



SAS Publishing



SAS[®] Information Map Studio 2.1: Tips and Techniques

The correct bibliographic citation for this manual is as follows: SAS Institute Inc. 2005. *SAS® Information Map Studio 2.1: Tips and Techniques*. Cary, NC: SAS Institute Inc.

SAS® Information Map Studio 2.1: Tips and Techniques

Copyright © 2005, SAS Institute Inc., Cary, NC, USA

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

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

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

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, July 2005

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. ® indicates USA registration.

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

Contents

Chapter 1 △ **About This Book** 1

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

Chapter 2 △ **Tips for Setting Up Your Environment** 3

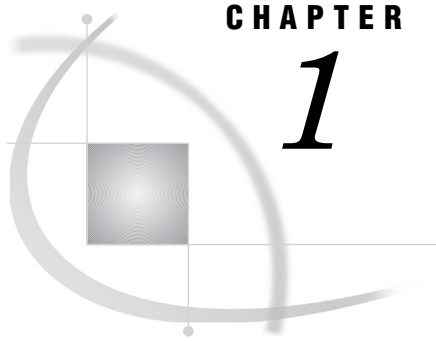
- Make User-Defined Formats Available 3
- Make SAS Programs Available as SAS Stored Processes for Information Maps 6
- Restrict Access to Specified Rows by Using a SAS Stored Process 8
- Make Detail Data Available for Drill-Through 10
- Display Detail Data for a Large Cube 12
- Increase the Java Heap Size for SAS Information Map Studio 13

Chapter 3 △ **Tips for Creating Information Maps** 15

- Use a SAS Stored Process with an Information Map 15
- Use SAS Stored Process Input Parameters with an Information Map 17
- Use a Table Alias 18
- Create Conditional SQL Code 19
- Specify a User-Defined Format That Groups Values 20
- Create Data Items from Hierarchies 21

Glossary 23

Index 29



CHAPTER

1

About 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 1.0.1 and 2.1. 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.

Intended Audience

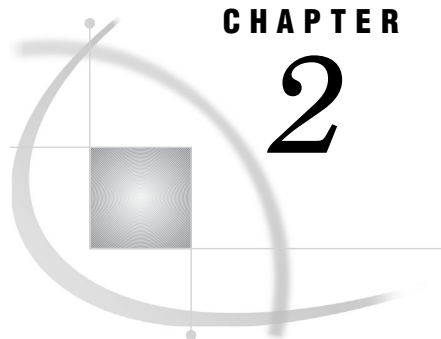
This book assumes that the reader has a working understanding of SAS Information Map Studio and SAS Information Maps. It is also useful for the reader to have some knowledge of related SAS business intelligence products such as SAS Management Console.

Online Help

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

Use any of the following methods to get Help for SAS Information Map Studio:

- From the menu bar, select **Help ► SAS Information Map Studio**.
- Click **Help** (when available) in the application windows and dialog boxes.
- Press F1 in most of the application windows and dialog boxes.



CHAPTER

2

Tips for Setting Up Your Environment

<i>Make User-Defined Formats Available</i>	3
<i>Tip</i>	3
<i>Technique</i>	4
<i>Use the LIBRARY Libref</i>	4
<i>Use the FMTSEARCH System Option</i>	4
<i>Make SAS Programs Available as SAS Stored Processes for Information Maps</i>	6
<i>Tip</i>	6
<i>Technique</i>	6
<i>Restrict Access to Specified Rows by Using a SAS Stored Process</i>	8
<i>Tip</i>	8
<i>Technique</i>	8
<i>Example Scenario: Description</i>	8
<i>Example Scenario: The Basic Query</i>	8
<i>Example Scenario: The Subquery</i>	8
<i>Example Scenario: The Completed Query</i>	9
<i>Example Scenario: The Stored Process</i>	9
<i>Make Detail Data Available for Drill-Through</i>	10
<i>Tip</i>	10
<i>Technique</i>	10
<i>Make Detail Data Available to a Cube</i>	10
<i>Make Detail Data Available to an OLAP Server</i>	10
<i>Make Detail Data Available to an Information Map</i>	11
<i>Display Detail Data for a Large Cube</i>	12
<i>Tip</i>	12
<i>Technique</i>	12
<i>Increase the Java Heap Size for SAS Information Map Studio</i>	13
<i>Tip</i>	13
<i>Technique</i>	13

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 performing 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')
```

\triangle

Use the FMTSEARCH System Option

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

- For all but the z/OS environment, perform 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"
```

- 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 fmplib1 "SAS-config-dir\Lev1\Data\orformat"  
-fmtsearch (fmplib1.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 **fmplib1**:

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

\triangle

- For the z/OS environment, perform the following steps:
 - 1 Add an AUTOEXEC system option to the SAS launch command as shown in the following example:

* 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.


```
SAS-config-dir/Levl/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/Levl/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, perform 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, perform 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.data1;
  /* 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 the Data Library Manager plug-in for SAS Management Console, SAS ETL 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. Δ

The following example shows the modified sample program:

```
*ProcessBody;
Libname source 'path-to-source-data';
Libname stpout (work); /* The output library and table must already */
                        /* be defined in metadata */

Data stpout.result_set;
  Set source.data1;
  /* more code */
run;
```

4 Save the program.

5 Define the program in the metadata repository as a stored process (using either SAS Enterprise Guide or the Stored Process Manager plug-in for SAS Management Console). 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 17.)

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 the Stored Process Manager plug-in, 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 15.

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

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 detail 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 of 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 table for Drill-Through from list** (to select a table from the list of tables on the Drill-Through page)

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 detail 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, perform the following steps:

- 1 In the Data Library Manager plug-in for 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.
 - 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 of the properties dialog box.
- 2 Click [Advanced Options](#).
- 3 Select the **Library is Pre-Assigned** check box on the **Pre-Assign** tab in the Advanced Options dialog box.

- 4 Select the **Assign** tab of the properties dialog box and ensure that the selected application server is the server container that contains your OLAP server.
- 5 Click **OK** to save the settings.
- 6 Restart the OLAP server.

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.
- 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.

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 13.

To increase the number of drill-through rows that your OLAP server can handle, you can change an OLAP server definition setting in the Server Manager plug-in for SAS Management Console by performing 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 the Server Manager plug-in for SAS Management Console, find the node that represents your physical OLAP server (indicated by the  icon).
- 2 Right-click the node, and then select **Properties** from the drop-down menu.
- 3 In the properties dialog box, select the **Options** tab, and then click **Advanced Options**.
- 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.
- 5 Click **OK** 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 maximum heap size for SAS Information Map Studio has the following default values:

- SAS Information Map Studio 1.0.1 has a maximum heap size of 64MB.
- SAS Information Map Studio 2.1 has a maximum heap size of 1024MB.

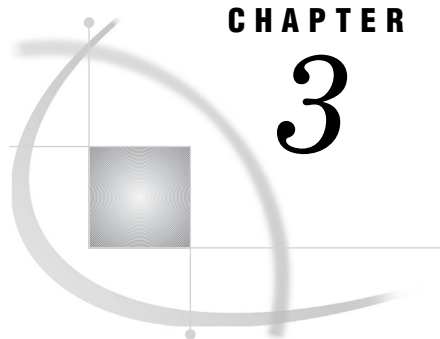
To increase the heap size for SAS Information Map Studio, you can change the settings for its minimum heap size (**-Xms**) and maximum heap size (**-Xmx**) by performing the following steps:

- 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 For SAS Information Map Studio 1.0.1, add the following heap size parameters to the list of **CommandLineArgs** parameters. For SAS Information Map Studio 2.1, 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.



CHAPTER

3

Tips for Creating Information Maps

<i>Use a SAS Stored Process with an Information Map</i>	15
<i>Tip</i>	15
<i>Technique</i>	15
<i>Use SAS Stored Process Input Parameters with an Information Map</i>	17
<i>Tip</i>	17
<i>Technique</i>	17
<i>Use a Table Alias</i>	18
<i>Tip</i>	18
<i>Technique</i>	18
<i>Example 1: Recursive Join</i>	18
<i>Example 2: Alternate Joins between a Pair of Tables</i>	18
<i>Create Conditional SQL Code</i>	19
<i>Tip</i>	19
<i>Technique</i>	19
<i>Specify a User-Defined Format That Groups Values</i>	20
<i>Tip</i>	20
<i>Technique</i>	20
<i>Create Data Items from Hierarchies</i>	21
<i>Tip</i>	21
<i>Technique</i>	21

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 ETL Studio, SAS Enterprise Guide, or the steps in “Make SAS Programs Available as SAS Stored Processes for Information Maps” on page 6), perform 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 17.) Δ

- 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. Perform 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 the Data Library Manager plug-in for SAS Management Console, SAS ETL 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 Select **Tools** \blacktriangleright **Stored Processes** from the SAS Information Map Studio menu bar and select the desired stored process.
- 6 Click 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 perform 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 the Stored Process Manager plug-in of SAS Management Console, SAS ETL 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 15.

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

Technique

The following examples describe scenarios where you would want to create a table alias:

Example 1: Recursive Join

You have an Employee table that includes the Employee_ID, Employee_Name, and Manager_ID columns. You create an alias for 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 2: Alternate Joins between a Pair of Tables

You have 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 copy 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 data items.

Technique

You can use a CASE statement in a 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 data item that converts the abbreviations for country names to their full names:

- 1 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 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'
end
```

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

- 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 **OK** to finish creating the data item and to exit the Data Item Properties dialog box.

Specify a User-Defined Format That Groups Values

Tip

Some user-defined formats are defined to group ranges of values and to represent each range with a single formatted value. For example, a format might group the data item values of 0 through 50 and display a formatted value of **LOW** for that range of values.

If you want to assign this type of grouping format to a data item, then you must specify the format in the data item's expression.

Technique

To specify a format that groups the values of a data item, perform the following steps in SAS Information Map Studio:

- 1 Open the information map that you want to work with.
- 2 Open the Data Item Properties dialog box for a data item by double-clicking the data item or by right-clicking the data item and selecting **Properties** from the drop-down menu.
- 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 **Character**.
- 5 In the **Expression Text** field, you will see the data item's expression. Cut or copy the expression from the field, replace it with the PUT function, and use the original expression as the value for the first argument in the function call:

```
PUT(data-item-expression,user-defined-format)
```

In this function call, **data-item-expression** is the expression that you cut or copied from the field, and **user-defined-format** is the name of your format.

Note: Do not enclose the format name in quotation marks. \triangle

- 6 Click **Validate Expression** to check for errors in the expression.
- 7 Click **OK** to apply the validated expression to the data item and to exit the Expression Editor dialog box.
- 8 If a message appears that says that your format settings have been changed, click **OK** to dismiss the message dialog box.
- 9 On the **Classifications/Formats** tab, select **Default Format** as the format type.
- 10 Click **OK** 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:

- 1 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
```

- 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 **OK** 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.

alias

1) a logical representation of a physical table. An information map can use multiple aliases to represent the same physical table. 2) a name that is used in an information map to refer to a logical representation of a physical data table. Each table within an information map is a logical representation of a physical data 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

an item in an information map that is either a logical view of a physical data field or a calculation. Data items are usually customized to produce a logical, business-relevant view of the physical data. Related data items are grouped into information maps and are used for building queries and reports.

data source

physical data (cube or table) that an information map accesses. An information map accesses its data sources via their definitions in the SAS Metadata Repository.

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.

expression

the combination of data elements, literals, functions, and mathematical operators that are used to derive the value of a data item or a filter.

filter

criteria in a query (such as in an SQL WHERE clause) for subsetting a result set.

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

in an application, an area of memory in which the storage is dynamically allocated and released.

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 provides a business-relevant view of physical data. Users of query and reporting applications such as SAS Web Report Studio can easily build business reports by using information maps as the building blocks for their reports.

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

1) a name that is temporarily associated with a SAS library. The complete name of a SAS table consists of two words, separated by a period. The libref, which is the first

word, indicates the library. The second word is the name of the specific SAS table. For example, in VLIB.NEWBDAY, the libref VLIB tells SAS which library contains the table NEWBDAY. You assign a libref with a LIBNAME statement or with an operating system command. 2) a valid SAS name that serves as a shortcut name to associate with metadata objects that are in a 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 or column whose value 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 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 repository

a collection of related metadata objects, such as the metadata for a set of tables and columns that are maintained by an application. A SAS Metadata Repository is an example.

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 in a database management system.

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. See also join.

row-level security

controls that define access to data at a low level, specifying who can access particular rows within a table. Row-level controls are typically used to subset data by a user characteristic such as employee ID or organizational unit. For example, a table that contains patient medical information might have row-level controls that enable each doctor to see only those rows that contain data about that doctor's patients.

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 Maps

See information map.

SAS library

a collection of one or more database files that are recognized by SAS and that are referenced and stored as a unit.

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 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**
- aliases, table 18
 - alternate joins
 - table aliases used in 18
- C**
- conditional SQL code
 - creating for data items 19
 - cubes
 - displaying detail data for large cubes 12
 - making detail data available to 10
- D**
- data items
 - conditional SQL code for 19
 - creating from hierarchies 21
 - detail data
 - displaying for large cubes 12
 - making available for drill-through 10
 - making available to cubes 10
 - making available to information maps 11
 - making available to OLAP servers 10
 - drill-through
 - making detail data available for 10
- F**
- FMTSEARCH system option
 - making user-defined formats available 4
 - formats
 - grouping values with user-defined formats 20
 - making user-defined formats available 3
- G**
- grouping values
 - user-defined formats for 20
- H**
- heap size 13
 - hierarchies
 - creating data items from 21
- I**
- information maps
 - making detail data available to 11
 - stored process input parameters with 17
 - stored processes used with 15
 - input parameters, stored process 17
- J**
- Java heap size 13
 - joins
 - table aliases in 18
- L**
- LIBRARY libref
 - making user-defined formats available 4
- O**
- OLAP reports
 - drill-through to detail data for 10
 - OLAP servers
 - making detail data available to 10
- P**
- parameters, stored process 17
- R**
- recursive joins
 - table aliases used in 18
 - row-level access to tables 8
- S**
- SAS programs
 - converting to stored processes 6
 - security
 - restricting access to table rows 8
 - SQL code, conditional 19
 - stored processes
 - converting SAS programs to 6
 - information maps used with 15
 - input parameters with information maps 17
 - restricting access to table rows 8
- T**
- table aliases 18
 - alternate joins and 18
 - recursive joins and 18
 - tables
 - restricting access to rows 8
- U**
- user-defined formats
 - grouping values with 20
 - making available 3

Your Turn

If you have comments or suggestions about *SAS Information Map Studio 2.1: 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

