 Name the format catalog formats.sas7bcat.
- 2 Place the catalog in the directory that is assigned to the LIBRARY libref. By default, the SAS Configuration Wizard assigns the LIBRARY libref to the SAS-config-dir\Lev1\SASMain\SASEnvironment\SASFormats directory, where SAS-config-dir is the SAS configuration directory.

Note: For the z/OS* environment, in the SASRX REXX exec, you must also add the following LIBRARY ALLOCATE command, which points to the *SAS-config-dir/* Lev1/SASMain/SASEnvironment/SASFormats directory:

```
allocate dd(library)
path('SAS-config-dir/Lev1/SASMain/SASEnvironment/SASFormats')
△
```

Use the FMTSEARCH System Option

Use the FMTSEARCH system option to make your user-defined formats available by completing one of the following sets of steps:

- □ For all but the z/OS environment, complete the following steps:
 - 1 Add a -CONFIG system option to the SAS configuration file SAS-config-dir\Lev1\SASMain\sasv9.cfg that points to a configuration file for handling user-defined format catalogs. For example, you might add the following -CONFIG system option:

-config "SAS-config-dir\Lev1\SASMain\userfmt.cfg"

* z/OS is the successor to the OS/390 operating system. SAS 9.1 for z/OS is supported on both z/OS and OS/390 operating systems and, throughout this document, any reference to z/OS also applies to OS/390, unless otherwise stated. 2 Then, in that user-defined format configuration file (for example, userfmt.cfg), add a SET system option and a FMTSEARCH system option. For example, you might add the following system options:

-set fmtlib1 "SAS-config-dir\Lev1\Data\orformat"
-fmtsearch (fmtlib1.orionfmt)

In this example, SAS-config-dir\Lev1\Data\orformat is the location of the format catalog, and orionfmt (filename orionfmt.sas7bcat) is the name of the format catalog.Note: For UNIX environments, the variable name that you specify in the SET option must be converted to uppercase. For example, in the following option, you would specify FMTLIB1 instead of fmtlib1:

```
-set FMTLIB1 "SAS-config-dir\Lev1\Data\orformat"
```

Δ

 \Box For the z/OS environment, complete the following steps:

1 Add an AUTOEXEC system option to the SAS launch command as shown in the following example:

```
SAS-config-dir/Lev1/SASMain/startsas.sh
    o("autoexec="./WorkspaceServer/userfmt.sas"")
```

In this example, **startsas.sh** is your SAS launch command script, and **userfmt.sas** is the name of the SAS autoexec file. When you enter the command, you must enter it all on one line.

2 Within the autoexec file, use the LIBNAME statement to assign the format library and the OPTIONS statement to set the FMTSEARCH system option. For example, you might specify the following statements:

libname fmtlib1 'SAS-config-dir/Lev1/Data/orformat';
options fmtsearch=(fmtlib1.orionfmt);

Make SAS Programs Available as SAS Stored Processes for Information Maps

Tip

To use a SAS program to update or create data for an information map, convert the SAS program to a SAS Stored Process and make it available for use with the information map.

Technique

To convert a SAS program to a stored process that is available for use with an information map, complete the following steps:

1 A table and its library must be defined in metadata before the table can be used as a data source for an information map. (For this reason, you cannot begin using the Work library for your output until after you first create a permanent location for your output.)

To define the program's output table (or tables) and its associated SAS library in the metadata repository, complete the following steps:

a If the output table does not physically exist, then run the program one time to create the table and to store it in a permanent library. The table that is created must contain header information about the columns and their attributes; rows are not required. Here is a sample program:

```
Libname source 'path-to-source-data';
Libname stpout 'c:\temp';
Data stpout.result_set;
  Set source.datal;
  /* more code */
run;
```

Note: This sample program contains LIBNAME statements for both a source library and an output library. The source library is the permanent (original) location of the data. This sample program reads data from the source table, modifies or updates it, and outputs the results to the output table. \triangle

b Use either Data Library Manager (in SAS Management Console), SAS Data Integration Studio, or the SAS Metadata LIBNAME Engine to define the output table and its library to the metadata repository. When you specify the libref for the library definition, use the one that you specified in the program. For the above sample program, you would specify a libref of **STPOUT**.

- 2 In the SAS program, insert a *ProcessBody statement immediately after your global (%GLOBAL) macro variable declarations. This comment statement is required when the SAS program references macro variables (global or reserved) and the stored process runs on a SAS Workspace Server.
- 3 Modify the LIBNAME statement for your output table so that it references the Work library of the workspace server. The stored process will now write its output to this temporary location.

Note: This step is optional. You can continue to use the permanent table and library that you created in step 1. However, using the Work library prevents the original data from being over-written and enables multiple, concurrent users of a stored process to each have access to a temporary, private version of the resulting data. \triangle

The following example shows the modified sample program:

- 4 Save the program.
- 5 Define the program in the metadata repository as a stored process (using either BI Manager in SAS Management Console or SAS Enterprise Guide). Specify that execution will take place on a workspace server and then specify an output type of NONE. If you use global macro variables in your SAS program and want to use them to prompt users for values, then add a stored process input parameter for each global macro variable that your code references. (For information about using stored process input parameters with an information map, see "Use SAS Stored Process Input Parameters with an Information Map" on page 43.)

Note: The stored process must run on a *workspace server*. A stored process that runs on a stored process server cannot be used by an information map. If you use input parameters with the stored process, then be aware that the workspace server supports single selection parameters only. Δ

6 Using BI Manager, navigate to the stored process and verify that your SAS Information Map Studio users have ReadMetadata access to the stored process.

The stored process can now be used with an information map. For details, see "Use a SAS Stored Process with an Information Map" on page 40.

For more information about stored processes, see "SAS Stored Processes" in the SAS 9.1.3 Integration Technologies: Developer's Guide at http://support.sas.com/rnd/itech/doc9/dev_guide/stprocess/index.html.

Restrict Access to Specified Rows by Using a SAS Stored Process

Tip

You can use a SAS Stored Process to restrict an information map user's access to certain rows in a table. A stored process that is associated with an information map is executed before any of the queries that are generated from that information map. Therefore, the stored process can be used to perform any type of data subsetting that must occur before a query is run.

Technique

Background

To restrict access to certain rows of data in your table, you can create a stored process that filters out those rows so that the information map user queries against only the remaining subset of data. To determine whom to restrict access to, reference the _METAPERSON and _METAUSER reserved macro variables in the stored process's source code (SAS program). These macro variables are system variables that automatically contain the user's metadata user name (_METAPERSON) and login (_METAUSER).

The following example scenario describes how to create a stored process that uses the _METAPERSON reserved macro variable to restrict row-level access.

Note: In the example, the report user is also the stored process user because the report user runs the stored process every time he or she runs the report. \triangle

Example Scenario: Description

Your sales organization has a SAS library named Orstar. This library contains an orders table (Order_Fact) that lists orders by employee ID and an employees table (Organization_Dim) that contains a list of salespeople.

You have an information map that enables the salespeople to view the orders in the orders table. You need to ensure that when a salesperson (report user) uses your information map to generate an order report, that person will see only his or her own orders. To enforce this policy, you need to create a stored process that you can associate with the information map.

Example Scenario: The Basic Query

The stored process that you create needs to contain a query that filters out employee IDs that do not belong to the report user. The query has the following basic structure:

```
Employee_ID IN (
SELECT org.Employee_ID
FROM
    orstar.Organization_Dim AS org,
    <subquery:
    generates-lookup-table-containing-report-user's-employee_id>;
    WHERE org.Employee_ID = lookup.Employee_ID
)
```

Example Scenario: The Subquery

You know that the names in the Employee_Name column of the Organization_Dim table are used as the employees' metadata user names (_METAPERSON). By using the following SELECT statement, you can determine the employee ID of the employee who is running the report.

```
SELECT Employee_ID FROM orstar.Organization_Dim AS o2
WHERE (o2.Organization_Dim = "&_metaperson") AS lookup
```

Example Scenario: The Completed Query

Incorporate the subquery into the basic structure that you created, and you get the following query:

```
Employee_ID IN (
SELECT org.Employee_ID
FROM
            orstar.Organization_Dim AS org,
            (SELECT Employee_ID FROM orstar.Organization_Dim AS o2
                WHERE (o2.Employee_Name = "&_metaperson")) AS lookup
WHERE org.Employee_ID = lookup.Employee_ID
)
```

Example Scenario: The Stored Process

The following example shows how the query that you created is incorporated into a SAS program that controls access to the Order_Fact table. To convert this SAS program into a stored process and use it with an information map, see "Make SAS Programs Available as SAS Stored Processes for Information Maps" on page 6.

```
*ProcessBody;
libname orstar 'c:\Orstar';
libname ortemp (work);
proc sql;
  CREATE TABLE ortemp.Orders_rls AS
  SELECT * FROM orstar.Order Fact
       WHERE
           Employee ID IN (
           SELECT org.Employee_ID
              FROM
                 orstar.Organization Dim AS org,
                 (SELECT Employee_ID
                    FROM orstar.Organization Dim
                       AS o2
                    WHERE (o2.Employee_Name = "&_metaperson"))
                       AS lookup
             WHERE org.Employee_ID = lookup.Employee_ID
           );
quit;
```

Make Detail Data Available for Drill-Through

Tip

You can drill through an OLAP report to the underlying detail data only after you make the detail data available to the cube, its SAS OLAP Server, and the information map that you use for creating the report.

Technique

Make Detail Data Available to a Cube

You can use either SAS OLAP Cube Studio or the OLAP procedure to make detail data available to the cube.

- □ In SAS OLAP Cube Studio, you can specify a table for drill-through when you create or edit the cube using the Cube Designer wizard. On the Drill-Through page of the wizard, select one the following radio buttons:
 - □ **Use input table for Drill-Through** (to use the table that you selected on the Input page of the wizard).

Select or define a registered data source to be used for 1 the viewing of detail data at reporting time. Once you hav may view information about that data source or continue the data source is not registered press Define Table to g	e selected a data source by pressing the Next bu
No table for Drill-Through <u>Use input table for Drill-Through: CLASS</u> Select table for Drill-Through from list:	
Select a Detail Table:	Table Proventies
Foundation	Calumi Properties
	Define Table
	Table Options

□ Select table for Drill-Through from list (to select a table from the list of tables on the Drill-Through page. If only a repository folder is listed, then select the repository folder in order to see the tables).

 No table for Drill-Through Use input table for Drill-Through: CLASS Select table for Drill-Through from list: 	
Select a Detail Table:	Table Proverties
CLASS PRDSALE RETAIL	Calumi Properties
SHOES EMPINFO	Define Table
SALARY	Table Optione

For more information about the Cube Designer wizard, see the SAS OLAP Cube Studio Help.

□ In the PROC OLAP statement, use the DRILLTHROUGH_TABLE option to specify the name of the drill-through table to use. For more information about the DRILLTHROUGH_TABLE option, see "PROC OLAP Statement" in the SAS OLAP Server: User's Guide.

Make Detail Data Available to an OLAP Server

In order for the OLAP server to make detail data available for a cube, the SAS library for the table that contains the detail data must be defined so that the OLAP server can access it. The simplest way to define the library to the server is to pre-assign it in the metadata repository.

To specify a library as pre-assigned for an OLAP server, complete the following steps:

- 1 In Data Library Manager (in SAS Management Console), find the **SAS Libraries** folder and perform one of the following tasks to get to the dialog box that lets you select advanced options:
 - □ For a new library, right-click the **SAS Libraries** folder and select **New Library** to start the New Library Wizard. Then navigate to the wizard page that enables you to specify library options such as the libref.

5-1	Libref:	CLASS
	Engine:	BASE
X	Enable WebDA	√ Support
No.	Content Server:	Calcor One 1
1-1-1	Path Specification:	Available items: Selected items:
		SASEnvironment/SASCode\
		New Edit Delete
		Advanced Options.

□ For an existing library, open the **SAS Libraries** folder and right-click the desired library. Select **Properties** from the drop-down menu, and then select the **Options** tab in the properties dialog box.

sashelp Properties	× ×
General Options As	ssign Notes Extended Attributes Authorization
Libref:	sashelp
Engine:	BASE
Enable WebDA	/ Support
Content Server:	(Select One)
Path Specification:	Available items: Selected items:
	SASEnvironment/SASCode/Jobs
	c:templordsalecube
	New Edit Delete
	Advanced Options
	OK Cancel Help

2 Click Advanced Options.

3 Select the Library is pre-assigned check box on the Pre-Assign tab in the Advanced Options dialog box.

A	Advanced Options	×
	Pre-Assign Options for any host Host-specific options	
	✓ Library is pre-assigned	
	OK Cancel Help	

4 On the **Assign** tab of the properties dialog box or the server selection page of the New Library Wizard, ensure that the selected application server is the server container that contains your OLAP server.

sashelp Properties			×
General Ontions Assign	Notes Extended Attributes	Authorization	
SASMain			
SASWan			
			_
		ок	Cancel Help

- 5 Click **ok** in the properties dialog box, or finish entering information in the wizard.
- 6 Restart the OLAP server.

Note: If you want to enable users to view column labels when they drill through to detail data, then before you restart the server, complete the steps in "Make the Column Labels of Drill-Through Tables Available" on page 19. \triangle

The selected library is assigned after the selected OLAP server starts. After the OLAP server starts, ensure that the library is pre-assigned to the correct SAS OLAP server.

Make Detail Data Available to an Information Map

In order for an information map to produce a report that has drill-through capabilities, an option must first be set in the information map. You can set this option in one of two ways:

□ When you save a new information map, select the Allow (OLAP) drill-through to detail data check box in the Save As dialog box before you save the information map.

🖀 Save As	×
Save in: 🦳 Marcel 💌 🗢 🛛	⊇ 🗙 🖆 🏥 ▼ T <u>o</u> ols
Name: class	
Allow (OLAP) drill-through to detail data	
Description:	
	-
Save	Cancel <u>H</u> elp

For an existing information map, open the information map, right-click it, and then select Properties from its drop-down menu. Select the Allow (OLAP)
 drill-through to detail data check box on the Definition tab in the Information Map Properties dialog box.

😹 Information M	ap Properties	×
Information map na	ame: class	
Folder:	BIP Tree/ReportStudio/Maps/Marcel/	
Created:	Aug 19, 2005 11:40:19 AM EDT	
Author:	Marcel Dupree	
Modified:	Feb 6, 2006 11:14:34 AM EST	
Last modified b	y: Marcel Dupree	
De <u>s</u> cription:		
Allow (OLA	AP) drill-through to detail data	
		OK Cancel <u>H</u> elp

Make the Column Labels of Drill-Through Tables Available

Tip

If you want to view column labels when you drill through to detail data for a cube, then you must set an option in Server Manager (in SAS Management Console) to make the column labels available.

Note: This option is set for the physical OLAP server, so the setting applies to all the cubes that are assigned to that server. \triangle

Technique

Before you can make column labels available, you must first ensure that SAS 9.1.3 Service Pack 4 is installed and then upgrade the metadata for any existing metadata repositories. For more information, see "Upgrading Repository Metadata" in the Metadata Manager Help in SAS Management Console.

After the metadata is upgraded, complete the following steps to make the column labels of drill-through tables available:

1 In the navigation tree for Server Manager, find the node that represents your physical OLAP server.



2 Select the server, and then select File ▶ Properties from the menu bar.

3 In the properties dialog box, select the **Options** tab, and then click **Advanced Options**.

SASMain - OLAP Server	Properties			×
General Options Notes	Extended Attributes Authorization			
Major Version Number:	9			
Minor Version Number:	1			_
Software Version:	9.1			_
Vendor:	-			
	SAS Institute			
		Adva	nced Option	s
		OK	Cancel	Help
		ОК	cancel	пер

4 In the Advanced Options dialog box, select the **Server** tab, and then select the **Use** the drill-through table column labels check box.

Advanced Options				×
Query Thread Pool Performance	Debug Query Deb Performance Lo	ouq Server 丨 ogging	Debuq Cache Cache	Journals Server
Buffer size for rowse	t access object:	1000		
Buffer size for cellset	access object:	1000		
Maximum number of re	egion execution threads:	8		
Maximum number of fl	attened rows:	300000		
Maximum memory size	e for flattened rowset:	268435456		
🔽 Use the drill-throu	igh table column labels			
		ОК	Cancel	Help

- 5 Click ox to save the setting.
- 6 Restart the OLAP server.

Display Detail Data for a Large Cube

Tip

If your cube contains an extremely large amount of detail data, then in order to view that data from within SAS Information Map Studio, you might need to increase the Java heap size for SAS Information Map Studio or increase the maximum number of drill-through rows that your SAS OLAP Server can handle.

Technique

For information about increasing the heap size, see "Increase the Java Heap Size for SAS Information Map Studio" on page 25.

To increase the number of drill-through rows that your OLAP server can handle, you can change an OLAP server definition setting in Server Manager (in SAS Management Console) by completing the following steps. (The default number of drill-through rows that can be displayed by a query is 300,000 rows.)

1 In the navigation tree for Server Manager, find the node that represents your physical OLAP server.



2 Select the server, and then select File ▶ Properties from the menu bar.

3 In the properties dialog box, select the **Options** tab, and then click **Advanced Options**.

SASMain - OLAP Server	Properties X
General Options Notes	Extended Attributes Authorization
Major Version Number:	9
Minor Version Number:	1
Software Version:	9.1
Vendor:	SAS Institute
	Advanced Options
	OK Cancel Help

24 Technique \triangle Chapter 2

4 In the Advanced Options dialog box, select the **Server** tab, and then enter the desired value for the **Maximum number of flattened rows** field.

Advanced Options				×
Query Thread Pool Performance	Debug Query Det Performance Lo	buq Server	Debuq Cache Cache	l .lournals Server
Buffer size for rowse	t access object:	1000		
Buffer size for cellset	access object:	1000		
Maximum number of re	egion execution threads:	8		
Maximum number of fl	attened rows:	300000		
Maximum memory size	e for flattened rowset:	268435456		
🔽 Use the drill-throu	gh table column labels			
		ОК	Cancel	Help

5 Click ox to save the setting.

Increase the Java Heap Size for SAS Information Map Studio

Tip

If your data source contains an extremely large amount of data, then you might need to increase the Java heap size for SAS Information Map Studio.

Technique

The default maximum heap size for SAS Information Map Studio 3.1 is 1024MB. To increase the heap size for SAS Information Map Studio, you can change the settings for its minimum heap size $(-\mathbf{Xms})$ and maximum heap size $(-\mathbf{Xmx})$ by completing the following steps:

- 1 In the installation directory for SAS Information Map Studio, locate the mapstudio.ini file.
- 2 Open the file using a text editor.
- 3 Locate the line that begins with CommandLineArgs.
- 4 Locate the following parameters and change the values:

-Xmsnm|g -Xmxnm|g

In these parameters, n is the number of megabytes or gigabytes, m represents megabytes, and g represents gigabytes.

5 Save the file before starting SAS Information Map Studio.

Tips for Creating Information Maps

Create a Filter with a Measure, Category, or Column 27 Tip 27 Technique 28 Background 28 Example Scenario: Filter Data After Aggregating 29 Example Scenario: Filter Data Before Aggregating 34 Example Scenario: Hide a Filter from SAS Web Report Studio 38 Use a SAS Stored Process with an Information Map 40 Tip 40 Technique 40 Use SAS Stored Process Input Parameters with an Information Map 43 Tip 43 Technique 43 Use a Table Alias 44 *Tip* 44 Technique 44 Background 44 Example Scenario: Recursive Join 44 Example Scenario: Alternate Joins between a Pair of Tables 44 Create Conditional SQL Code 45 Tip 45 Technique 45 Create Data Items from Hierarchies 47 *Tip* **47** Technique 47

Create a Filter with a Measure, Category, or Column

Tip

When you create a filter in SAS Information Map Studio, you can specify category data items, measure data items, and table columns in the filter expression. Use the following criteria to determine which of those items to use in the filter expression:

□ To filter aggregated data, create a measure data item, and then use that data item in the filter expression.

Note: If you use a measure data item in a SAS Information Map Studio filter, then that filter will not be available in SAS Web Report Studio. In SAS Web Report

Studio, if you want to use a filter that references a measure data item, then you must create the filter within a report in SAS Web Report Studio. \triangle

- □ To filter individual data values, create a category data item from the column, and then use the category data item in the filter expression, or use a table column in the filter expression.
- □ To hide a filter from SAS Web Report Studio users, use only table columns in the filter expression because SAS Web Report Studio does not display filters that are based solely on table columns.

For step-by-step instructions for creating filters, see the SAS Information Map Studio Help.

Technique

Background

The contents of a filter expression determine the type of SQL statements that are generated for the filter.

- If you specify a measure data item (whose values are by definition aggregated) in a filter expression, then the SQL query that is generated for the filter will contain a HAVING clause, which means that the filter is evaluated after the data items in the query are aggregated.
- □ If you specify a category data item or table column (whose values are not aggregated) in a filter expression , then the SQL query that is generated will contain a WHERE clause, which means that the filter is evaluated before the data items in the query are aggregated. A HAVING clause is generated only if you apply an aggregate function to a category data item or table column in the filter expression. In that case, because the category data item or table column is aggregated, the SQL that is generated will contain a HAVING clause.

The following sections describe scenarios where filters are created with a measure data item, a category data item, and a table column. Two data sources are used:

- □ a transaction table (ORDER_FACT) that contains sales transactions for a company's products. The table includes the following columns:
 - \square Product_ID
 - □ Employee_ID
 - Order_Date
 - □ Total_Retail_Price
 - \Box CostPrice_Per_Unit
 - □ Quantity
- □ a products table (PRODUCT_DIM) that contains information about the company's products. The table includes the following columns:
 - \square Product_ID

 \Box Product_Name

ORDER_FACT is inner joined to PRODUCT_DIM on the Product_ID column. The Product Name and Order ID category data items have been created from the Product_Name column in PRODUCT_DIM and the Order_ID column in ORDER_FACT.

Example Scenario: Filter Data After Aggregating

You want to determine which of your company's products are profitable. Click **Edit** on the **Definition** tab in the Data Item Properties dialog box to open the Expression Editor dialog box. Use the Total_Retail_Price, CostPrice_Per_Unit, and Quantity columns to create a Gross Profit measure data item that calculates the gross profit from product sales.

🛃 Expressio	n Editor	X
Name:	Gross Profit	
Description:		Í
	J	1
<u>T</u> ype:	Numeric 🔽	1
Expression T		1
	FACT.Total_Retail_Price>> - (< <order_fact.costprice_per_unit>> * FACT.Quantity>>)</order_fact.costprice_per_unit>	
+ - *	/ ** AND OR NOT = <> < <= > >= =* '_' (_)	
Undo	Redo Validate Expression Add to Expression	
<u>F</u> unctions	Data <u>S</u> ources	
Data Elem	ents:	
Column Qu	al Data ORDER_FACT 	
Quantity O	rdered	
	OK Cancel <u>H</u> elp	

On the **Classifications/Formats** tab in the Data Item Properties dialog box, select **Measure** as the classification of the data item and **SUM(argument)** as the aggregate function.

🛃 Data Item Properties:	Classifications/Formats	×
Data item name: Gross Pro	fit	
Definition Classifications	Formats Permissions/Value Generation Custom	
Classifications and ag		
Classification:	Aggregations (select at least one): Default aggregation: SUM(argument) Default aggregation:	
C Category € <u>M</u> easure	Sum(argument)	
	Select All Clear All	
Formatting Format type: Default Format	Default format from data source	
Default Format		
\$ \$ASCI		
\$BINARY		
\$CHAR		
Valid width range:	0	
Valid decimal range:	0 mm	
	OK Cancel Help	

After you create the data item, create a filter that checks to see if a product's gross profit is greater than 0. Because you want to evaluate an aggregated value (the gross profit for all transactions for a product), use the Gross Profit measure data item in the filter expression.

😹 New Filte	r						X
							_
<u>F</u> ilter name:	Which products are profitable	?					
Descripti <u>o</u> n:							
Data <u>i</u> tem:	Gross Profit			•	Edit	Data Item	
Condition:	ls greater than			•			
_[Value(<u>s</u>): –							
Enter valu	e(s)			-			
	0(0)						
o							
							_
					Comb	inations ₹	
							-
. ∃ Filter exp	ression:						
			ок	Car		Help	1
					1001	Tich	

Select the Product Name data item, the Gross Profit data item, and the filter for a test query.

🖀 Test the Information Map					×
Name of information map: untitled ┌─Data selection for test					
<u>A</u> vailable items:	Selected items:				
Show all items	ttem ➡ Product Name	Column	Properties Edit Edit	+	
Gross Profit	Y Which produ	•			
Eind	1	<u>V</u> iew Query			
No description for Which products are profitable	?			<u>*</u>	
Note: You may select only 1 filter from the available items list					
Output options					
Display aggregated values (group by categories)	ry)				
C Display a <u>d</u> etailed list of all values (no grouping	ng)				
☑ Limit the query result to: 100	(number of rows)				
		<u>R</u> un Test	Close	Help	

The SQL query that is generated should look similar to the following display:

🛃 View Que	ny	×
options Loca LIBNAME org	ale=en_US; gold BASE "DribataSources(SAS)Orion/orgold" ;	ſ
Proc SQL; C SELECT	reate Table %DATA% as	
FROM	table1 Product_Name AS DIR_2 LABEL=Product Name', SUM((table0.Total_Retail_Price - (table0.CostPrice_Per_Unit * table0.Guantity))) AS DIR_3 LABEL='Gross Profit' FORMAT=DOLLAR6.0	
GROUP BY	orgold.ORDER_FACT table0 Inner join orgold PRODUCT_DIM table1 on table0 Product_ID = table1 Product_ID	
HAVING	DIR_3>0	
quit;		
	Glose	1

≩ R	esults	
	Product Name	Gross Profit
1	11m Liga Football Boots	\$2,306
2	11m Men's Team Football Boots	\$2,750
3	2-Layer Cotton	\$9,639
4	2-Layer Suit	\$7,627
5	2bwet 3 Cb Swim Suit	\$2,567
6	2bwet 3 Cb Swimming Trunks	\$1,669
7	2bwet 3 Solid Bikini	\$2,382
8	3top Men's Sandal	\$3,606
9	3top Women's Sandal	\$2,538
10	4men Men's Air Golden Shoes	\$4,270
11	4men Men's Training Shorts	\$1,587
12	A-team Polo Pique	\$3,350
13	A-team Polo Pique w/Short Sleeve	\$8,970
14	A-team Shorts w/Pockets and Embroidery	\$746
15	A-team T-Shirt Round Neck	\$182
10	la na	
	<u>V</u> iew Query	Close

And the result set would be similar to the following:

Example Scenario: Filter Data Before Aggregating

Now you want to evaluate the gross profit per sales transaction. So create a Gross Profit by Transaction measure data item that contains the same expression as the Gross Profit measure data item. But on the **Classifications/Formats** tab, select **Category** as the classification of the data item.

nition Classifications	s/Formats Permissions/Value Generation Custom
lassifications and ag	gregations
Classification:	Aggregations (select at least one): Default aggregation:
Category	AVG(argument)
C Measure	COUNT + NMISS
	COUNT + NMISS {Distinct}
	COUNT(argument)
	Select All Clear All
ormatting Eormat type: Default Format	Default format from data source
Format type: Default Format	
Format type:	
Eormat type: Default Format Default Format	
Default Format Default Format S SASCII SBINARY	
Eormat type: Default Format Default Format \$ \$ASCII	
Eormat type: Default Format S SASCII SBINARY SCHAR	Default format from data source
Eormat type: Default Format S SASCII SBINARY SCHAR	
Eormat type: Default Format S SASCII SBINARY SCHAR	
Eormat type: Default Format S SASCII SBINARY SCHAR	

Create the same filter as in the previous example, but use the category data item instead of the measure data item because you do not want to filter the aggregated values.

Eiter name: Which transactions are profitable?	
Descripti <u>o</u> n:	
Data item: Gross Profit by Transaction Edit Data item	
Condition: Is greater than	
Value(s):	
Enter value(s)	
0	
Combinations ₹	1
OK Cancel <u>H</u> elp	

Select the Product Name data item, the Order ID data item, the Gross Profit by Transaction data item, and the filter for a test query.

🛃 Test the Information Map					×			
Name of information map: untitled								
Data selection for test								
<u>A</u> vailable items:	Selected items:							
Show all items	Item	Role	Properties					
Product Name	💼 Product Name		Edit					
	Crder ID	Column	Edit					
Gross Profit	Gross Profit	Column	Edit					
	Y Which trans							
🛛 🗌 🖵 Which transactions are profita 🛛 🛠								
	·							
								
<u>F</u> ind		View Query						
			_					
Data item description:								
No description for Which transactions are profitable?								
Note: You may select only 1 filter from the available items list								
Output options								
Display aggregated values (group by categ)	ory)							
C Display a detailed list of all values (no group	pina')							
, <u>-</u>								
☑ Limit the query result to: 100 (number of rows)								
					-			
		<u>R</u> un Test	Close	Help				

The SQL that is generated should look similar to the following display:

두 View Que	ery
options Loc: LIBNAME on	ale=en_US; gold BASE "D:/DataSources\SAS\Orionlorgold";
Proc SQL; C SELECT DIS	Sreate Table %DATA% as STINCT Table1 Product_Name AS DIR_16 LABEL="Product Name", Table0.Order_DA SDIR_12 LABEL="Order ID" FORMAT=F12.0,
FROM	(table0.Total_Retail_Price - (table0.CostPrice_Per_Unit *table0.Guantity)) AS DIR_13 LABEL='Gross Profit by Transaction' FORMAT=DOLLAR6.0 orgold.ORDER_FACT table0 inner join orgold PRODUCT_DM table1 on table0 Product_D = table1 Product_D
WHERE GROUP BY	(table0.Total_Retail_Price - (table0.CostPrice_Per_Unit *table0.Quantity)) > 0
	1, 2, 3
quit;	
	Qose

💦 R	esults			×
	Product Name	Order ID	Gross Profit by Transaction	
1	11m Liga Football Boots	1230041747	\$39	-
2	11m Liga Football Boots	1230137028	\$20	
3	11m Liga Football Boots	1230195661	\$20	
4	11m Liga Football Boots	1230232968	\$20	-
5	11m Liga Football Boots	1230255184	\$20	
6	11m Liga Football Boots	1230262331	\$20	
7	11m Liga Football Boots	1230457172	\$20	
8	11m Liga Football Boots	1230462176	\$39	
9	11m Liga Football Boots	1230572685	\$59	
10	11m Liga Football Boots	1230596062	\$20	
11	11m Liga Football Boots	1230732318	\$59	
12	11m Liga Football Boots	1230768998	\$20	
13	11m Liga Football Boots	1231267931	\$39	
14	11m Liga Football Boots	1231660896	\$20	
15	11m Liga Football Boots	1231845872	\$39	
16	11m Liga Football Boots	1231885520	\$39	
	la construction de la constructi	1001000000	****	•
		<u>⊻</u> iew Q	uery <u>C</u> lose	

And the result set would be similar to the following:

Example Scenario: Hide a Filter from SAS Web Report Studio

You want to apply a prefilter that prevents anyone who uses the ORDER_FACT table from viewing transactions that occurred before the year 2000. And, you do not want your SAS Web Report Studio users to use this filter because it will already be applied as a prefilter.

To create a filter that screens order dates, click **Edit Data Item** in the New Filter dialog box to select the Order_Date table column as the value of the **Data item** field. Then in the **Value(s)** section, select the date that you want to filter against.

Note: The New Filter dialog box displays the table column's label ("Date Order was placed by Customer") instead of its name. \triangle

Ne	w Fil	ter						
Filte	r nam	e: 🔽	ear >-	- 200	1			
_	criptic		car 2-	- 2000	,			
Des	cripu	m.						
Data	a įtem:	<	<ordi< td=""><td>ER_F/</td><td>ACT.D</td><td>ate O</td><td>rder w</td><td>as placed by Customer>> Edit Data Item</td></ordi<>	ER_F/	ACT.D	ate O	rder w	as placed by Customer>> Edit Data Item
Con	dition:	Is	after	or eq	ual to			•
Val	lue(s)	:						
Er	nter va	alue(s)					_
Г								
ľ	/alue:	Θs			time.			
		Οþ	oday				7	J
E	Þ		Janu	iary	•	200	10 ÷	
	Sun	Mon	Tue	Wed		Fri	Sat	
	26	27	28	29	30	31	01	
	02	03	04	05	06	07	08	
	09	10	11	12	13	14	15	
	16	17	18	19	20	21	22	
	23	24	25	26	27	28	29	
	30	31	01	02	03	04	05	
	Cu	urrent	Date/	time				
-					_			
								<u>C</u> ombinations ∓
- F	ilter e	xpres	sion:					
_		-		Date	Orde	r wa:	s plac	ed by Customer>> after or equal to 01Jan2000
Mat	e: nor	antha	646 H	ill ha	adder	lwha	n nace	ssary to indicate default item grouping.
1400	o. par	enthe	aca w	m be	awwei	- whe	n nece	ssary w marcare deraar nem grooping.
								OK Cancel Help

Then assign the filter to prefilter the ORDER_FACT table.

Note: For step-by-step instructions for assigning prefilters, see the SAS Information Map Studio Help. \triangle

🖳 Information Map Properties	×
Information map name: Gross Profit Information Map Definition General Prefitters Required Tables Custom Use the arrow buttons to assign prefitters to tables for all users. Prefitters subset the data in their associated tables before any queries are run against the tables.	1
Available filters: Which products are profitable? Which transactions are profitable? Wear >= 2000 Mew Edit Delete	
Description:	
Expression:	
OK Cancel Help	

When SAS Web Report Studio users open this information map, this filter will not be displayed to them. And, any time that the ORDER_FACT table is used in a query, the data will be pre-screened to include orders only from year 2000 and after.

Use a SAS Stored Process with an Information Map

Tip

In order to use a SAS Stored Process with an information map, the information map must use the same data source as the stored process, and the stored process must be selected for the information map.

Technique

A stored process that is associated with an information map is executed before any of the queries that are generated from that information map. This processing order enables you to use SAS tools such as the DATA step or the macro language to process the data that will be used as input for the information map.

Preprocessing data often involves subsetting or updating the data on a per-user basis (for example, when a user is prompted to enter parameter values). For this reason, it is often helpful to use the Work library of the SAS Workspace Server to store temporary copies of data that has been preprocessed for a particular user.

After you create a stored process (using either SAS Data Integration Studio, SAS Enterprise Guide, or the steps in "Make SAS Programs Available as SAS Stored Processes for Information Maps" on page 6), complete the following steps to use the stored process with an information map:

1 Ensure that the stored process is defined to run on a *workspace server*; a stored process that runs on a stored process server cannot be used by an information map.

Note: If you use input parameters with the stored process, then be aware that the workspace server supports single selection parameters only. (For information about using input parameters, see "Use SAS Stored Process Input Parameters with an Information Map" on page 43.) \triangle

- 2 View your stored process's source code (SAS program). Ensure that the following statement is inserted immediately after your global (%GLOBAL) macro variable declarations:
 - *ProcessBody;

This comment statement is required when the SAS program references macro variables (global or reserved) and the stored process runs on a workspace server.

- 3 Ensure that each table that your stored process outputs to is defined in the metadata repository. A table and its library must be defined in metadata before the table can be used as a data source for an information map. Complete the following steps to define an output table and its library to the metadata repository:
 - **a** If an output table does not physically exist, then run the stored process one time to create the table and to store it in a permanent library. The table that is created must contain header information about the columns and their attributes; rows are not required.

Note: For this step, ensure that the source code outputs to a permanent library and not a temporary library. If you want the stored process to use a temporary library (such as Work), you can temporarily change the source code to reference a permanent library and then change back to a temporary library after you complete this table creation step. Δ

- **b** Use either Data Library Manager (in SAS Management Console), SAS Data Integration Studio, or the SAS Metadata LIBNAME Engine to define the output table and its library to the metadata repository. When you specify the libref for the library definition, use the one that you specified in the source code.
- 4 Ensure that the data source that you specify for the information map is the table that your stored process outputs to. For example, if your stored process output is written to the RESULT_SET table in the STPOUT library, then you must use that table as the data source for your information map.

5 From the SAS Information Map Studio menu bar, select Tools ► Stored
 Processes, and then select the desired stored process in the Stored Processes dialog box.

😹 Stored Proce	\$\$65	×
-Stored process		
C Do not use a	a stored process	
	ected stored process	
C orion_pari C orion_pari C orion_pari O shoes_wi	m_notRequired_default m_notRequired_noDefault m_Required_Default m_required_noDefault thPrompt_noDefault_notRequired_ws thPrompt_noDefault_required_ws	
Properties		
Folder:	BIP Tree/ReportStudio/Shared/Reports/StoredProcesses/sastmd/prompted	
	۲. Example 2 and	
Description:	uses SASHELP.SHOES dataset; prompts for region and product	
Keywords:		
More Prop	erties	
	OK Cancel <u>H</u> elp	

6 Click **ox** to associate the stored process with the information map and to exit the Stored Processes dialog box.

When a query that is generated from the information map is executed, the associated stored process will run before the query code is processed.

For more information about stored processes, see "SAS Stored Processes" in the SAS 9.1.3 Integration Technologies: Developer's Guide at http://support.sas.com/ rnd/itech/doc9/dev_guide/stprocess/index.html.

Use SAS Stored Process Input Parameters with an Information Map

Tip

If you use a SAS Stored Process with an information map, and you want the stored process to prompt your report user for information, then add an input parameter for each value that you want to prompt for.

Technique

If your stored process's source code (SAS program) references macro variables that you want to use for prompting a user for values, then complete the following steps:

- 1 Add an input parameter for a macro variable if you want to prompt a user for a value for that variable. Use either BI Manager (in SAS Management Console), SAS Data Integration Studio, or SAS Enterprise Guide to add input parameters to the stored process.
- 2 Associate the stored process with an information map. To associate the stored process with an information map, see "Use a SAS Stored Process with an Information Map" on page 40.

When a query that is generated from this information map is executed, the stored process will first prompt the user for parameter values before the query is run.

For more information about stored processes and stored process input parameters, see "SAS Stored Processes" in the SAS 9.1.3 Integration Technologies: Developer's Guide at http://support.sas.com/rnd/itech/doc9/dev_guide/stprocess/index.html.

Use a Table Alias

Tip

Use an alias to support a recursive join or an alternate join between a pair of tables.

Technique

Background

When you insert an alias, you insert an additional logical representation of a table that is already referenced by an information map. The following examples describe scenarios where you would want to insert multiple logical representations of a given table:

Example Scenario: Recursive Join

You insert an Employee table that includes the Employee_ID, Employee_Name, and Manager_ID columns. Then you insert an additional logical representation of the Employee table so that you can create a relationship between Manager_ID and Employee_ID to get the manager's name for each employee.

Example Scenario: Alternate Joins between a Pair of Tables

You insert an Order table that includes the Origin_Location and Destination_Location columns, and a Geography table that stores country values in its Location column. In order to compare origin countries with destination countries, you create a relationship between each location column in the Order table and its own respective logical representation of the Geography table. The equivalent SQL code would look like the following example:

```
FROM
Order INNER JOIN Geography
ON Order.Origin_Location = Geography.Location
INNER JOIN Geography as Geography_Alias
ON Order.Destination_Location = Geography_Alias.Location
```

Create Conditional SQL Code

Tip

The Expression Editor in SAS Information Map Studio enables you to create conditional SQL code for your relational data items.

Technique

You can use a CASE statement in a relational data item's expression to perform conditional processing on the data item's values. The steps in the following example describe how to create a relational data item that converts the abbreviations for country names to their full names:

- 1 In SAS Information Map Studio, open the relational information map that you want to work with.
- 2 From the menu bar, select Insert ► Data Item.
- 3 In the Type field, select Character.
- 4 On the **Definition** tab in the Data Item Properties dialog box, click **Edit** to open the Expression Editor dialog box.
- 5 In the Expression Text area, enter the following expression:

```
case <<GEOGRAPHY_DIM.Country Abbreviation>>
    when 'AU' then 'Australia'
    when 'BE' then 'Belgium'
    when 'DE' then 'Germany'
    when 'DK' then 'Denmark'
    when 'ES' then 'Spain'
    when 'FR' then 'France'
    when 'NL' then 'Netherlands'
    when 'IT' then 'Italy'
    when 'GB' then 'United Kingdom'
    when 'US' then 'United States'
    else 'Other'
    cond
```

end

In this expression, the table name is GEOGRAPHY_DIM, and the column label is Country Abbreviation.

🖳 Expression Editor
Name: Country
Description:
Type: Character
Expression Text:
<pre>case <<geography_dim.country abbreviation="">></geography_dim.country></pre>
Undo Redo Validate Expression Add to Expression
Eunctions Data Sources Categories: Functions: Character Character Character String Matching Counting Aggregation Date and Time DBCS Descriptive Statistics External Files Financial Mathematical
OK Cancel Help

- 6 Click Validate Expression to check for errors in the expression.
- 7 Click **ok** to apply the expression to the data item and to exit the Expression Editor dialog box.
- 8 Click **ox** to finish creating the data item and to exit the Data Item Properties dialog box.

Create Data Items from Hierarchies

Tip

If you use the **Physical Data** pane in the SAS Information Map Studio main window to "automatically" create a data item from a hierarchy (using either the arrow buttons in the window or the **Insert** option on the hierarchy's drop-down menu), then when the MDX code is generated for a query, the data item resolves to the set of members from the first level below the All level of that hierarchy.

If you want a hierarchy data item to resolve to something other than the first level's members, then you can use the Expression Editor to customize the data item's expression.

Technique

The steps in the following example describe how to create a category data item that represents the last-month member of a Year-Quarter-Month hierarchy named YQM:

- In SAS Information Map Studio, open the information map that you want to work with.
- 2 From the menu bar, select **Insert** ► **Data Item**.
- 3 On the **Definition** tab in the Data Item Properties dialog box, click **Edit** to open the Expression Editor dialog box.

- 4 In the **Type** field, select **OLAP** Category.
- 5 In the Expression Text area, enter the following expression:

[YQM].[All YQM].LastChild.LastChild.LastChild

📑 Expression	n Editor	×
Name:	Last Month	
 Description:		
<u>T</u> ype:	OLAP Category	
Expression T	'ext:	
[YQM].[All Y	YQM],LastChild,LastChild,LastChild	
+ - *	I AND OR NOT = < <= >= : (_) (_)	
Undo	Redo Validate Expression Add to Expression	
Functions	Data Sources	
Categories	s: Functions:	
Hierard Level F Logical Membe	sion Functions chy Functions Functions I runctions ic Functions inctions Functions Functions	
	OK Cancel <u>H</u> elp	

- 6 Click Validate Expression to check for errors in the expression.
- 7 Click **ox** to apply the expression to the data item and to exit the Expression Editor dialog box.
- 8 Click **ox** to finish creating the data item and to exit the Data Item Properties dialog box.

Glossary

access control

a method of controlling the type of access that each user has to data, to metadata, or to an application's functionality.

aggregate function

a function that summarizes data and produces a statistic such as a sum, an average, a minimum, or a maximum.

alias

(1) a logical representation of a physical table as it is defined in the metadata repository. An information map can use multiple aliases to represent the same table. (2) a name that is used in an information map to refer to a logical representation of a table. Each table that is a data source for an information map is a logical representation of a table and each logical representation must have a unique name.

catalog

See SAS catalog.

category

a data item whose distinct values are used to group measure data items, using an applied aggregate function.

classification

an attribute of data items that determines how they will be processed in a query. Data items can be classified as either categories or measures.

cube

a set of data that is organized and structured in a hierarchical, multidimensional arrangement. A cube includes measures, and it can have numerous dimensions and levels of data.

data item

in an information map, an item that represents either physical data (a table column, an OLAP hierarchy, or an OLAP measure) or a calculation. Data items are used for building queries. Data items are usually customized in order to present the physical data in a form that is relevant and meaningful to a business user.

data source

the physical data (cube or table), as it is defined in a SAS Metadata Repository, that an information map consumer can query through an information map. The metadata for the physical data provides SAS Information Map Studio with the information that it needs in order to access the physical data.

DBMS (database management system)

a software application that enables you to create and manipulate data that is stored in the form of databases.

detail data

nonsummarized (or partially summarized) factual information that pertains to a single area of interest, such as sales figures, inventory data, or human-resource data.

dimension

a group of closely related hierarchies. Hierarchies within a dimension typically represent different groupings of information that pertains to a single concept. For example, a Time dimension might consist of two hierarchies: (1) Year, Month, Date, and (2) Year, Week, Day. See also hierarchy.

d rill-through

the ability to retrieve and display to a user the detail data from which the summarized data in a cube is derived, when that detail data is stored in a separate data table.

drill-through table

a view, data set, or other data file that contains data that is used to define a cube. Drill-through tables can be used by client applications to provide a view from processed data into the underlying data source.

expression

a combination of data elements, literals, functions, and mathematical operators. An expression can be used to derive a value or to specify a condition that determines whether or how data is processed.

filter

in an information map, criteria that subset data. When a query is generated from an information map, the filter is converted to a query-language statement (for example, an SQL WHERE clause).

format

a pattern that SAS uses to determine how the values of a variable or data item should be written or displayed. SAS provides a set of standard formats and also enables you to define your own formats.

global macro variable

a macro variable that can be referenced in either global or local scope in a SAS program, except where there is a local macro variable that has the same name. A global macro variable exists until the end of the session or program.

heap

an area of memory that is allocated and released dynamically.

hierarchy

an arrangement of members of a dimension into levels that are based on parent-child relationships. Members of a hierarchy are arranged from more general to more specific. For example, in a Time dimension, a hierarchy might consist of the members Year, Quarter, Month, and Day. In a Geography dimension, a hierarchy might consist of the members Country, State or Province, and City. More than one hierarchy can be defined for a dimension. Each hierarchy provides a navigational path that enables users to drill down to increasing levels of detail. See also member, level.

information map

a collection of data items and filters that describes and presents data in a form that is relevant and meaningful to a business user. A user of a query and reporting application such as SAS Web Report Studio can easily build a business report by using the parts of an information map as the building blocks for queries.

inner join

a join between two tables that returns all of the rows in one table that have one or more matching rows in the other table.

join

(1) the act of combining data from two or more tables in order to produce a single result set. (2) a specification that describes how you want data from two or more tables to be combined. The specification can be in the form of Structured Query Language (SQL) programming code, or it can be done interactively through a software user interface.

level

an element of a dimension hierarchy. Levels describe the dimension from the highest (most summarized) level to the lowest (most detailed) level. For example, possible levels for a Geography dimension are Country, Region, State or Province, and City.

libref (library reference)

a short name for the full physical name of a SAS library. In the context of the SAS Metadata Repository, a libref is associated with a SAS library when the library is defined in the metadata repository.

login

a combination of a user ID, a password, and an authentication domain. Each login provides access to a particular set of computing resources. In a SAS metadata environment, each login can belong to only one individual or group. However, each individual or group can own multiple logins.

macro variable

a variable that is part of the SAS macro programming language. The value of a macro variable is a string that remains constant until you change it. Macro variables are sometimes referred to as symbolic variables.

MDX (multidimensional expressions) language

a standardized, high-level language that is used for querying multidimensional data sources. The MDX language is the multidimensional equivalent of SQL (Structured Query Language).

measure

(1) a data item whose values are aggregated (unless otherwise specified) and which can be used in computations or analytical expressions. Typically, these values are numeric. (2) a special dimension that usually represents numeric data values that are analyzed.

member

a name that represents a particular data element within a dimension. For example, September 1996 might be a member of the Time dimension. A member can be either unique or non-unique. For example, 1997 and 1998 represent unique members in the Year level of a Time dimension. January represents non-unique members in the Month level, because there can be more than one January in the Time dimension if the Time dimension contains data for more than one year.

metadata

data about data. For example, metadata typically describes resources that are shared by multiple applications within an organization. These resources can include software, servers, data sources, network connections, and so on. Metadata can also be used to define application users and to manage users' access to resources. Maintaining metadata in a central location is more efficient than specifying and maintaining the same information separately for each application.

metadata LIBNAME engine

the SAS engine that processes and augments data that is identified by metadata. The metadata engine retrieves information about a target SAS data library from metadata objects in a specified metadata repository.

metadata server

a server that provides metadata management services to one or more client applications. A SAS Metadata Server is an example.

OLAP (online analytical processing)

a software technology that enables users to dynamically analyze data that is stored in cubes.

physical data

data values that are stored on any kind of physical data-storage media, such as disk or tape.

prefilter

in an information map, a mandatory filter that pre-screens and subsets the data in its associated table before any other part of a query is run. The two types of prefilters are authorization-based prefilters and general prefilters. An authorization-based prefilter applies to a specific user or group, and a general prefilter applies to all users.

prompt

a parameter that enables a user to enter a value at run time.

query

a set of instructions that requests particular information from one or more data sources.

relationship

the association, between tables in an information map, that generates a database join in a query.

SAS application server

a server that provides SAS services to a client. In the SAS Open Metadata Architecture, the metadata for a SAS application server specifies one or more server components that provide SAS services to a client.

SAS catalog

a SAS file that stores many different kinds of information in smaller units called catalog entries. A single SAS catalog can contain several different types of catalog entries.

SAS Information Map

See information map.

SAS library

a collection of one or more files that are recognized by SAS and that are referenced and stored as a unit. SAS libraries can be defined in a SAS Metadata Repository to provide centralized definitions for SAS applications.

SAS Metadata Repository

a repository that is used by the SAS Metadata Server to store and retrieve metadata. See also SAS Metadata Server.

SAS Metadata Server

a multi-user server that enables users to read metadata from or write metadata to one or more SAS Metadata Repositories.

SAS OLAP Server

a SAS application server that provides access to multidimensional data. The data is queried using the multidimensional expressions (MDX) language.

SAS program

a group of SAS statements that guide SAS through a process or series of processes in order to read and transform input data and to generate output. The DATA step and the procedure step, used alone or in combination, form the basis of SAS programs.

SAS statement

a string of SAS keywords, SAS names, and special characters and operators that instructs SAS to perform an operation or that gives information to SAS. Each SAS statement ends with a semicolon.

SAS Stored Process

a SAS program that is stored on a server and which can be executed as requested by client applications. SAS Information Maps can use only SAS Stored Processes that run on a SAS Workspace Server.

SAS system option

an option that affects the processing of an entire SAS program or interactive SAS session from the time the option is specified until it is changed. Examples of items that are controlled by SAS system options include the appearance of SAS output, the handling of some files that are used by SAS, the use of system variables, the processing of observations in SAS data sets, features of SAS initialization, and the way SAS interacts with your host operating environment.

SAS Workspace Server

a SAS application server that provides access to Foundation SAS features such as the SAS programming language and SAS libraries.

SQL (Structured Query Language)

a standardized, high-level query language that is used in relational database management systems to create and manipulate database management system objects.

stored process

See SAS Stored Process.

system option

See SAS system option.

table

a two-dimensional representation of data, in which the data values are arranged in rows and columns.

user-defined format

See format.

XML (Extensible Markup Language)

a markup language that structures information by tagging it for content, meaning, or use. Structured information contains both content (for example, words or numbers) and an indication of what role the content plays. For example, content in a section heading has a different meaning from content in a database table.

Index

A

aggregating data filtering after 29 filtering before 34 aliases, table 44 alternate joins table aliases used in 44

С

categories creating filters with 34 conditional SQL code creating for data items 45 cubes displaying detail data for large cubes 22 making detail data available to 11

D

data items conditional SQL code for 45 creating from hierarchies 47 detail data displaying for large cubes 22 making available for drill-through 11 making available to cubes 11 making available to information maps 17 making available to OLAP servers 13 drill-through making detail data available for 11 drill-through tables making column labels available 19

F

filtering aggregating data after 34 aggregating data before 29 filters creating 27 hiding from SAS Web Report Studio 38 FMTSEARCH system option making user-defined formats available 4 formats making user-defined formats available 4

H

HAVING clause 28 heap size 25 hierarchies creating data items from 47

I

information maps making detail data available to 17 stored process input parameters with 43 stored processes used with 40 input parameters, stored process 43

J

Java heap size 25 joins table aliases in 44

L

labels, drill-through table column making available 19 LIBRARY libref making user-defined formats available 4

Μ

measures creating filters with 29

0

OLAP reports display drill-through table column labels in 19 drill-through to detail data for 11 OLAP servers making detail data available to 13

Р

parameters, stored process 43 prefilters hiding from SAS Web Report Studio 38

R

recursive joins table aliases used in 44 row-level access to tables 8

S

SAS programs converting to stored processes 6 SAS Web Report Studio hiding prefilters from 38 security restricting access to table rows 8 SQL code conditional 45 HAVING clause 28 WHERE clause 28 stored processes converting SAS programs to 6 information maps used with 40 input parameters with information maps 43 restricting access to table rows 8

Т

table aliases 44 alternate joins and 44 recursive joins and 44 table column labels, drill-through making available 19 table columns creating filters with 38 tables restricting access to rows 8

U

user-defined formats making available 4

W

WHERE clause 28

Your Turn

If you have comments or suggestions about SAS Information Map Studio 3.1: Tips and Techniques, please send them to us on a photocopy of this page, or send us electronic mail.

For comments about this book, please return the photocopy to

SAS Publishing SAS Campus Drive Cary, NC 27513 E-mail: yourturn@sas.com

For suggestions about the software, please return the photocopy to

SAS Institute Inc. Technical Support Division SAS Campus Drive Cary, NC 27513 E-mail: suggest@sas.com

SAS Publishing gives you the tools to flourish in any environment with SAS[®]!

Whether you are new to the workforce or an experienced professional, you need a way to distinguish yourself in this rapidly changing and competitive job market. SAS Publishing provides you with a wide range of resources, from software to online training to publications to set yourself apart.

Build Your SAS Skills with SAS Learning Edition

SAS Learning Edition is your personal learning version of the world's leading business intelligence and analytic software. It provides a unique opportunity to gain hands-on experience and learn how SAS gives you the power to perform.

support.sas.com/LE

Personalize Your Training with SAS Self-Paced e-Learning

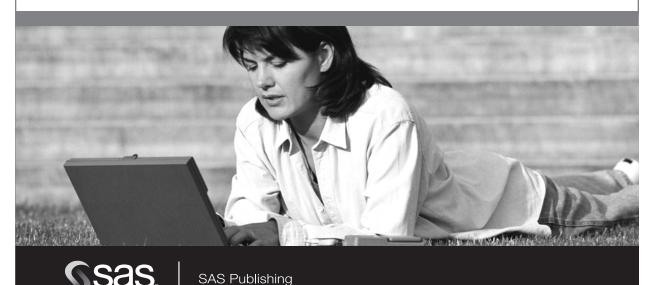
You are in complete control of your learning environment with SAS Self-Paced e-Learning! Gain immediate 24/7 access to SAS training directly from your desktop, using only a standard Web browser. If you do not have SAS installed, you can use SAS Learning Edition for all Base SAS e-learning.

support.sas.com/selfpaced

Expand Your Knowledge with Books from SAS Publishing

SAS Press offers user-friendly books for all skill levels, covering such topics as univariate and multivariate statistics, linear models, mixed models, fixed effects regression and more. View our complete catalog and get free access to the latest reference documentation by visiting us online.

support.sas.com/pubs



SAS and all other SAS Institute Inc. product or service names are registered trademarks or trademarks of SAS Institute Inc. in the USA and other countries. (a) indicates USA registration Other brand and product names are trademarks of their respective companies. (c) 2005 SAS Institute Inc. All rights reserved. 345193US.0805