



SAS Publishing



SAS[®] Web Analytics 5.1

Administrator's Guide

The correct bibliographic citation for this manual is as follows: SAS Institute Inc. 2005. *SAS® Web Analytics 5.1: Administrator's Guide*. Cary, NC: SAS Institute Inc.

SAS® Web Analytics 5.1: Administrator's Guide

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 this software and related documentation by the U.S. government is subject to the Agreement with SAS Institute and the restrictions 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 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	△ Overview of the SAS Web Analytics 5.1 Solution	1
About the Documentation for System Administrators		2
What's Included in the SAS Web Analytics 5.1 Solution		2
Why Does My Organization Need the SAS Web Analytics 5.1 Solution?		2
Web Mart and Data Structures		4
Input for the SAS Web Analytics Solution		8
ETL Processes for the SAS Web Analytics Solution		10
Output from the SAS Web Analytics Solution		13
Getting Started with SAS Web Analytics 5.1: Administration Tasks		16
Chapter 2	△ Creating a New Web Mart	19
Overview: Creating a New Web Mart		19
Creating a New Web Mart by Using the %WAETL Macro		19
Creating a New Web Mart by Using the SAS e-Data ETL Administrator		20
Registering a New Web Mart		24
Chapter 3	△ Customizing the Properties of a Web Mart	27
Overview: The Properties of a SAS Web Analytics Web Mart		28
The Properties Window of the SAS e-Data ETL Administrator		30
The Detail Tables Properties		31
The Temporary Location Properties		34
The Web Log Location Properties		35
The Processing Properties		35
The Parsing Properties		41
The Filtering Properties		45
The Compressed Files Properties		50
The Execution Tuning Properties		51
The Advanced Customizations Frame		52
How the Customizing Process Works in SAS e-Data ETL Software		61
Chapter 4	△ Setting Up ETL Processing for SAS e-Data ETL Software	63
Working with Web Logs		63
Using the Daily.sas Program to Perform the ETL Processes		69
Using the %EDATAETL Macro to Extract and Load Web Log Data		71
Chapter 5	△ The SAS e-Data ETL Extract Process	73
Overview of the Extract Process		73
Running the Extract Process		77
Chapter 6	△ Reference Notes for SAS e-Data ETL Extract Process	79
Overview: The Modules in the Extract Process		80
The Modules in the Extract Process		80

The Temporary Working Area	95
Chapter 7 △ The SAS e-Data ETL Load Process	97
Overview of the Load Process	97
Running the Load Process	98
Chapter 8 △ The Modules in the SAS e-Data ETL Load Process	99
Overview: The Modules of the Load Process	99
The Modules in the Load Process	100
Contents of the Detail Library	105
Chapter 9 △ Using the SAS Web Analytics Administrator to Manage Your Web Mart(s)	111
Navigating in the SAS Web Analytics Administrator	112
Chapter 10 △ Administration of SAS Web Analytics Reports	131
Working with Report Groups	131
Creating a New Report	137
Drillable Reports	144
Variables You Can Set for Reports	146
Chapter 11 △ Scorecard Administration	149
Overview: Scorecard	149
Data Requirements for a Scorecard	150
Defining a New Scorecard: Preparation	150
Running a Custom Scorecard	152
How to Create a New Scorecard	152
Adding Variables That Are Not in the Web Log to a Scorecard	156
Testing the Scorecard Output: The %WADECIDE Macro	157
The Table Layout of the Scorecard Report	158
How Special Cases are Handled	161
Chapter 12 △ Dashboard Administration	163
Overview: Dashboard	163
How to Create a New Dashboard	165
Testing the Dashboard Output: The %WADECIDE Macro	170
Chapter 13 △ Setting Up a Segmentation Report	171
Data Requirements for the Default Segmentation Report	171
Creating a Segmentation Report	171
Testing a Segmentation Report	174
Running a Segmentation Report	174
Chapter 14 △ Using Stored Processes in the SAS Web Analytics Application	175
Working with Stored Processes for Report Definitions	175
Appendix 1 △ Macro Reference for the SAS Web Analytics 5.1 Solution	179
Overview: Macros for SAS Web Analytics 5.1	179

The %WADECIDE Macro 180

The %WAETL Macro 184

Appendix 2 △ Alphabetical List of SAS Web Analytics Reports 189

List of Traffic Reports 189

List of Non-Traffic Reports 191

Alphabetical List of SAS Web Analytics Reports 191

Appendix 3 △ The SUMMARY Engine 193

The Waadmsum Data Set 193

Overview: The Waconfig Data Set 200

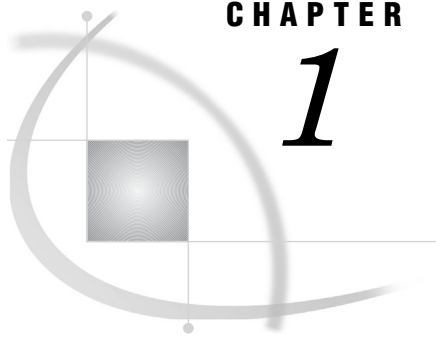
The Parameters in the Waconfig Data Set 200

Changing Values in the Waconfig Data Set 204

Appendix 4 △ The Summary Data Sets for SAS Web Analytics 5.1 209

Glossary 241

Index 245



CHAPTER

1

Overview of the SAS Web Analytics 5.1 Solution

<i>About the Documentation for System Administrators</i>	2
<i>What's Included in the SAS Web Analytics 5.1 Solution</i>	2
<i>What Is SAS e-Data ETL Software?</i>	2
<i>What Is SAS Web Analytics 5.1 Software?</i>	2
<i>Why Does My Organization Need the SAS Web Analytics 5.1 Solution?</i>	2
<i>The Business Objectives of Retail Industries</i>	3
<i>Business Objectives of Financial and Service Organizations</i>	3
<i>Business Objectives of Telecommunications Organizations</i>	3
<i>Capabilities That Are Required to Meet Business Objectives</i>	3
<i>Web Mart and Data Structures</i>	4
<i>SAS Business Intelligence Architecture</i>	4
<i>Directory Structure</i>	4
<i>Webdatasrv Directory</i>	5
<i>E-data-etl Directory</i>	6
<i>Data Directory</i>	6
<i>Temporary Working Directory</i>	7
<i>The Directory that Contains the Web Logs</i>	7
<i>Creating Additional Directories for a New Web Mart</i>	7
<i>Input for the SAS Web Analytics Solution</i>	8
<i>Types of Data Sources for the SAS Web Analytics Solution</i>	8
<i>Web Server Log Files</i>	8
<i>Other Data Sources</i>	9
<i>ETL Processes for the SAS Web Analytics Solution</i>	10
<i>How the Extract Process Prepares to Read the Web Logs</i>	11
<i>The Extract Process: About Temporary Data Sets</i>	11
<i>Customizing the Extract Process</i>	12
<i>The Load Process</i>	12
<i>Customizing the Summarizing and Reporting Processes in SAS Web Analytics</i>	12
<i>Output from the SAS Web Analytics Solution</i>	13
<i>The SAS Web Analytics Report Viewer</i>	13
<i>Supplied Reports</i>	14
<i>Setting Up Reports</i>	15
<i>The Scorecard</i>	15
<i>The Dashboard</i>	15
<i>The Segmentation Report</i>	15
<i>Path Analysis Reports</i>	15
<i>The Funnel Report</i>	16
<i>Getting Started with SAS Web Analytics 5.1: Administration Tasks</i>	16

About the Documentation for System Administrators

After you have installed all of the components of the SAS Web Analytics 5.1 solution successfully, use this documentation to help you set up and configure an analysis-ready Web mart. Setup and configuration includes the following tasks:

- Initialize a Web mart.
- Register a Web mart.
- Edit Web mart properties in the SAS e-Data ETL Administrator.
- Customize the ETL processing of the detail data sets by using the SAS Web Analytics Administrator.
- Configure SAS summary data sets by using the SAS Web Analytics Administrator. These summary data sets are queried from the SAS Web Analytics Report Viewer to display the reports.
- Process the Web log data (using the Daily.sas program).
- Create new traffic reports.
- Set up the variables and the metrics for scorecards, dashboards, and segmentation reports by using the SAS Web Analytics Administrator.

What's Included in the SAS Web Analytics 5.1 Solution

The software bundle that comprises the SAS Web Analytics solution includes SAS®9 and the following applications:

- SAS e-Data ETL software
- SAS Web Analytics 5.1 software

What Is SAS e-Data ETL Software?

SAS e-Data ETL software is a data sourcing tool for clickstream data. It is used to extract large volumes of raw clickstream data, identify the data by visit, and then parse, filter, and load the data into SAS detail data sets. You use the SAS e-Data ETL software to create Web marts and to edit Web mart properties.

What Is SAS Web Analytics 5.1 Software?

SAS Web Analytics 5.1 software is a clickstream analysis and graphic reporting tool that performs the following functions:

- reads the detail data sets that are generated by SAS e-Data ETL software
- generates the summary data sets (part of the Web mart) from the detail data sets
- reads data into the Web mart from sources other than Web logs (optional)
- provides a Web-based interface for dynamically producing reports about Web site visitors and their behavior

Why Does My Organization Need the SAS Web Analytics 5.1 Solution?

Organizations that do business on the Web typically want to increase the revenue that is generated from their Web sites or to increase customer retention. However, the

nature of Web server log files makes it difficult to transform the raw Web server log data into information that organizations can easily use to accomplish their key business objectives. The SAS Web Analytics solution facilitates this transformation of raw data into useful information.

The Business Objectives of Retail Industries

Specific business objectives can vary across industries. Retail organizations typically have the following key business objectives for their Web sites:

- understanding the effects of online and offline campaigns in order to improve the perceived value of the Web site for customers or to improve customer experience at their Web site
- providing insight into distinct browsing, interest, and buying patterns
- reducing marketing expenses by targeting the right customers

Business Objectives of Financial and Service Organizations

Financial and services organizations typically have the following key business objectives for their Web sites:

- reducing costs by encouraging customers to use the Web site rather than more expensive call centers or personal visits to loan officers
- increasing customer “wallet share” by taking advantage of cross-selling and up-selling of other related products

Business Objectives of Telecommunications Organizations

Telecommunications organizations typically have the following business objectives for their Web sites:

- efficiently handling the enormous volumes of Web server log data that must be processed daily
- integrating a wide variety of data sources, including wireless logs, contract data, and point-of-sales information, in order to obtain a more complete view of customers

Capabilities That Are Required to Meet Business Objectives

In order to accomplish the business objectives of your organization, regardless of its specific type, you need the following capabilities:

- the ability to read and extract Web server log data using a process that can be easily customized
- the ability to transform the data into a useful format
- the ability to load the data into an analysis-ready Web mart
- the ability to generate dynamic, customizable reports that provide the necessary information for your organization to accomplish its key objectives

The SAS Web Analytics solution provides these capabilities. The software enables a wide variety of organizations to better understand the activity on their Web sites and, in turn, helps them to accomplish their key business objectives.

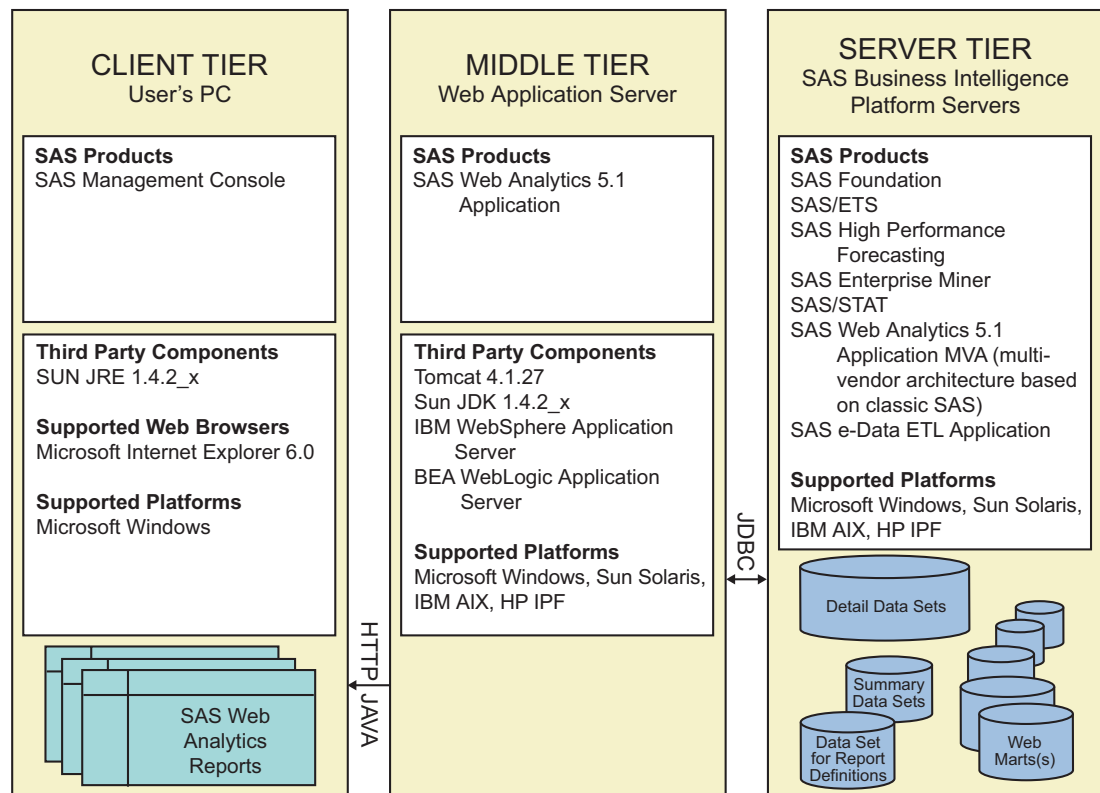
Web Mart and Data Structures

SAS Business Intelligence Architecture

The SAS Web Analytics 5.1 solution is built upon the three-tier architecture of the SAS®9 Business Intelligence platform, which is made up of the following tiers:

Server tier	The ETL processes and the SAS programs that produce reports reside on the server tier.
Middle tier	The Java code for the SAS Web Analytics Administrator and the SAS Web Analytics Report Viewer resides on the middle tier.
Client tier	The Web browser that invokes the SAS Web Analytics Administrator and Report Viewer resides on the client tier.

Display 1.1 The Three-Tier Architecture of SAS Web Analytics 5.1

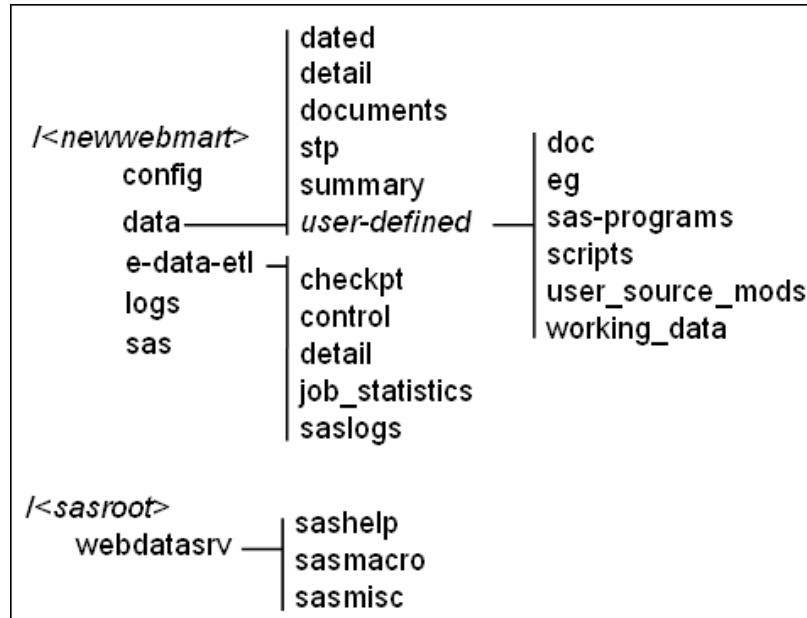


Directory Structure

Several directories are created when you create a new Web mart, and they are always associated with that specific Web mart. Other directories exist in your SAS root directory path. Additionally, you can create directories that might be needed for

customized processing. The following diagram illustrates the directory structure for a SAS Web Analytics Web mart:

Figure 1.1 Directory Structure for a SAS Web Analytics Web Mart



The following directories are significant to the administration of SAS Web Analytics:

- **webdatasrv**
- **e-data-etl**
- **data**
- a user-defined directory that contains the Web logs to be processed
- a temporary working directory

Webdatasrv Directory

When you install SAS e-Data ETL software in the Windows operating environment, the **webdatasrv** directory (Figure 1.1 on page 5) is created in your SAS root directory (in the server tier). The **webdatasrv** directory contains—among other libraries—the Sashelp library, which contains the Wbetl catalog. The Sashelp.Wbetl catalog is significant because it contains the source entries for the SAS e-Data ETL processes.

Note: An example of the path to the **webdatasrv** directory is **C:\Program Files\SAS\SAS 9.1\webdatasrv**. △

The **webdatasrv** directory contains the following libraries:

- | | |
|-----------|---|
| Sashelp | contains the distributed catalogs and data sets for the SAS e-Data ETL application. |
| Sasmacros | contains the macros for the SAS e-Data ETL application. |
| Sasmisc | contains miscellaneous SAS program files. |

Note: If you are installing SAS e-Data ETL software under UNIX, then the directory that contains the miscellaneous files is the **misc** directory. △

E-data-etl Directory

When you initialize a new Web mart by using the %WAETL macro, the %WAETL macro creates a set of permanent directories for that Web mart on the server tier (a SAS Business Intelligence Platform server) in a directory location that you specify.

The **e-data-etl** directory contains the libraries that contain the data sets that control the SAS e-Data ETL software processes. An **e-data-etl** directory exists for each Web mart that is created (see Figure 1.1 on page 5).

The following table describes the libraries that are automatically created within the **e-data-etl** directory for each Web mart:

Table 1.1 Libraries within an E-data-etl Directory

Name	Description
Checkpoint	Contains the contents of the detail data set from the last run of the Load process.
Control	Contains modified data sets from the Sashelp library. Examples of data sets that you might customize include Wbconfig and Wbfields.
Detail	Contains detail data sets such as Weblog_Detail_1.sas7bdat.
Job_Statistics	Contains the Job_Statistics data set for the Extract and Load processes. Using the Job_Statistics data set, you can produce reports to examine the success and run time of a process.
Saslogs	Contains the SAS logs from the child jobs that are run in separate, concurrent SAS sessions (to optimize the jobs). Save these logs so that you can provide them to SAS Technical Support if you have a problem.
Summary	Contains data sets for internal use only.

Data Directory

The **data** directory for each Web mart contains the detail data sets, summary data sets, and other data sets that are stored in the following directories:

Table 1.2 Directories within the Data Directory

Name	Description
dated	The storage area for the daily session (or visit) summary data sets. The daily session summary data sets are named according to the following convention: Session_yyyymmdd. For example, the daily summary data set that includes all of the sessions on January 15, 2004, is named Session_20040115.
detail	The storage area for the daily detail data sets. The detail data sets are named according to the following convention: Detail_yyyymmdd. For example, the daily detail data set for January 15, 2004, is named Detail_20040115.
documents	The location of the metadata data sets for the SAS Web Analytics reports.
stp	The storage area for the temporary output data sets that are generated by the stored processes. The SAS Web Analytics Report Viewer looks for the output data set(s) in the stp directory in order to create the requested report. There are no permanently stored entries in this directory; all stored process output is deleted after it is displayed.

summary	The storage area for the summarized data sets, which are named according to the following convention: the descriptive name that identifies the type of data that has been summarized (such as Referrer_Search_Term) is followed by a string of text that identifies the time interval for the summarized data (such as _DAY, _WEEK, _MONTH, _QTR, or _YEAR). For example, Referrer_Search_Term_MONTH.sas7bdat specifies the summary data set that contains all of the referrer search terms that were summarized by month (for a specified Web mart).
user-defined directory	A directory that you have defined if you need a separate storage location for supplemental data. For example, you can use this directory to store data for a metric (such as server CPU usage) that is not created by the SAS Web Analytics ETL processes.

Temporary Working Directory

SAS e-Data ETL software creates a temporary working directory for the data sets that are created during the Extract and Load processes. You specify the name of this directory during the creation of the Web mart. See the Temporary Location step (“Creating a New Web Mart by Using the SAS e-Data ETL Administrator” on page 20) for more information.

The Directory that Contains the Web Logs

The SAS e-Data ETL Extract process looks for Web logs to process (for a selected Web mart) in the location that you specify in the SAS e-Data ETL Administrator. To either specify or change the location of the Web logs to process, see “The Web Log Location Properties” on page 35.

Creating Additional Directories for a New Web Mart

You might want to create some additional directories after defining a new Web mart. You should store these directories in the same directory path as the directories that are automatically created, within the **e-data-etl** directory. The following table lists the directories you can manually create, along with descriptions of what each directory might contain.

Table 1.3 Web Mart Directories That You Can Create

Name	Description
doc	Contains documentation that you might want to add. For example, you might list customizations that you have made.
eg	Contains site-specific Enterprise Guide projects if you are using SAS Enterprise Guide.
sas_programs	Contains the SAS programs that are used during processing. For example, you can store the SAS code that is necessary to allocate the libraries or to run the Extract and Load processes. You can use these SAS programs with your scheduling package to automate your SAS e-Data ETL processes.
scripts	Contains VBScripts or Perl scripts that are used during the automation process.

Name	Description
user_source_mods	Contains modifications to source entries in the Wbectl catalog. The Usermods library reference points to this directory.
working_data	Contains the Web logs that SAS e-Data ETL software needs to process. These logs need to be moved from their original storage location to this directory in order to be processed. You can use scripts to automate the moving of the Web logs.

Input for the SAS Web Analytics Solution

Types of Data Sources for the SAS Web Analytics Solution

The SAS Web Analytics solution can read the following types of data sources:

- Web server log files (by using e-Data ETL software)
- non-Web log sources (formatted as SAS data sets) such as demographic data about customers from your organization's sales channels

Web Server Log Files

The data in a typical log file from a Web server looks similar to the following display:

Display 1.2 Example of the Output from a Typical Web Server Log File

```
202.77.159.98 - - [05/Apr/1999:02:49:20 -0400] "GET / HTTP/1.1" 200 17434 "-" "Mozilla/4.0
(compatible; MSIE 4.01; MSIECrawler; Windows NT)"
cosmo.dartmouth.edu - - [05/Apr/1999:02:49:25 -0400] "GET
/usergroups/sugi/sugi21/abstracts/abs148.html HTTP/1.0" 200 2434
"http://informant.dartmouth.edu/" "The Informant"
taz.northernlight.com - - [05/Apr/1999:02:49:48 -0400] "GET /service/techsup/faq/af/af111.html
HTTP/1.1" 304 - "-" "Gulliver/1.2"
cache2.kolumbus.net - - [05/Apr/1999:02:49:53 -0400] "GET /SASstyle.css HTTP/1.0" 304 - "-"
"Mozilla/3.01 (compatible;)"
t3o28p45.telia.com - - [05/Apr/1999:02:51:14 -0400] "GET / HTTP/1.1" 200 17434 "-" "Mozilla/4.0
(compatible; MSIE 4.01; Windows 95)"
proxy1b.isu.net.sa - - [05/Apr/1999:02:52:35 -0400] "GET / HTTP/1.0" 200 17434 "-" "Mozilla/4.0
(compatible; MSIE 4.01; Windows 95)"
ncahost.nippon-cargo.co.jp - - [05/Apr/1999:02:52:41 -0400] "GET
/japan/software/technologies/univ_acc.html HTTP/1.0" 200 4549
"http://www.infoseek.co.jp/Titles?col=Search+These+Results&qt=%CA%D1%B4%B9&oq=%A5%D5%A5%E9%A5%C3%
A5%C8%A5%D5%A5%A1%A5%A4%A5%EB&sv=JP&lk=no&frames&nh=10&qp=0" "Mozilla/4.06 [ja] (Win95; I)"
ncahost.nippon-cargo.co.jp - - [05/Apr/1999:02:52:42 -0400] "GET /japan/css/2level.css HTTP/1.0"
200 1231 "-" "Mozilla/4.06 [ja] (Win95; I)"
e450-1.ucg.com - - [05/Apr/1999:02:52:56 -0400] "GET /software/tutorials/gis/under20.htm
HTTP/1.0" 200 1582 "-" "AltaVista Intranet U2.0 www.search400.com webmaster@www.search400.com"
e450-1.ucg.com - - [05/Apr/1999:02:52:57 -0400] "GET /software/tutorials/gis/wexp12.htm HTTP/1.0"
200 1844 "-" "AltaVista Intranet U2.0 www.search400.com webmaster@www.search400.com"
e450-1.ucg.com - - [05/Apr/1999:02:52:58 -0400] "GET
/offices/asiapacific/korea/faq/install195/miner6.htm HTTP/1.0" 200 1075 "-" "AltaVista Intranet
U2.0 www.search400.com webmaster@www.search400.com"
e450-1.ucg.com - - [05/Apr/1999:02:52:58 -0400] "GET
/offices/asiapacific/korea/faq/install195/er6.htm HTTP/1.0" 200 911 "-" "AltaVista Intranet U2.0
www.search400.com webmaster@www.search400.com"
207.1.255.34 - - [05/Apr/1999:02:53:00 -0400] "GET /SASHome.txt.html HTTP/1.0" 304 - "-"
```

The SAS e-Data ETL program can recognize and read all the major Web log formats. You can also modify the SAS Web Analytics solution to read custom Web logs (see “Overview: Custom Web Logs” on page 64). The SAS Web Analytics solution supports the following common Web log formats:

Table 1.4 Common Web Log Formats That Are Supported by the SAS Web Analytics Solution

Web Log Format	Type of Server
Common Log Format (CLF)	Apache Web servers
Extended Log Format (ELF)	Microsoft Internet Information Server (IIS) Web servers
Microsoft Proxy	Microsoft Proxy servers
Netscape/iPlanet	Netscape and iPlanet Web servers
IIS (original)	Microsoft personal Web servers

Use the SAS e-Data ETL Administrator to specify the location of your Web log file(s) (see “The Web Log Location Properties” on page 35). When you run the Daily.sas file each day, the Extract process reads any Web log files that are in the location that you have specified.

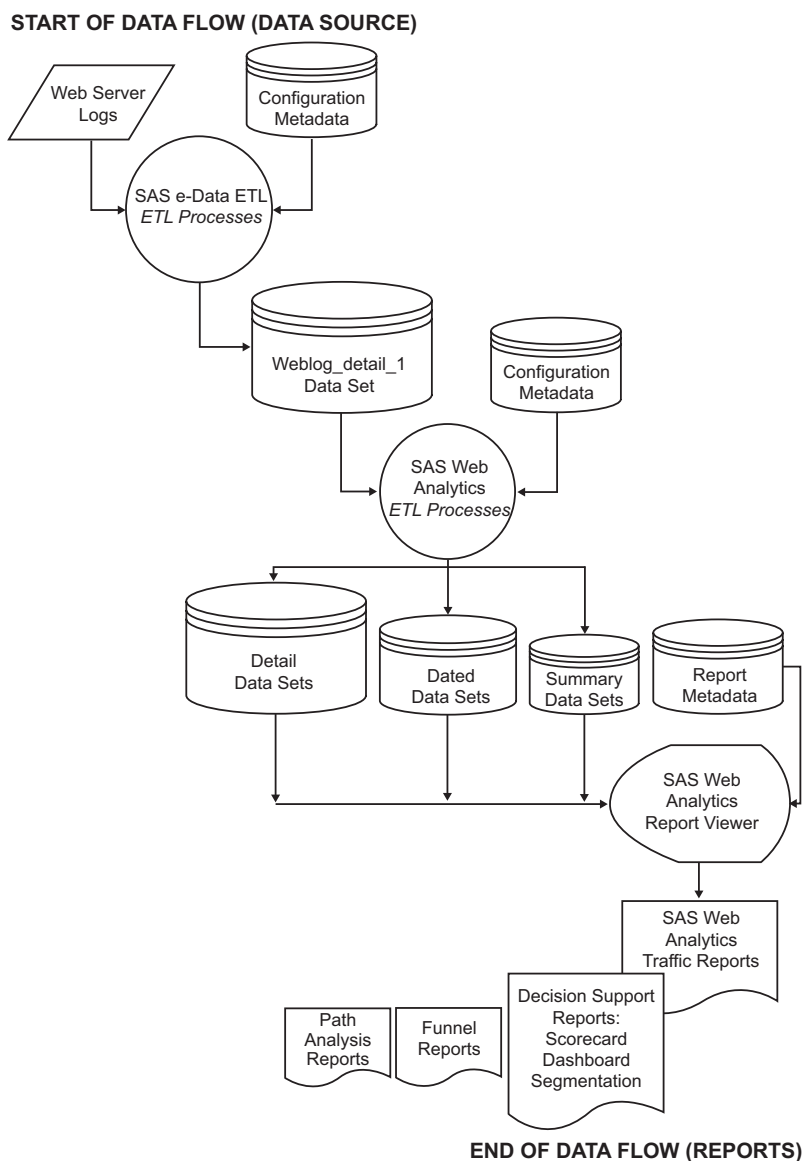
Other Data Sources

You can add a custom data source (such as a table of online product sales figures) to your Web mart if you have prepared your Web site structure and if you have customized your Web log to include an identifier that makes a link between the Web mart and the external data source. You also need to create custom SAS code that runs during the ETL processes to summarize any new variables and add them to your summarized data sets.

ETL Processes for the SAS Web Analytics Solution

The following diagram illustrates the flow of data for the SAS Web Analytics solution:

Display 1.3 Data Flow for SAS Web Analytics



The Daily.sas program invokes two macros that perform the extraction, transformation, and loading (ETL) processes—turning the raw data from Web server log files to an analysis-ready format in SAS data sets.

The Daily.sas program is generated when you create a new Web mart by using the %WAETL macro with the INITIALIZE argument for the PROGRAM parameter. The Daily.sas program runs the macros shown in the following table, which comprise the ETL portion of the SAS Web Analytics solution:

Table 1.5 Macros That the Daily.sas Program Runs

Name	Description
%EDATAETL (PROGRAM=EXTRACT)	Reads the Web server logs and creates temporary SAS data sets. The data is extracted by running multiple SAS sessions concurrently. To run only the Extract process, see “Running the Extract Process” on page 77. To customize the Extract process, see “Overview: The Modules in the Extract Process” on page 80.
%EDATAETL (PROGRAM=LOAD)	Reads the temporary data sets that are created by the Extract process. Creates detail SAS data sets, including the Weblog_Detail_1 data set. To run only the Load process, see “Running the Load Process” on page 98. To customize the Load process, see “Overview: The Modules of the Load Process” on page 99.
%WAETL (PROGRAM=WAREHOUSE)	Reads the Weblog_detail_1 data set that is created by the Load process. The WAREHOUSE argument also creates the summary data sets that are referenced in order to present SAS Web Analytics reports to the user. For more about the WAREHOUSE argument, see “Using the %WAETL Macro to Load Data into an Existing Web Mart” on page 187.

To generate the Daily.sas program, see “Using the %WAETL Macro to Load Data into an Existing Web Mart” on page 187. To run the Daily.sas program, see “Using the Daily.sas Program to Perform the ETL Processes” on page 69.

Note: After ETL processing is finished, use the SAS Web Analytics Report Viewer to view online reports. When you select a report to view in the SAS Web Analytics Report Viewer, the information about which variables to display for the selected report is then retrieved from the summary data sets and the metadata data sets in order to create the SAS Web Analytics report. △

How the Extract Process Prepares to Read the Web Logs

From a functional perspective, the Extract process performs the following tasks before it reads each Web server log file:

- identifies the available Web server log files to be read and determines the type of Web server logs to be read by consulting the Control.Wbfields data set (a data set that contains metadata).
- builds the reader jobs (SAS programs) that will read the Web server log files. The Web server log files are divided among many reader jobs. The specific number of reader jobs is determined by the value of the Number_of_Parallel_Reader_Jobs property that you specify within the SAS e-Data ETL Administrator.
- builds the INPUT statements for each reader job. These statements specify both the Web server log type and other fields that are defined in the Control.Wbfields data set.

The Extract Process: About Temporary Data Sets

The Extract process then executes the reader jobs that read the Web server log files. Each reader job groups its data into temporary data sets. As each row of the Web log is read, the reader job performs the following actions:

- filters out the non-page items, the requests from special clients, the requests from spiders and robots, and the requests that generated error status codes
- identifies cookie information if cookie information is present
- identifies any requests for executable programs such as CGI programs
- identifies the visits and the clickstream path that each visitor followed
- generates the path analysis information for the Web site

After the Web logs have been read, the Extract process executes the analysis jobs. The analysis jobs restructure the temporary data sets according to the visitor ID and the datetime stamp.

Customizing the Extract Process

The Extract process, Sashelp.Wbetl.Extract.Source, is invoked by the %EDATAETL macro within the Daily.sas program. The Extract process includes a number of modules. The modules within the Extract process that are available for customization are called custom access modules (CAMs). You can customize the following CAMs:

- User_assignments_after_input
- User_assignments_before_output
- User_assignments_by_session_id

To edit the CAMs in the Extract process, see “Modules That You Can Modify in the Extract Process” on page 76. For a list of all of the modules (both customizable and non-customizable) that are executed by the Extract process, see “Overview: The Modules in the Extract Process” on page 80.

The Load Process

The Load process, Sashelp.Wbetl.Load.Source, is also invoked by the %EDATAETL macro within the Daily.sas program. Although the Load process includes a number of modules, the only custom access module (CAM) is the User_assignments_for_data_mining CAM.

The User_assignments_for_data_mining CAM prepares your Web log data for data mining. User_assignments_for_data_mining creates a view called Detail.Web_Mine that contains all the current Web log detail data for page views, including CGI parameters and cookies. You might customize the code in User_assignments_for_data_mining for your Web site so that data mining operations in your Web site are more effective.

The User_assignments_for_data_mining CAM is automatically executed at the end of the Load process. To edit the User_assignments_for_data_mining CAM, see “User_assignments_for_data_mining” on page 100.

Customizing the Summarizing and Reporting Processes in SAS Web Analytics

You can customize the settings that control the summarizing and reporting processes as described in the following table:

Table 1.6 Settings That You Can Customize in SAS Web Analytics

Category of Information	Description
Web Analytics Configuration	Controls processing details such as the name of the default Summary data set and the number of days available in the _DAY summaries.
User Interface Settings	Controls the configuration and appearance of reports, such as the foreground color.
Summaries	Controls how to create and process the summary data sets.
Scorecards	Controls how to process a selected Scorecard.
Dashboards	Controls how to process a selected Dashboard.
Segmentations	Controls the definitions for the automatic segmentation process.

For more information about the SAS data sets that contain these settings, see “The Waadmsum Data Set” on page 193.

Output from the SAS Web Analytics Solution

The SAS Web Analytics Report Viewer

The SAS Web Analytics Report Viewer (see Display 1.4 on page 13) is a Web-based application that users can access from any supported Web browser. It enables users to view SAS Web Analytics reports from available Web mart data.

Display 1.4 The SAS Web Analytics Report Viewer

Group	Id	Summary Level	First Date	Last Date
Visitor	Unique Visitors	Day	01/01/04	05/28/04
		Week	12/28/03	05/23/04
		Month	01/01/04	05/01/04
		Quarter	01/01/04	04/01/04
		Year	01/01/04	01/01/04
Browser and Platform	Browsers	Day	01/01/04	05/28/04
		Week	12/28/03	05/23/04
		Month	01/01/04	05/01/04
		Quarter	01/01/04	04/01/04
		Year	01/01/04	01/01/04
	Browser Versions	Day	01/01/04	05/28/04
		Week	12/28/03	05/23/04
		Month	01/01/04	05/01/04
		Quarter	01/01/04	04/01/04
		Year	01/01/04	01/01/04
	Platforms	Day	01/01/04	05/28/04
		Week	12/28/03	05/23/04
		Month	01/01/04	05/01/04
		Quarter	01/01/04	04/01/04
		Year	01/01/04	01/01/04

Users can select either HTML or Java to display a report in the SAS Web Analytics Report Viewer. If they want to perform new analyses or use new data from the Web mart, then they can run the decision support reports individually (without reprocessing all of the Web log data).

For details about navigating in the SAS Web Analytics Report Viewer, click **Help** in the SAS Web Analytics Report Viewer, or see the *SAS Web Analytics: User's Guide*.

Supplied Reports

The SAS Web Analytics Report Viewer supplies a set of standard reports. You can modify existing report definitions and can create custom reports by using the SAS Web Analytics Administrator.

The standard SAS Web Analytics reports appear by category in the navigation tree at the left of the SAS Web Analytics Administrator. For detailed descriptions of each report, see the *SAS Web Analytics: User's Guide*. For more information about using the SAS Web Analytics Administrator Administrator to set up reports and report groups, see Chapter 9, “Using the SAS Web Analytics Administrator to Manage Your Web Mart(s),” on page 111.

Table 1.7 Categories of SAS Web Analytics Reports

Report Category	Function	Types of Reports
Traffic	Monitors the volume of activity and health of the Web site.	Visitor Browser and Platform Status Codes Navigation Overview Referrer
Scorecard	Determines which variables in the input data set have a statistically significant impact on the target metric.	Site Metrics
Dashboard	Displays the values for each Web site metric.	Site Metrics
Funnel	Shows how many visits viewed a specific sequence of pages.	Interactive Funnel
Path Analysis	Shows the different paths that visitors took to get from one page to another.	Entry Path Referrer Entry Paths Interactive Path Analysis
Segmentation	Creates a set of rules (segments) that help predict which visitors will return to the site.	Repeat Visitor—Totals Repeat Visitor—Averages

SAS Web Analytics reports help you perform the following tasks:

- determine which pages of your Web site are the most frequently visited
- determine which pages of your Web site that visitors find the most or the least interesting

- ❑ determine which pages of your Web site are buried too deeply for visitors to find
- ❑ determine the pages to which repeat visitors return, and determine whether they see the special offers that you are promoting in a campaign
- ❑ discover why potential purchasers abandon their transactions
- ❑ reduce business costs

Setting Up Reports

Use the SAS Web Analytics Administrator to set up default SAS Web Analytics reports and to create new reports (see “The Traffic Page” on page 123). The following sections describe the requirements for setting up the reports that you need to customize.

The Scorecard

The scorecard requires a single data set that contains a set of metric variables that are summarized by day. The default input data set is `Summary.Daily_Total_Day`. The required columns for this data set are the following variables:

- ❑ Date
- ❑ a target variable (the default target variable is visit count)
- ❑ one or more input variables

In order to produce the scorecard statistics for a given date, at least thirty continuous days of data must exist in the input data set (`Summary.Daily_Total_Day`, by default) in the time interval before that date.

The Dashboard

The dashboard requires a single data set that contains a set of metric variables that are summarized by day. The default dashboard uses the `Summary.Daily_Total_Day` data set, which contains only the daily metric data from the Web log. Additional metric variables can be added to the `Summary.Daily_Total_Day` data set or to a new daily data set that is created for the dashboard (see Chapter 12, “Dashboard Administration,” on page 163).

The dashboard is generated by code within the `%WAETL` macro. After the `%WAETL` macro summarizes all of the detail data, the default data set called `Daily_Total_Day` (located in the `Summary` library) is used as input to the dashboard. By default, the `Summary.Daily_Total_Day` data set contains values for all of the traffic-type metrics and health-type metrics for the Web site.

In order to produce the dashboard statistics for a given date, at least thirty continuous days of data must exist in the input dataset (`Summary.Daily_Total_Day`, by default) before that date.

The Segmentation Report

Segmentation analysis requires a persistent visitor ID because it tracks visitor action over a period of time. Creating customized versions of a segmentation analysis is not recommended if the visitor ID is defined by using the following methods:

- ❑ The visitor ID consists of the user agent and the IP address.
- ❑ A nonpersistent cookie is used as the visitor ID.

Path Analysis Reports

To set up an Interactive Path Analysis report, a user defines the following parameters for a specified path of interest in the SAS Web Analytics Report Viewer:

- the dates to use in the report
- a start page and/or an end page that define the limits of the specified path
- the number of paths to display
- the maximum length of the path, which can be 1 to 7 pages
- the type of visual chart to use to display the data
- the scheme to use to display the report

The Funnel Report

SAS Web Analytics software includes a stored process for the predefined funnel report. The funnel report is a predefined report that is available in the SAS Web Analytics Report Viewer. The `Stp_wafunnel.sas` stored process invokes the `%WAFUNNEL` macro. For details about how to run the stored process that generates the funnel report, see “The Stored Process for the Funnel Report” on page 175.

Getting Started with SAS Web Analytics 5.1: Administration Tasks

Administrative tasks for using the SAS Web Analytics solution to analyze a Web site fall into the following categories:

- 1 setting up a new Web mart
- 2 customizing and processing data
- 3 setting up and customizing reports

The following tables show the general administrative tasks that you might perform in order to prepare an analysis-ready Web mart.

Table 1.8 Setting Up a New Web Mart

Step	How Often	Action	Application to Use
1	Once	Initialize an empty Web mart by calling the <code>%WAETL</code> macro with the <code>INITIALIZE</code> argument for the <code>PROGRAM</code> parameter. The <code>Daily.sas</code> program is created.	SAS session
2	As needed	Edit the properties of the Extract process (such as those specifying how to handle the requests from spider or robot clients, or how to count the page requests that result in an error code) for a Web mart.	SAS e-Data ETL Administrator
3	As needed	Edit the properties of your new Web mart by customizing SAS code.	SAS session
4	Once	Register the new Web mart so that the Web mart data is recognized by the SAS Web Analytics software (and appears in the SAS Web Analytics Report Viewer).	SAS Web Analytics Administrator

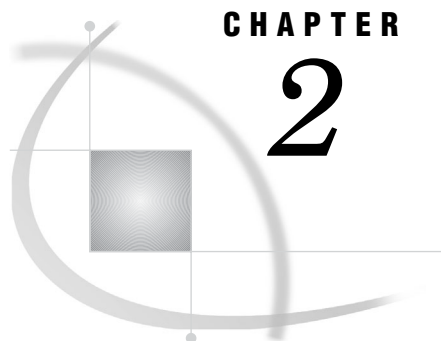
Table 1.9 Customizing and ETL Processing Tasks

Step	How Often	Action	Application to Use
5	As needed (optional)	Create and define new variables for the detail data set by editing its metadata in the Config.Waconfig data set. Set up new variables for summarizations by editing the metadata in the Config.Waadmsum data set.	SAS Web Analytics Administrator
6	As needed	If new variables are created, then perform a summarization (create new summary data sets) by calling the %WAETL macro with the WAREHOUSE argument for the PROGRAM parameter.	SAS session
7	As needed (daily)	Copy the Web server log files from the Web server to the location that is expected by the SAS e-Data ETL Extract process.	Windows operation or a batch file
8	As needed	Delete any data sets that were created from previous executions of Daily.sas (or any of the SAS Web Analytics ETL processes).	SAS session
9	As needed (daily)	Run the Daily.sas program.	SAS session
10	As needed (daily)	Move the processed Web logs to an archive location.	Windows operation or a batch file

Table 1.10 Setting Up and Customizing Reports

Step	How Often	Action	Application to Use
11	As needed	If you have defined new variables, then edit an existing report or define a new report to add the new variables to it and, if necessary, create a new report group.	SAS Web Analytics Administrator
12	As needed	If you have defined new report groups or new reports, then verify that the new report groups and the reports appear in the SAS Web Analytics Report Viewer.	SAS Web Analytics Report Viewer
13	As desired	Define new traffic reports. ¹	SAS Web Analytics Administrator
14	As desired	If you want to create a decision support report (scorecard, dashboard, or segmentation report), then provide the required parameters and run the %WADECIDE macro.	SAS session SAS Web Analytics Administrator SAS Web Analytics Report Viewer

¹ Execution of a customized or new report depends on the availability of data for the analysis.



CHAPTER

2

Creating a New Web Mart

<i>Overview: Creating a New Web Mart</i>	19
<i>Creating a New Web Mart by Using the %WAETL Macro</i>	19
<i>The Functions of the %WAETL Macro</i>	20
<i>Creating a New Web Mart by Using the SAS e-Data ETL Administrator</i>	20
<i>Registering a New Web Mart</i>	24

Overview: Creating a New Web Mart

You can create a new Web mart by using either one of the following methods:

- ❑ Call the %WAETL macro with the INITIALIZE argument for the PROGRAM parameter in the SAS Program Editor window.
- ❑ Use the SAS e-Data ETL Administrator. If you use the SAS e-Data ETL Administrator to create a new Web mart, then you must also run the %WAETL macro with the INITIALIZE argument for the PROGRAM parameter (described in the previous method) so that the SAS Web Analytics software can locate the Web server log files and the other components of the new Web mart.
- ❑ Copy an existing Web Mart and then edit the copy (see “Copying a Web Mart in the SAS Web Analytics Administrator” on page 117).

After you initialize a new Web mart and before you process any Web log data for the Web mart for the first time, you must register the Web mart. To register a new Web mart, see “Registering a New Web Mart” on page 24.

For information about editing the properties of a Web mart, see Chapter 3, “Customizing the Properties of a Web Mart,” on page 27.

Creating a New Web Mart by Using the %WAETL Macro

To create a new Web mart, invoke the %WAETL macro by typing the following code in a SAS session:

```
%waetl(program=initialize,
        swamart=newwebmartpath
        );
```

PROGRAM	specifies the %WAETL macro function that you request. Specify the argument INITIALIZE in order to set up a new SAS Web Analytics Web mart. Although the argument must match the spelling as shown here, it is not case-sensitive.
----------------	---

SWAMART specifies the path to the root directory of the SAS Web Analytics Web mart that you are setting up. Because the %WAETL macro uses this information to create the directory that you have specified, the directory must not already exist.

For more about the %WAETL macro, see “The %WAETL Macro” on page 184.

The Functions of the %WAETL Macro

The %WAETL macro (with the INITIALIZE argument for the PROGRAM parameter) performs the following functions:

- Creates a new directory for the new Web mart from the argument that you specify for the SWAMART parameter. The name of the new directory is the name of your new Web mart.
- Creates the default set of subdirectories that must exist before it can load data into the new Web mart.

Note: If you want a customized directory structure for your Web mart, then you must manually create it. △

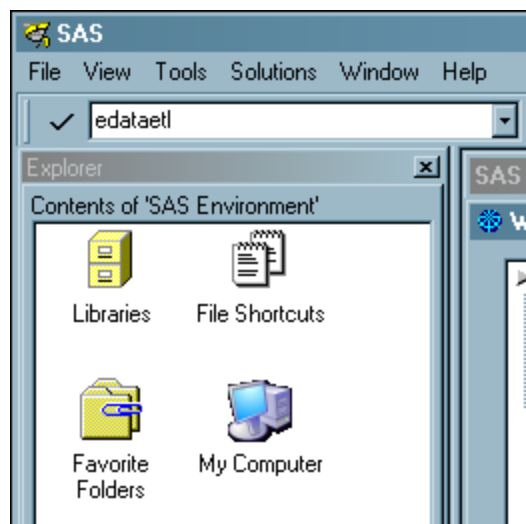
- Creates the metadata that will control the standard operations in the Web mart.
- Creates the Daily.sas program (and other programs) that will help you use the Web mart in the future. The Daily.sas program, which is created in the **sas** directory, is a program that you can use for all subsequent invocations of the %WAETL macro for the Web mart that you have just created.

Creating a New Web Mart by Using the SAS e-Data ETL Administrator

Follow these steps to create a new Web mart by using the SAS e-Data ETL Administrator:

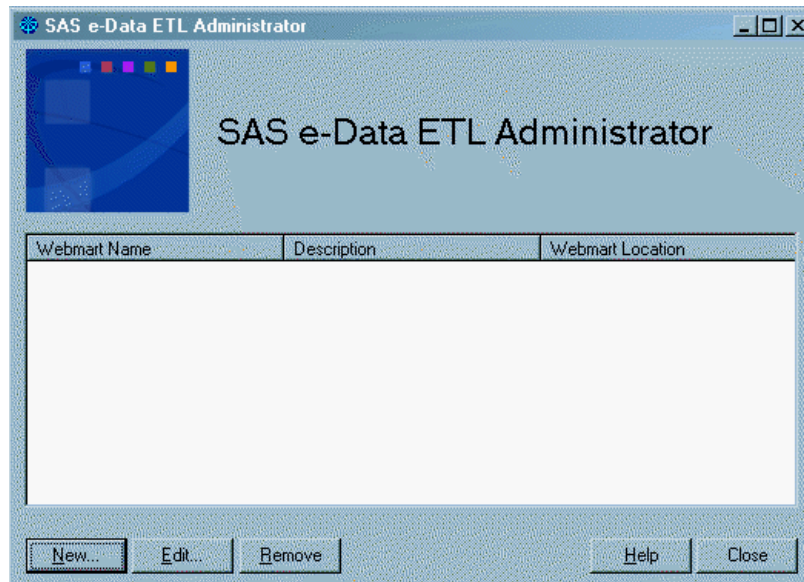
- 1 To open the SAS e-Data ETL Administrator main window, type **edataetl** on the SAS command line and press ENTER.

Display 2.1 SAS Command Line



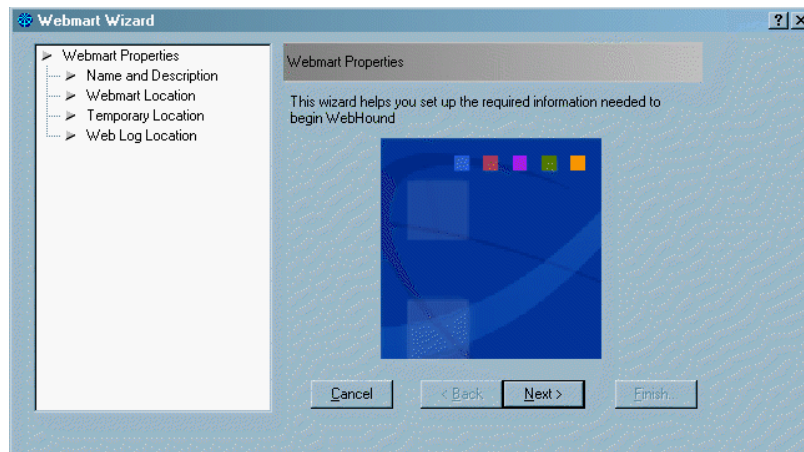
The SAS e-Data ETL Administrator main window appears.

Display 2.2 SAS e-Data ETL Administrator Main Window with No Web Marts Defined

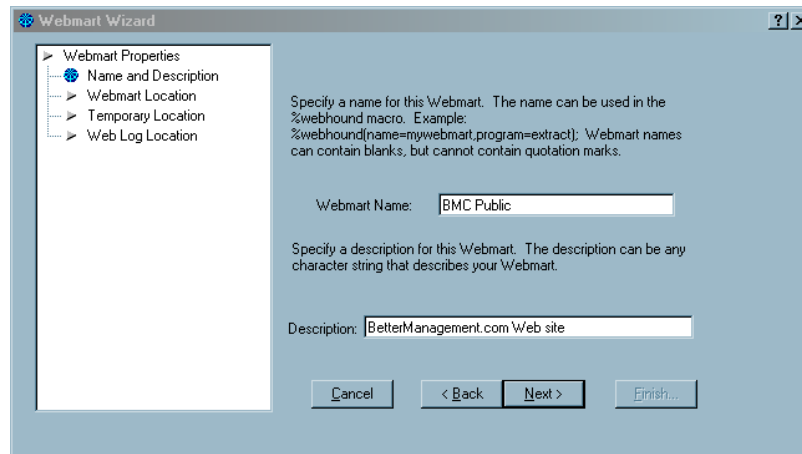


- 2 To define a new Web mart, select **New** from the bottom of the SAS e-Data ETL Administrator main window. The Webmart Wizard window opens.

Display 2.3 Introductory Window of the SAS e-Data ETL Webmart Wizard



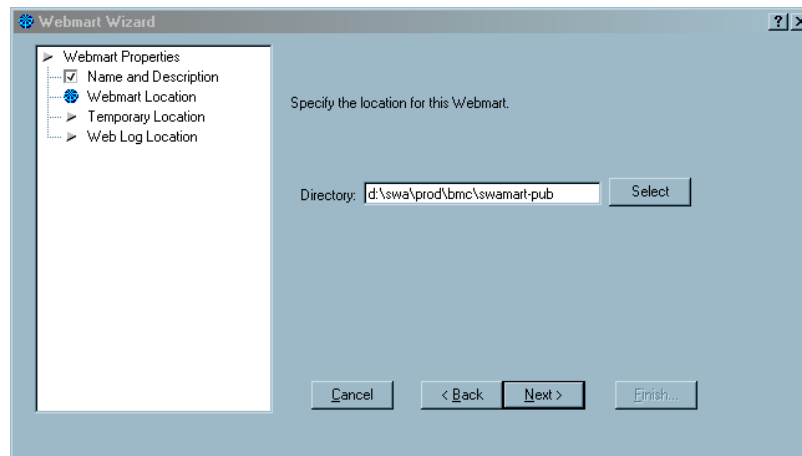
- 3 From the Webmart Wizard window, select **Next**.
- 4 For the **Name and Description** property, enter a name and a description for the Web mart that you are building. Select **Next** to continue defining the Web mart.

Display 2.4 Webmart Wizard: Name and Description Step


The screenshot shows the 'Webmart Wizard' dialog box. On the left, a tree view under 'Webmart Properties' has 'Name and Description' selected. The main area contains instructions: 'Specify a name for this Webmart. The name can be used in the %webhound macro. Example: %webhound(name=mywebmart,program=extract); Webmart names can contain blanks, but cannot contain quotation marks.' Below this is a text field for 'Webmart Name' containing 'BMC Public'. Another instruction follows: 'Specify a description for this Webmart. The description can be any character string that describes your Webmart.' Below this is a text field for 'Description' containing 'BetterManagement.com Web site'. At the bottom are buttons for 'Cancel', '< Back', 'Next >', and 'Finish...'.

- 5 For the **Webmart Location** property, type the pathname of the directory that will contain the Web mart, or use the **Select** button to select an existing directory. An example directory path is **d:\swa\prod\bmc\swamart-pub**.

Note: You cannot change this directory path after exiting the Webmart Wizard. △

Display 2.5 Webmart Wizard: Webmart Location Step


The screenshot shows the 'Webmart Wizard' dialog box at the 'Webmart Location' step. The tree view on the left now has 'Webmart Location' selected. The main area says 'Specify the location for this Webmart.' Below is a text field for 'Directory:' containing 'd:\swa\prod\bmc\swamart-pub' and a 'Select' button. At the bottom are buttons for 'Cancel', '< Back', 'Next >', and 'Finish...'.

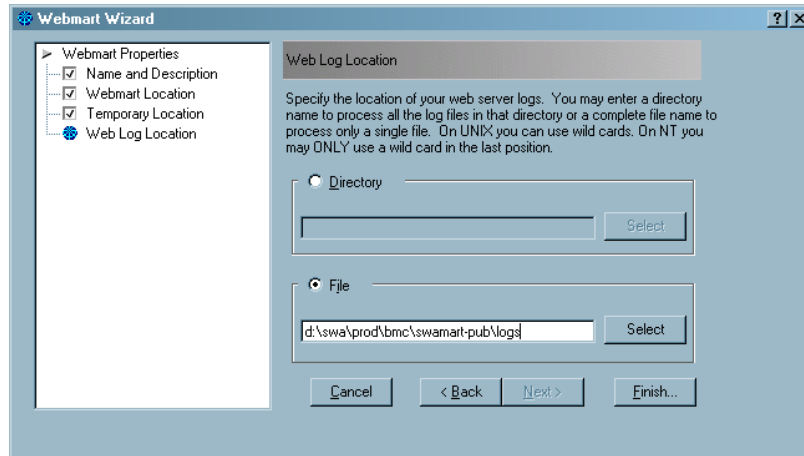
After choosing the directory for your Web mart, select **Next**.

- 6 For the **Temporary Location** property, the SAS e-Data ETL Administrator provides a default location for the temporary files. However, you can change this location by entering a new directory path or by using the **Select** button to select an existing directory. Be sure that the temporary file location provides enough disk space for the intermediate tables that will be created during SAS e-Data ETL processing.

After accepting the default location or entering the directory path for your temporary files, select **Next**.

- 7 For the **Web Log Location** property, type the location of the Web logs to be processed. The SAS e-Data ETL program can process a single file or all the files within a specified directory. To process all the files in a directory, select the **Directory** option. To process a specific file in the directory, select the **File** option. For either option, enter a directory (or a file) either by typing its pathname or by using the **Select** button to select the directory.

Display 2.6 Webmart Wizard: Web Log Location Step

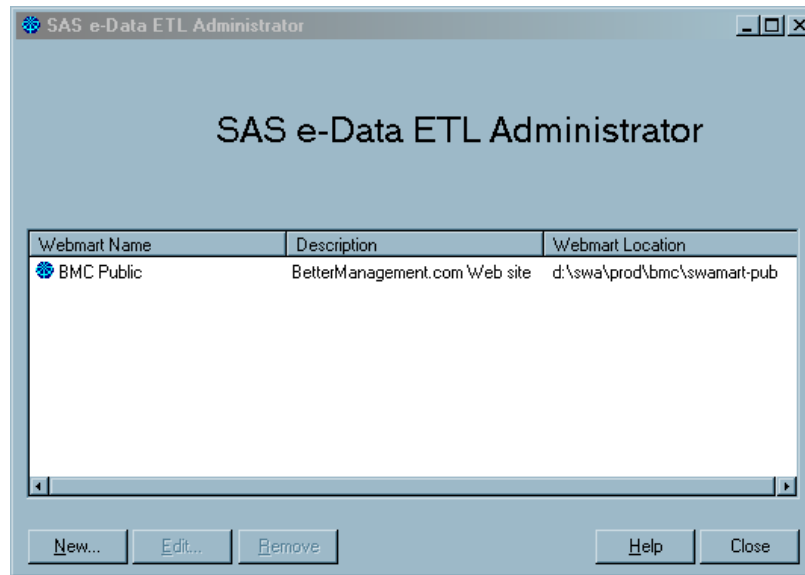


Enter the Web log location by either typing its path or by using **Select**. You can use the wildcard character (*) when specifying a location for Web log files in a UNIX or Windows operating environment. However, the asterisk is valid only in the last part of the directory or filename in the Windows environment.

Note: To process a single Web log file, select the **File** option. △

- 8 Select **Finish** to complete the creation of a new Web mart. SAS e-Data ETL software creates the directories that are required for the Web mart. Click **OK** to close the window and exit the Webmart Wizard.

The newly defined Web mart now appears in the SAS e-Data ETL Administrator main window.

Display 2.7 SAS e-Data ETL Administrator Main Window Displaying a Newly Defined Web Mart

Note: If you create a new Web mart using the SAS e-Data Administrator, then you must also initialize the new Web mart. See “Creating a New Web Mart by Using the %WAETL Macro” on page 19 for specific instructions about initializing a new Web mart.




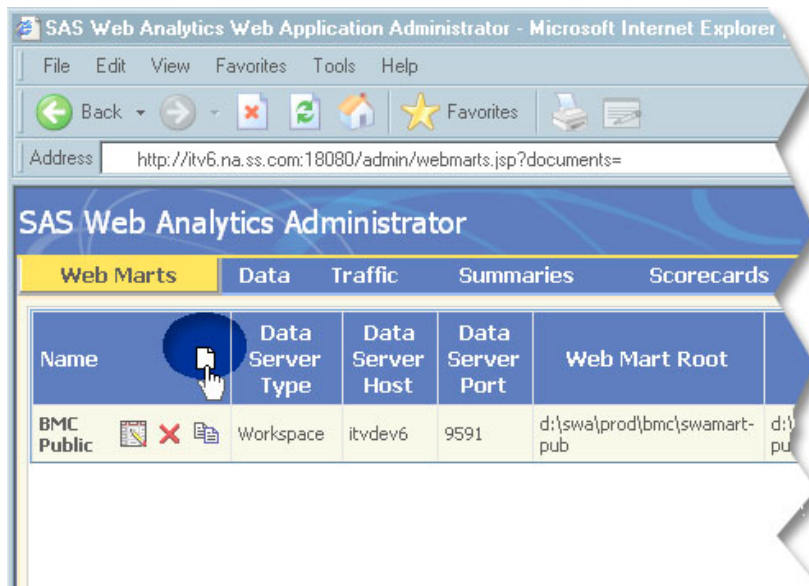
Registering a New Web Mart

After you initialize a new Web mart and before you process any Web log data for the Web mart for the first time, you must register the Web mart. When you register a Web mart, the Web mart is recognized by the SAS Web Analytics software and is listed in the SAS Web Analytics Report Viewer.

Use the SAS Web Analytics Administrator to register a new Web mart by following these steps:

- 1 Open the SAS Web Analytics Administrator in your browser by selecting the URL that was established during setup.
- 2 Select the **Web Marts** link at the top of the page.

- 3 Click  (see the following display) in the first row of the table that appears.

Display 2.8 Registering a New Web Mart

The Web Mart Form page appears.

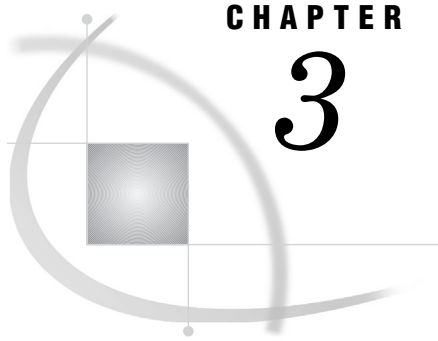
Display 2.9 The Web Mart Form for Registering a New Web Mart

The screenshot shows the "Web Mart Form" page in the SAS Web Analytics Administrator. The page has a header with "SAS Web Analytics Administrator" and "Web Analytics by SAS". Below the header is a navigation bar with tabs: "Web Marts", "Data", "Traffic", "Summaries", "Scorecards", "Dashboards", "Segmentations", and "Help SAS". The "Web Marts" tab is selected. The form is titled "Web Mart Form" and contains the following fields:

- Name: BMC PUB
- Data Server Type: Workspace (dropdown menu)
- Data Server Host: itvdev6
- Data Server Port: 8591
- Web Mart Root: D:\swa\dev\bmc\swamart-pub
- Reports Path: D:\swa\dev\bmc\swamart-pub\data\documents
- User: sasguest
- Password: (masked with dots)
- Stored Process Host: itvdev6
- Stored Process Port: 8601
- Stored Process User: sasguest
- Stored Process Password: (masked with dots)

At the bottom of the form are two buttons: "Submit" and "Cancel".

- 4 Consult with the authorities in your organization who are familiar with the configuration of the SAS solution on your server(s) to provide the information in the Web Mart Form page.
- 5 Click **Submit** to register the new Web mart.



CHAPTER

3

Customizing the Properties of a Web Mart

<i>Overview: The Properties of a SAS Web Analytics Web Mart</i>	28
<i>Saving Your Changes</i>	29
<i>Canceling Your Changes</i>	29
<i>The Properties Window of the SAS e-Data ETL Administrator</i>	30
<i>Selecting a Web Mart</i>	30
<i>Customizing a Single Property</i>	31
<i>Customizing Multiple Properties</i>	31
<i>The Detail Tables Properties</i>	31
<i>Navigating to the Detail Tables Frame</i>	32
<i>Data Set Options for the Detail Tables</i>	32
<i>Descriptions of the Detail Tables</i>	32
<i>Calculating the Number of History Tables to Keep</i>	34
<i>The Temporary Location Properties</i>	34
<i>Navigating to the Temporary Location Frame</i>	34
<i>Changing the Path for Temporary Files</i>	35
<i>Deleting Temporary Files</i>	35
<i>The Web Log Location Properties</i>	35
<i>Navigating to the Web Log Location Frame</i>	35
<i>Setting the Web Log Location</i>	35
<i>The Processing Properties</i>	35
<i>Navigating to the Processing Frame</i>	36
<i>Session Timeout</i>	36
<i>Definition of a Session</i>	36
<i>How Sessions Are Split</i>	36
<i>Preserving the Continuity of One Session across Multiple Servers</i>	37
<i>SAS Session Options</i>	37
<i>INFILE Options for Various Web Log Types</i>	37
<i>Changing the INFILE Option in the Control.Wbinfile Table</i>	37
<i>Adding or Deleting Fields in Your Web Server Log File</i>	38
<i>About DNS Lookup</i>	40
<i>Using the DNS Lookup Script</i>	41
<i>The Parsing Properties</i>	41
<i>Navigating to the Parsing Frame</i>	41
<i>Starting Level of URL Parsing</i>	42
<i>Number of URL Levels to Parse</i>	42
<i>CGI Delimiter</i>	42
<i>Cookie Delimiter</i>	42
<i>URL Prefix</i>	42
<i>CGI Program Locations</i>	43
<i>Identifying Executable URLs</i>	43
<i>Interpretation Values of the Executable URLs</i>	43

Identifying Web Browsers	43
Matching the Web Log Contents with Control.Wbbrowsr	44
Identifying Platforms	44
Matching the Web Log Contents with Control.Wbpltfrm	45
The Filtering Properties	45
Navigating to the Filtering Frame	46
Keep, Skip, Tally, and Force Non-Page View Actions	46
Special Client List	47
Editing the Contents of the Special Client List Table	47
Using the Wildcard Character in the Special Client List Table	47
Spiders	47
Editing the Contents of the Control.Wbspider Table	48
Identifying a New Spider	48
Non-Pages	48
How to Count or Exclude a MIME Type as a Page View	49
Bad Status Codes	49
The Compressed Files Properties	50
Navigating to the Compressed Files Frame	50
Processing Compressed Web Server Log Files	50
The Execution Tuning Properties	51
Navigating to the Execution Tuning Frame	51
Description of Number of Parallel Jobs Properties	51
Number of Restructure Buckets	52
Maximum Number of Open Files	52
The Advanced Customizations Frame	52
Navigating to the Advanced Customizations Frame	53
About Custom Access Modules (CAMs)	53
How to Modify a CAM	55
Deleting a CAM from the Work Library	59
How the Customizing Process Works in SAS e-Data ETL Software	61
The Control Library for Your Web Mart	61
Description of Specific Control Tables	61
Usermods Library	62

Overview: The Properties of a SAS Web Analytics Web Mart

To customize the properties of a Web mart (such as the location of the Web log(s) and the amount of data to store), use the SAS e-Data ETL Administrator. The SAS e-Data ETL Administrator is the Windows interface of the SAS e-Data ETL software.

After you select a Web mart and click **Edit** in the SAS e-Data ETL Administrator main window, the Properties window for the selected Web mart opens. In the Properties window, select any property from the tree view. Edit the components in the property frame that opens to the right of the tree view.

The following table briefly describes each of the properties of a Web mart that you can edit through its corresponding property frame in the Properties window. For detailed information about editing a property, see the corresponding subsection within this section.

Table 3.1 Property Frames of the Properties Window

Property Frame	Description
Webmart Location	Enables you to view the parent directory for the Web mart. This path is the location that you specified in the Webmart Wizard. You cannot change this directory path.
Detail Tables	Enables you to view and edit the number of Weblog_Detail, CGI_Parms, ReferrerParms, Pathing, and Cookie tables that are created and the amount of historical data that is kept for each table.
Temporary Location	Enables you to provide the location for the temporary files that are created and used during processing.
Web Log Location	Enables you to specify the location of your Web logs.
Processing	Enables you to specify what information to extract from your Web logs, define the length of a session, indicate which INFILE statement options to use with various Web log types, list the SAS session options to use during processing, and activate or deactivate DNS lookup.
Parsing	Enables you to specify how URLs are parsed, add new CGI program locations, and add new browsers or platforms that should be recognized.
Filtering	Enables you to specify what information from the Web log (such as special clients, spiders and robots, non-page requests, and bad status codes) should be included or excluded from processing.
Compressed Files	Enables you to identify whether your Web server logs files are compressed and to specify where the uncompressing executable file is located.
Execution Tuning	Enables you to modify processing parameters that affect performance.
Advanced Customizations	Enables you to copy the CAM source entries in the catalogs for SAS e-Data ETL processing (for example, the Wbetl, Wbmacros, Wbreptex, and Wbrephlp catalogs) to the Usermods library for editing.

Although the default values will work in many situations, you can use the Properties window to customize most of the properties of a Web mart for your specific needs.

Saving Your Changes

Select **OK** at any point in the process to save all changes that you have made during your editing session in the Web mart Properties window. After saving these changes, you return to the SAS e-Data ETL Administrator main window.

Canceling Your Changes

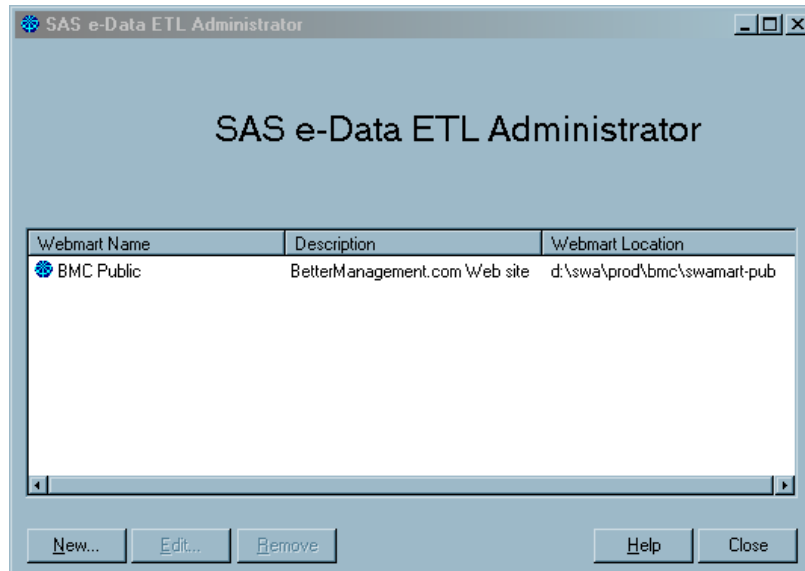
Select **Cancel** at any point to cancel all changes that you have made during your editing session in the Web mart Properties window. After canceling these changes, you return to the SAS e-Data ETL Administrator main window.

The Properties Window of the SAS e-Data ETL Administrator

Selecting a Web Mart

To select a Web mart, begin from the SAS e-Data ETL Administrator main window.

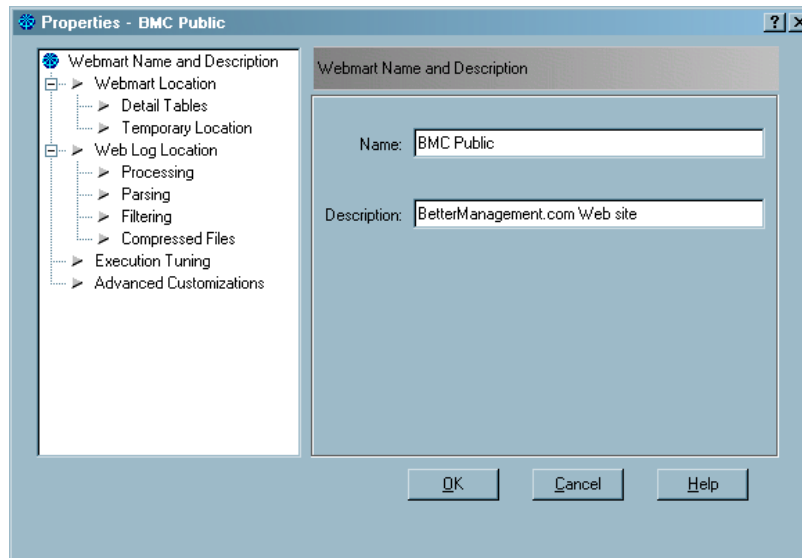
Display 3.1 SAS e-Data ETL Administrator Main Window Displaying a Web Mart Named BMC-Public



In the Webmart Name column, highlight the name of the Web mart whose properties you want to customize, and click **Edit**.

Note: If you do not see one or more Web marts listed in the SAS e-Data ETL Administrator main window, then you have not set up a Web mart. To set up a new Web mart, click **New** and follow the instructions (see “Creating a New Web Mart by Using the SAS e-Data ETL Administrator” on page 20). △

The Properties - <Web mart name> window opens. The properties of the Web mart are displayed in the tree view. The name and description of the selected Web mart are displayed in the frame on the right.

Display 3.2 Properties Window of the BMC Public Web Mart

Customizing a Single Property

To customize a single property of a Web mart, click a property from the tree view. When the property frame opens, make the modifications that are needed. If this is the last property that you want to customize, then select **OK**.

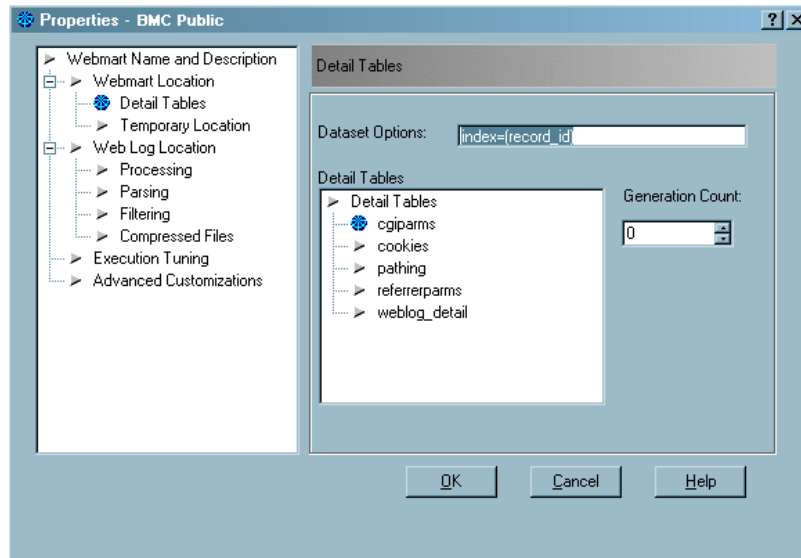
Customizing Multiple Properties

You can customize multiple properties of a Web mart. When you finish configuring a property, instead of selecting **OK**, select the next property to be customized from the tree view. When you finish making all your changes to the properties, select **OK**. The Web mart properties are saved all at once, and you return to the SAS e-Data ETL Administrator main window.

The Detail Tables Properties

The number of detail tables that are created during processing and the amount of historical data to be retained are important properties of a Web mart. These properties are specified in the detail tables (which are also called SAS detail data sets). The Detail Tables frame contains the following fields:

- ☐ a list of the detail table names
- ☐ **Generation Count**, which specifies the number of generations to keep for the highlighted detail table
- ☐ **Dataset Options** (not a required field), which enables you to specify options such as an index

Display 3.3 The Detail Tables Frame of the Properties Window

Navigating to the Detail Tables Frame

From the tree view on the left side of the Properties - <Web mart name> window, select **Detail Tables** under **Webmart Location**. The Detail Tables frame appears to the right of the tree view.

Data Set Options for the Detail Tables

Any data set options that the user enters are applied to the detail tables.

To increase efficiency, you can join one detail table to another detail table by using the **Dataset Options** field to set a common index, such as Record_ID, for each of the tables. To index a table by the variable Record_ID, type the following code in the **Dataset Options** field of the Detail Tables frame for that detail table:

```
index=(record_id)
```

Descriptions of the Detail Tables

The following table gives general descriptions of each of the six detail tables that are created by the SAS e-Data ETL application. The significance and default values of the modifiable fields in each table are also listed.

Table 3.2 Descriptions of the Detail Tables

Table Name	Description	Defaults for Key Fields
CGIParms	The CGIParms table contains name-value pairs that have been parsed from the query parameters in the original record of the Web server log file. The CGIParms table also contains a Record_ID field that you can use to match this information with its source URL.	The default value, 0, for Generation Count indicates that SAS e-Data ETL software will not keep any generations of CGIParms tables. The supplied default entry for Dataset Options enables the software to index this table by Record_ID.
Cookies	The Cookies table contains name-value pairs for the information that has been parsed from the Cookies field in the Web log. It also contains a Record_ID field to match this information in the CGIParms and Weblog_Detail tables.	The default value, 0, for Generation Count indicates that the SAS e-Data ETL software will not keep any tables. The supplied default entry for Dataset Options enables the software to index this table by Record_ID.
Pathing	The Pathing table contains one entry for each session that is held during a particular time period. The record for the entry contains the Session_ID and the duration of the session. Its datetime value is picked up from the date/time field of the first Web log record in the path. The Clickpath field contains a series of numbers that can be translated back to respective URLs by using the Unique_URLs table.	The default value, 3, for Generation Count indicates that SAS e-Data ETL software will keep three generations of the Pathing table after each run of the Load process.
ReferrerParms	The ReferrerParms table contains name-value pairs that have been parsed from the Referrer field in the original Web server log record. The ReferrerParms table provides you with a list of the keywords that are used by visitors at referring search engine sites. This table also contains a Record_ID field that you can use to match this information with its source URL.	The default value, 3, for Generation Count indicates that SAS e-Data ETL software will keep three generations of the ReferrerParms table after each run of the Load process. The supplied default entry for Dataset Options enables SAS e-Data ETL software to index this detail table by Record_ID.

Table Name	Description	Defaults for Key Fields
Weblog_Detail	<p>The Weblog_Detail table contains the same information as the input Web server data, but the information is contained in SAS tables and is sorted according to the session identifier. A Weblog_Detail table is generated every time you run the SAS e-Data ETL Load process.</p> <p>The Weblog_Detail table also contains a Record_ID field that can be used to match this information with information in the other detail tables.</p>	The default value, 3, for Generation Count indicates that SAS e-Data ETL software will keep three generations of the Weblog_Detail table after each run of the Load process.
Unique_URLs	The Unique_URLs table contains all the URLs that are referenced in the retained Pathing tables. A Unique_URLs table is generated every time you run the SAS e-Data ETL Load process.	<p>The default value is the URL for all page views minus any parameters. For example, the default value for http://www.orionstar.com/header.asp?loc=man=5GDKLJQLG6S92GFEO0AKHBAXFJSP3UVE is http://www.orionstar.com.</p>

Calculating the Number of History Tables to Keep

For the **Generation Count** field, type the number of generations of detail tables that you want to keep. Use the following formula to determine the number of generations to keep:

Days of detail you want to keep * Number of times you run the Load process per day

This formula applies to all of the detail tables.

Example calculation: If the Load process is being run once a day, and if you want nine days of Pathing data for reporting and analysis, then the **Generation Count** is 9.

Note: Consider the amount of disk space that is available when you select the number of generations of detail tables to keep. Depending on the size of your Web log, these detail tables might require a large amount of disk space. △

The Temporary Location Properties

The Temporary Location frame lists the directory path for the temporary files that are created during processing. When you initially open this window, the location that you see is the path that is provided in the Webmart Wizard.

Navigating to the Temporary Location Frame

From the tree view in the Properties - <Web mart name> window, select **Temporary Location**. The Temporary Location frame appears to the right of the tree view.

Changing the Path for Temporary Files

You can change the path for temporary files by entering a new directory path or by using the **Select** button to select an existing directory. Be sure that the temporary location provides enough disk space for the intermediate tables that will be created during processing.

Deleting Temporary Files

To remove the temporary files that are created during the Extract and Load processes, select the **Delete temporary files after loading web log files** check box. If you do not select this check box, then the files in this temporary directory are overwritten.

The Web Log Location Properties

In the Web Log Location frame, you specify the location of the Web logs to be processed.

You can use the wildcard character (*) when specifying a location for Web log files in a UNIX or Windows operating environment. However, the asterisk is valid only in the last part of the directory or filename in the Windows environment.

Navigating to the Web Log Location Frame

Select **Web Log Location** from the tree view in the Properties - <Web mart name> window. The Web Log Location frame appears to the right of the tree view.

Setting the Web Log Location

SAS e-Data ETL software can process either a single file or all the files that are contained within a specified directory. To process all the files in a directory, select the **Directory** option. To process a specific file in the directory, select the **File** option. For either option, enter the directory or filename either by typing its pathname or by using the **Select** button to select the directory.

CAUTION:

SAS e-Data ETL software processes all files in the directory that you specify for Web Log Location, whether or not they are Web server log files. Therefore, be sure that Web server log files are the only files in the Web Log Location directory. △

Note: The **File** option is a good choice when you set up a Web mart for the first time because it enables you to test a small subset of your Web log data. △

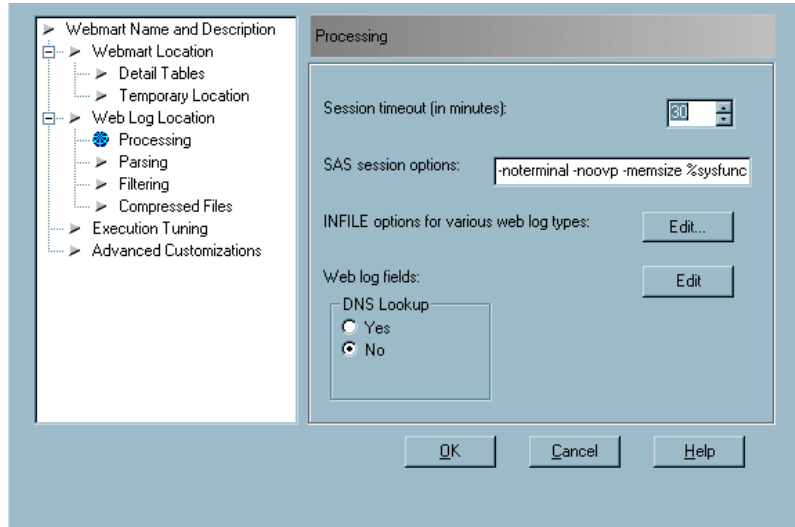
The Processing Properties

In the Processing frame, you can specify the following properties:

- the length of a SAS e-Data ETL session's timeout limit (in minutes)

- the SAS system options that are used during processing
- the INFILE options that are used for various types of Web server log files
- the fields or variables to extract from the Web server log files
- the option of converting a numeric IP address to the equivalent domain name (DNS lookup) during the Extract process

Display 3.4 The Processing Frame of the Properties Window



Navigating to the Processing Frame

From the tree view in the Properties - <Web mart name> window, select **Processing** under **Web Log Location**. The Processing frame (shown in the preceding display) appears to the right of the tree view.

Session Timeout

Definition of a Session

A session (also called a *visit*) is the period of time that a visitor spends at a Web site. A session begins when a visitor initially requests a page on the Web site. The session ends either when the visitor exits the Web site or when the session timeout limit, shown in the **Session timeout (in minutes)** field in the Processing frame, is exceeded before the visitor makes another request. The default session timeout limit is 30 minutes. Any visitor activity that continues beyond this limit is logged as a new or unique session.

How Sessions Are Split

Web server log files are generally copied and restarted once a day. When the Web server log files are processed, it is assumed that all sessions end at the end of the time span that is covered by each Web server log file.

For example, a Web server stops writing to Tuesday's log file at 1:00 a.m. each Wednesday. If a visitor browses the site at 12:55 a.m. and continues browsing until 1:05 a.m., then the activity is processed as two sessions.

Preserving the Continuity of One Session across Multiple Servers

To be sure that SAS e-Data ETL software correctly processes a single session that is spread across multiple servers (and is thereby recorded in multiple Web server log files), process all the Web server log files in the same run of the Extract process.

SAS Session Options

This property refers to the SAS system options that are used when invoking the concurrent SAS sessions for your parallel jobs. For more information about the SAS system options that you can use, see SAS Help and Documentation.

INFILE Options for Various Web Log Types

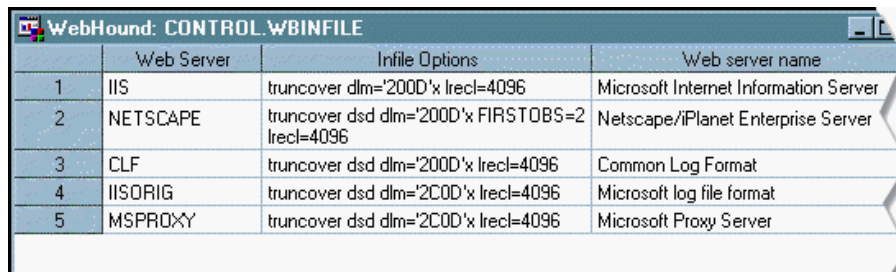
The INFILE options property enables you to specify the exact options for the SAS INFILE statement when the Web logs are read. An INFILE statement identifies an external file containing data that you want to read. It opens the file for input or, if the file is already open, makes it the current input file. This means that subsequent INPUT statements are read from this file until another file is made to be the current input file.

SAS e-Data ETL software can use one or more entries in this statement to serve as a template for reading the new Web log file type. See SAS Help and Documentation for more about the INFILE statement before making changes to the INFILE options property.

Changing the INFILE Option in the Control.Wbinfile Table

To set the INFILE options for processing Web logs, select the **Edit** button that is next to **INFILE options for various web log types** in the Processing frame. The Control.Wbinfile table opens for you to edit as shown in the following display.

Display 3.5 Control.Wbinfile Table



	Web Server	Infile Options	Web server name
1	IIS	trunccover dlm='200D'x lrecl=4096	Microsoft Internet Information Server
2	NETSCAPE	trunccover dsd dlm='200D'x FIRSTOBS=2 lrecl=4096	Netscape/iPlanet Enterprise Server
3	CLF	trunccover dsd dlm='200D'x lrecl=4096	Common Log Format
4	IISORIG	trunccover dsd dlm='2C0D'x lrecl=4096	Microsoft log file format
5	MSPROXY	trunccover dsd dlm='2C0D'x lrecl=4096	Microsoft Proxy Server

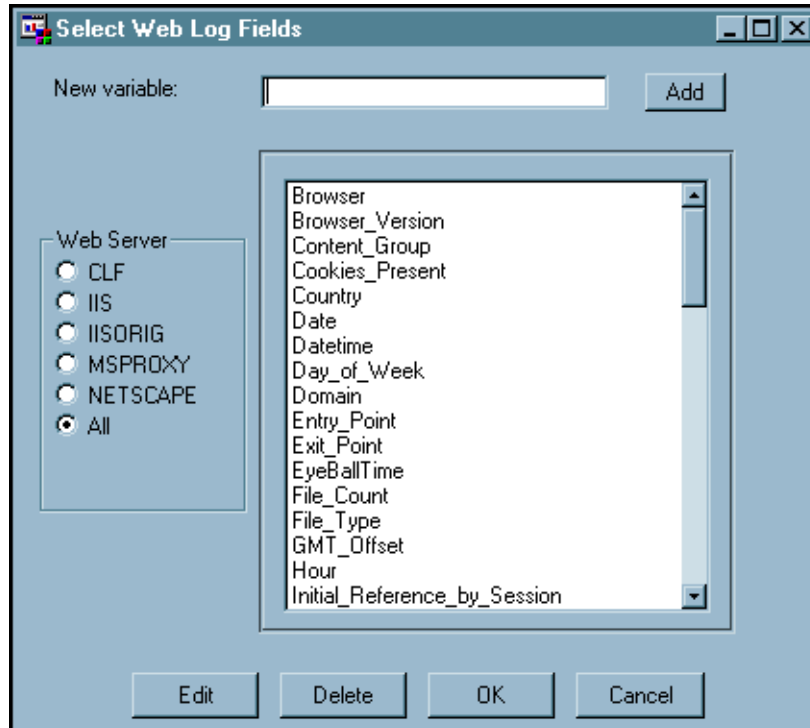
When you have finished making your modifications, close the table. In the dialog box that appears, click **Yes** to confirm any changes that you made to the table or click **No** to cancel any modifications that you made. The Processing frame re-appears.

Adding or Deleting Fields in Your Web Server Log File

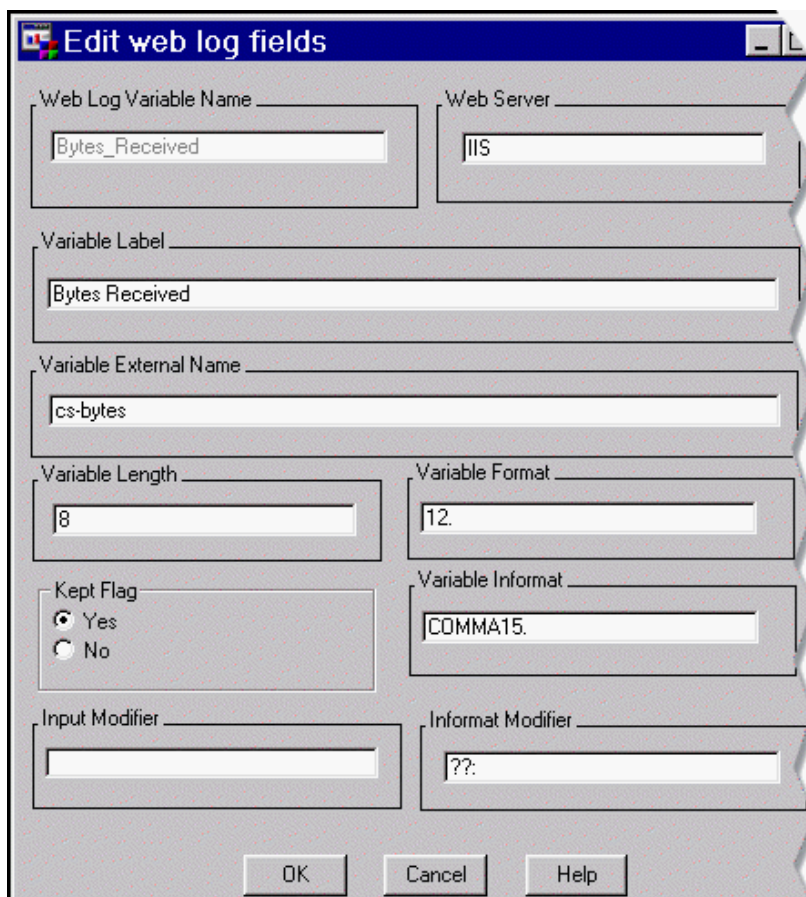
To add or delete the field variables that are extracted (by default) from your Web server log files, select the **Edit** button that is next to **Web log fields** in the Processing frame.

The major types of Web server log files that are used in the industry are listed on the left side of the Select Web Log Fields window under **Web Server**. When you select a particular type of Web server log file, its default fields are listed in the right pane.

Display 3.6 The Select Web Log Fields Window



You can edit each of the fields for your type of Web server log file. After you select the type of Web server log file, highlight the field variable that you want to edit from the list. Select **Edit** to open the Edit Web Log Fields window, as shown in the following display.

Display 3.7 The Edit Web Log Fields Window


Edit web log fields

Web Log Variable Name: Bytes_Received

Web Server: IIS

Variable Label: Bytes Received

Variable External Name: cs-bytes

Variable Length: 8

Variable Format: 12.

Kept Flag: ☒ Yes ☐ No

Variable Informat: COMMA15.

Input Modifier:

Informat Modifier: ??

OK Cancel Help

The following table lists the field variable parameters that you can edit in a Web server log file.

Table 3.3 Parameters of a Field Variable in a Web Server Log File

Parameter	Description
Web Log Variable Name	Enables you to alter the variable names in a Web log to fit the SAS syntax for variable names. Some variable names in Web server log files (those names that contain a hyphen, for example, such as cs-bytes) are not valid SAS variable names. In this case, you could use the Web Log Variable Name field to specify that the variable name cs-bytes should correspond to Bytes_Received.
Web Server	Indicates the software that handles the requests for content that are received from visitors to your Web site.
Variable Label	Indicates the text string that appears instead of the variable name in a report such as a PROC PRINT listing.
Variable External Name	Indicates the original name of the Web server log file variable.

Parameter	Description
Variable Length	Indicates the number of bytes of physical storage that each value of a variable uses in each observation of a SAS data set.
Kept Flag	Indicates whether or not this variable is stored in the data sets that are created. If you do not want to extract any data for this field variable from the Web server log file, then set Kept Flag to No .
Input Modifier	Used in an INPUT statement to position the pointer in the input buffer. You can specify a column position or a search string. The input modifier is placed before the variable name in the INPUT statement. Input modifiers can be specified for variables in the Wbfields metadata table to build the INPUT statement that is used to read a particular Web server log file. For example, the input modifier @10 places the pointer in column 10. The input modifier @ 'S-UA' places the pointer after the first occurrence of the string "S-UA."
Variable Format	Used in a FORMAT statement to control how SAS displays a variable value. For example, if you want a number to appear with commas, you can use the COMMAw.d format, where <i>w</i> represents the total field width and <i>d</i> represents how many decimal places are displayed.
Variable Informat	Used in an INPUT statement to control how SAS creates a variable's value (that is, a reading instruction). For example, to read a number that contains commas or certain other special characters, use the COMMAw. format, where <i>w</i> represents the number of columns to read.
Informat Modifier	Used in an INPUT statement in conjunction with an informat to control how SAS creates a variable's value. The informat modifier is placed between the variable name and the informat. For example, use a colon (:) to indicate that SAS should use the LIST style of input to capture a value from the input, and then use the informat to follow it as described previously to create the variable's value. Informat modifiers can be specified for variables in the Wbfields metadata table to build the INPUT statement that is used to read a particular Web server log file.

About DNS Lookup

Every Web server uses Domain Name Services (DNS) to resolve language-based domain names such as **godfrey.ibm.com** into a 32-bit numeric Internet Protocol (IP) address written as four numbers separated by periods, such as 9.50.115.89. Each of the four numbers that form a unique IP address on the Internet must be between 0 and 255. DNS uses IP addresses to identify a specific network and a host on that network.

If you activate DNS lookup, then SAS e-Data ETL software uses a process called *reverse domain name resolution* to resolve (convert) visitor IP addresses into their corresponding user-friendly domain names. A domain name can identify one or more IP

addresses. You can use domain names to determine the origin of visitors, either friendly or competitive, to your Web site.

Using the DNS Lookup Script

To avoid slower performance if you elect to resolve visitor IP addresses, use the DNS lookup script (supplied with SAS e-Data ETL software) *outside* SAS e-Data ETL processing. The DNS lookup script makes a copy of each Web server log file that contains both the IP addresses of visitors and their converted domain names. Use these copies as input to the Extract process and archive them to keep a record of the most current domain names that are associated with visitor IP addresses.

The Parsing Properties

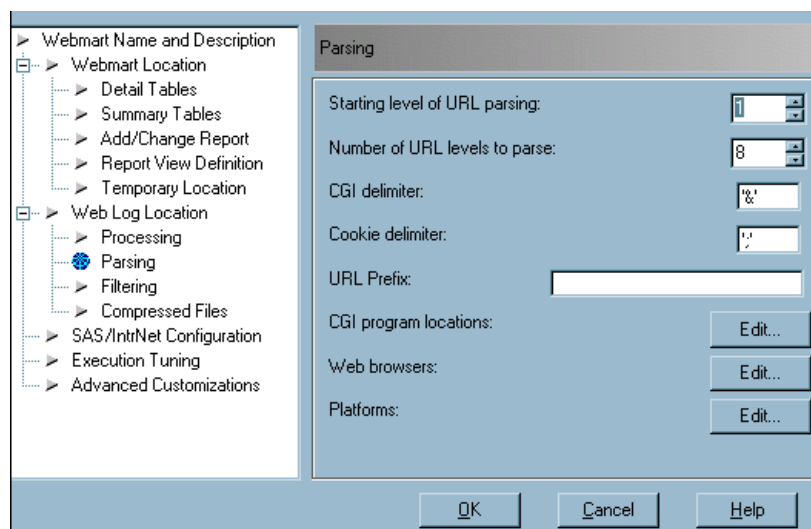
You can edit these properties for parsing your Web server log files:

- ☐ starting level for URL parsing
- ☐ number of URL levels to parse
- ☐ CGI delimiters to use for parsing
- ☐ cookie delimiters to use for parsing
- ☐ URL prefix
- ☐ location of executable programs and directories
- ☐ list of browsers that the SAS e-Data ETL application recognizes
- ☐ list of platform types that the SAS e-Data ETL application recognizes

Navigating to the Parsing Frame

From the tree view in the Properties - <Web mart name> window, select **Parsing** under **Web Log Location**. The Parsing frame (shown in the following display) appears to the right of the tree view.

Display 3.8 The Parsing Frame in the Properties Window



Starting Level of URL Parsing

The URL level equates to the logical directory structure of the Web server, which is delimited by a forward slash (/). Some Web sites might have the same values for the first few levels of all their Web pages, such as **/web/html/pages/**. In such cases, setting the starting level of URL parsing to 4 would exclude the first three common levels from processing. The default value for the starting level is 1.

Number of URL Levels to Parse

This property indicates how many directory levels to parse from a URL. For example, the URL **/web/html/pages/index.htm** has four levels. When you set the number of URL levels, remember that lengthy URLs might require additional horizontal space for display of each additional level in the output of a report. The default (and maximum) value for the number of levels to parse is 8.

CGI Delimiter

The CGI delimiter refers to the character(s) that separate the distinct name-value pairs in a URL. The ampersand (&), which is this product's default value, is most often used as the CGI delimiter. However, the international standard for URLs does allow other characters to be used as the CGI delimiter.

Note: When entering a CGI delimiter character into this field, enclose it in single quotation marks ('). △

Cookie Delimiter

The cookie delimiter refers to the character(s) that separate the distinct cookie name-value pairs when the fields that contain cookie values exist. The semicolon (;) is most often used as a delimiter. SAS e-Data ETL software uses the semicolon as its default. However, there are currently no standards for delimiters.

Note: When you type a cookie delimiter into this field, enclose it in single quotation marks ('). △

URL Prefix

The URL prefix consists of the URL scheme and authority (such as **http://www.sas.com**) that are used to prefix the requests that are logged by your Web server. Web servers that use standard Web log formats do not store the URL scheme and authority for individual requests, so you must supply this information to the SAS e-Data ETL application separately.

This value is stored in the URL_Prefix field of the Control.Wbconfig table. You can enter only one value in this field. The value that you specify in the URL_Prefix field will be prefixed to all the requests in your Web log.

In some cases, one URL prefix might not be sufficient. You might have multiple logical Web sites that are served by one Web server. As a result, you might want to prefix some requests with **http://www.site1.com** and the remaining requests with **http://www.site2.com**.

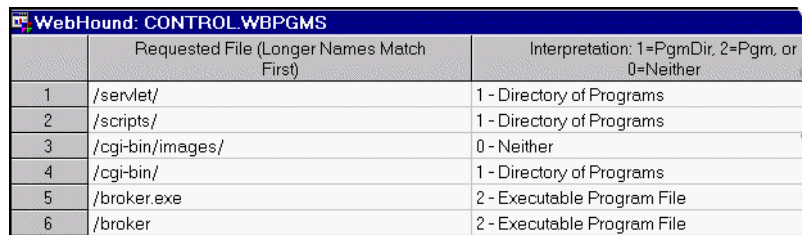
If one URL prefix will not meet your site's needs, then you must modify the code in the User_assignments_before_output custom access module (CAM) to set the value of

the URL prefix based on other information in the request. For example, your code can use the value of the Requested File field in the Web log record to determine which Web site contains the requested file. After you determine the Web site, then you can set the URL prefix.

CGI Program Locations

The Control.Wbpgms table contains the locations of executable programs and directories. Select the **Edit** button that is next to **CGI program locations** to open the Control.Wbpgms table.

Display 3.9 Control.Wbpgms Table



	Requested File (Longer Names Match First)	Interpretation: 1=PgmDir, 2=Pgm, or 0=Neither
1	/servlet/	1 - Directory of Programs
2	/scripts/	1 - Directory of Programs
3	/cgi-bin/images/	0 - Neither
4	/cgi-bin/	1 - Directory of Programs
5	/broker.exe	2 - Executable Program File
6	/broker	2 - Executable Program File

Identifying Executable URLs

The Control.Wbpgms table contains the interpretation assignments for the URLs. These assignments include the locations of executable programs. This table enables you to specify the interpretation assignment of each directory or file that is listed.

Interpretation Values of the Executable URLs

The interpretation values of the executable URLs and their meanings are listed in the following table.

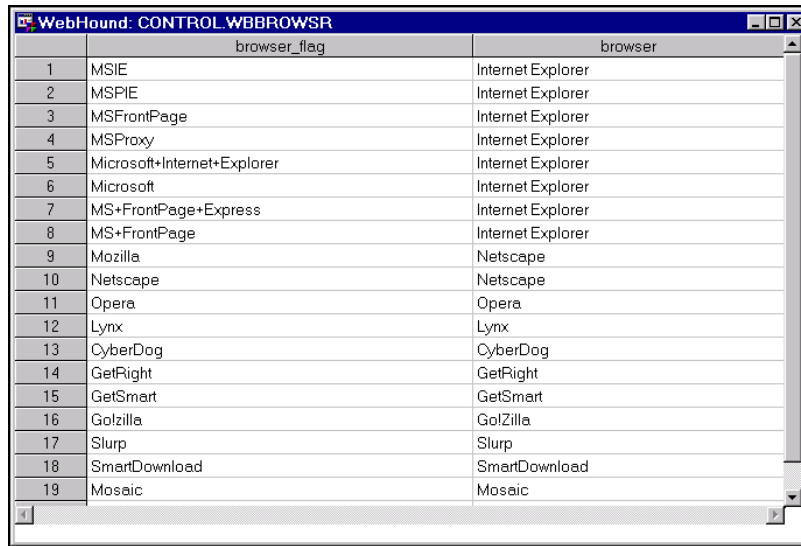
Table 3.4 Control.Wbpgms Table: Interpretation Values and Their Meanings

Interpretation Value	Meaning
1	All URLs under this directory are considered to be the locations of executable programs.
2	The specific program is considered to be an executable program.
0	The directory is parsed the same as all other URLs. Use type 0 to allow directories such as image directories to exist in a subdirectory that might otherwise contain all executable programs.

Identifying Web Browsers

The Control.Wbbrowsr table contains a list of all the browsers that are recognized by default. To open this table, select the **Edit** button that is next to **Web browsers** in the Parsing frame.

Display 3.10 Control.Wbbrowsr Table



	browser_flag	browser
1	MSIE	Internet Explorer
2	MSPIE	Internet Explorer
3	MSFrontPage	Internet Explorer
4	MSPProxy	Internet Explorer
5	Microsoft+Internet+Explorer	Internet Explorer
6	Microsoft	Internet Explorer
7	MS+FrontPage+Express	Internet Explorer
8	MS+FrontPage	Internet Explorer
9	Mozilla	Netscape
10	Netscape	Netscape
11	Opera	Opera
12	Lynx	Lynx
13	CyberDog	CyberDog
14	GetRight	GetRight
15	GetSmart	GetSmart
16	Golzilla	Golzilla
17	Slurp	Slurp
18	SmartDownload	SmartDownload
19	Mosaic	Mosaic

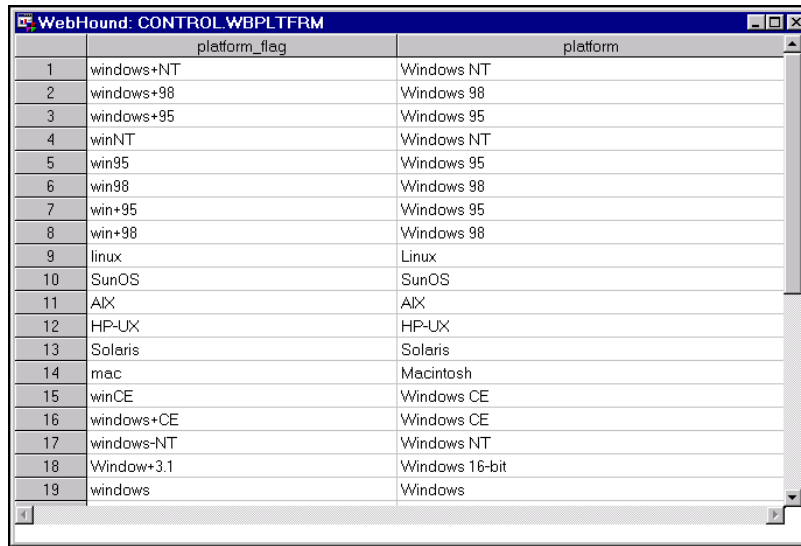
Matching the Web Log Contents with Control.Wbbrowsr

The name of the visitor's browser is captured by the User_Agent field of the Web log. During processing, the User_Agent field is compared to browser_flag strings in the Control.Wbbrowsr table.

If the User_Agent field matches one of the browser_flag strings, then the value for the corresponding browser is stored in the detail tables. You can specify additional values for browser_flag. If none of the values of browser_flag match the User_Agent field, then the value of browser is set to **other**.

Identifying Platforms

The Control.Wbpltfm table contains a list of all the operating system platforms that are recognized by default. To open this table, select the **Edit** button that is next to **Platforms** in the Parsing frame.

Display 3.11 Control.Wbpltfm Table


	platform_flag	platform
1	windows+NT	Windows NT
2	windows+98	Windows 98
3	windows+95	Windows 95
4	winNT	Windows NT
5	win95	Windows 95
6	win98	Windows 98
7	win+95	Windows 95
8	win+98	Windows 98
9	linux	Linux
10	SunOS	SunOS
11	AIX	AIX
12	HP-UX	HP-UX
13	Solaris	Solaris
14	mac	Macintosh
15	winCE	Windows CE
16	windows+CE	Windows CE
17	windows-NT	Windows NT
18	Window+3.1	Windows 16-bit
19	windows	Windows

Matching the Web Log Contents with Control.Wbpltfm

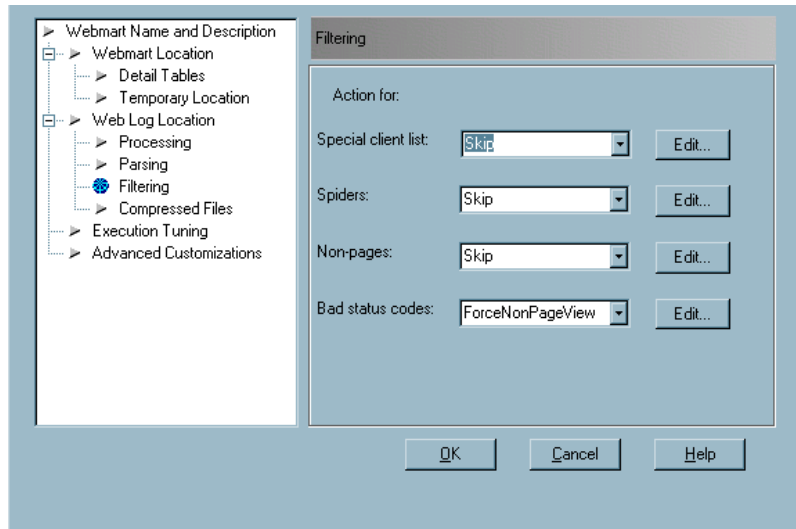
The name of the visitor's platform is captured in the User_Agent field of the Web server log file. During processing, the text in the User_Agent field is compared to platform_flag strings in the Control.Wbpltfm table.

If the User_Agent field matches one of the platform_flag strings, then the corresponding platform value is stored in the detail tables. You can specify additional values for platform_flag. If none of the values of platform_flag are present in the User_Agent field, then the value of platform is set to **other**.

The Filtering Properties

You can set properties in the Filtering frame to determine how to handle the following requests:

- ☐ requests from clients that are designated as special, such as clients whose requests should be excluded from the final results
- ☐ requests from spider or robot clients
- ☐ images or other non-text requests
- ☐ page requests that resulted in a bad status code (indicating that an error occurred)

Display 3.12 Filtering Frame of the Properties Window

Navigating to the Filtering Frame

From the tree view in the Properties - *<Web mart name>* window, select **Filtering** under **Web Log Location**. The Filtering frame appears to the right of the tree view.

Keep, Skip, Tally, and Force Non-Page View Actions

For each of the filtering properties (special client list, spiders, non-pages, and bad status codes), you can select a **Keep**, **Skip**, **Tally**, or **ForceNonPageView** action.

Keep	Includes and processes all incoming records of this type from the Web log.
Skip	Ignores all incoming records of this type during processing.
Tally	Counts the number of this type of record, but does not perform any further analysis. SAS e-Data ETL software does not include analytical data for this type of record in the Weblog_Detail table.
ForceNonPageView	Does not count these requests along with legitimate page requests, but does include data for the records in the Weblog_Detail table. This option is useful, for example, when you want to use the SAS Web Analytics software to produce a report that lists all the requests that produced error messages, but you don't want to count them with the other requests.

The following table shows how SAS e-Data ETL software processes a filtering property according to the selected action.

Table 3.5 How SAS e-Data ETL Software Processes Filtering Properties According to Action

Action	Filtering Property Is Counted as a Page	Filtering Property Is Included in the Weblog_Detail Table
Keep	✓	✓
Skip		
Tally	✓	
ForceNonPageView		✓

Special Client List

The special client list property specifies how SAS e-Data ETL software should treat records that originate from any client in the special client list. This list contains host names or IP addresses that identify clients that need to have special treatment. For example, you can use this list to exclude records that originate from within a company, so that the analysts can concentrate on Web traffic that originates from customers only. By default, this table initially contains no entries.

The default action for special clients is **Skip**. During processing, SAS e-Data ETL software ignores all the records (in the Web server log data) from the special clients in this list.

Editing the Contents of the Special Client List Table

To add an IP address to the special client list table (Control.Wbspec1), select the **Edit** button that is next to **Special client list**. The Control.Wbspec1 table opens. When you first open this table, it does not contain any entries. Add the host names or IP addresses that you want SAS e-Data ETL software to treat as special clients.

Using the Wildcard Character in the Special Client List Table

If you want all the client IDs from the same domain to be treated as special clients, then use a wildcard character (single leading or trailing asterisk) to match multiple client IDs. Here are the acceptable uses of the wildcard character to specify special clients:

Table 3.6 Examples of Using the Wildcard Character

Wildcard Usage	Effect
10.*	matches all client IDs that start with "10."
34.12.*	matches all client IDs that start with "34.12."
*.sas.com	matches all client IDs that end with ".sas.com"
*.org	matches all client IDs that end in ".org"

Spiders

Most search engine portals use a spider or robot program to search and catalog Web pages. Spiders and robots frequently access all the pages in a Web site while they search for content.

If the requests from spiders and robots were not segregated when SAS e-Data ETL software analyzes a Web server traffic pattern, then the statistics such as Most Frequent Visitors and the Entry and Exit Points would, at worst, be skewed or inflated. At best, the data that describes client activity would not generally represent the mindful selections of human clients.

The purpose of identifying the spider clients in the Control.Wbspider table is to enable the hits from spiders or robot programs (as opposed to hits from human clients) to be excluded from analysis.

As with the special client list, the default action for spider clients is **Skip**. During processing, the actions of spider and robot clients that are listed in the Control.Wbspider table are ignored.

Editing the Contents of the Control.Wbspider Table

To access the Control.Wbspider table, select the **Edit** button that is next to **Spiders**. The Control.Wbspider table displays. To add a new spider client or to delete a spider or robot client from the Control.Wbspider table, highlight a row and select **Edit** on the tool bar. Select the appropriate action from the drop-down menu.

Display 3.13 Contents of a Sample Control.Wbspider Table

WebHound: CONTROL.WBSPIDER			
	Client_ID	spider	portal
1	206.129.98.7	search.wport.com	allothers
2	206.129.98.16	search3.wport.com	allothers
3	195.20.224.73	crawlit.crawler.de	allothers
4	204.245.193.49	m49.pierian.com	allothers
5	216.200.196.14	www.ip3000.com	allothers
6	216.200.196.13	www.ip3000.com	allothers
7	216.200.196.12	www.ip3000.com	allothers
8	216.200.196.15	www.ip3000.com	allothers
9	216.200.196.11	www.ip3000.com	allothers
10	216.200.196.9	www.ip3000.com	allothers
11	216.200.196.10	www.ip3000.com	allothers
12	216.200.196.8	www.ip3000.com	allothers
13	209.67.247.156	156.128/25.247.67.209.in-addr.arpa	allothers
14	209.67.247.153	153.128/25.247.67.209.in-addr.arpa	allothers
15	204.32.117.130	sjv-ca56c-130.rasserver.net	allothers
16	209.67.247.157	157.128/25.247.67.209.in-addr.arpa	allothers
17	209.67.247.204	204.128/25.247.67.209.in-addr.arpa	allothers
18	209.67.247.203	203.128/25.247.67.209.in-addr.arpa	allothers
19	209.67.247.202	202.128/25.247.67.209.in-addr.arpa	allothers

Identifying a New Spider

Spiders or robots might make requests to every page of your Web site. If a visitor's clickstream is unusually long, then a message is written to the SAS log warning that the request probably came from a spider or robot. You might want to add this visitor to your Control.Wbspider table.

Non-Pages

An example of a non-page type of resource is an image (GIF or JPG) file or an audio (AU or WAV) file. Each requested item, whether it is a non-page GIF file or a genuine

HTML file, is called a *hit* when Web log data is being analyzed. However, by default, only HTML files, ASP programs, CGI programs, ASPX programs, JavaHTML files, PDF files, and text- or script-based Cold Fusion files are counted as page views.

Processing non-page resources provides little useful data for analysis but requires a great deal of resources. If you set non-pages to an action other than **skip**, then loading this type of data into the detail tables will greatly increase the time that it takes to process Web data and can unnecessarily increase the demand for data storage.

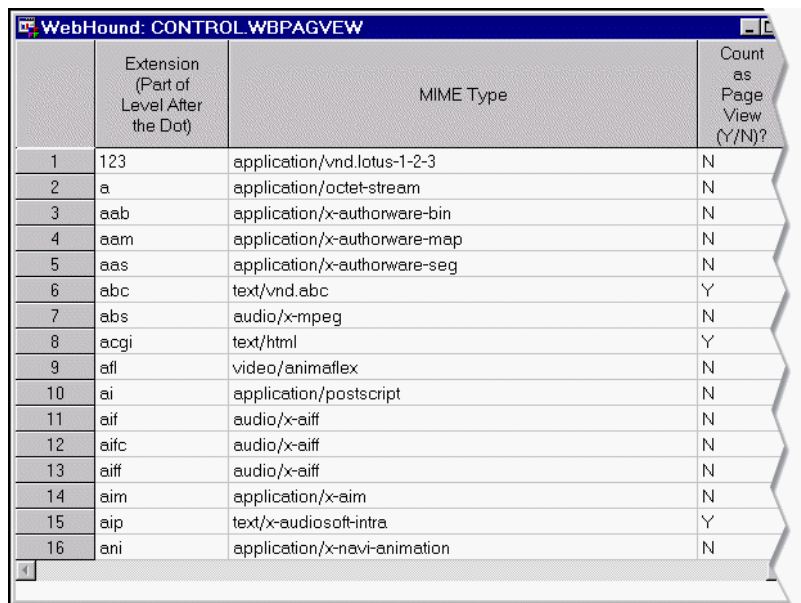
The default action **skip** instructs SAS e-Data ETL software to ignore and discard all requests for non-pages during processing.

How to Count or Exclude a MIME Type as a Page View

In addition to editing the action property for non-page items, you can specify whether to count certain types of files as pages for the page view information. The Control.Wbpagview table associates each type of file, known as a MIME type, with a file extension. You edit the Control.Wbpagview table in order to instruct SAS e-Data ETL software to exclude particular MIME types (such as JavaScript files, Perl scripts, or other types of files) as page views.

To open the Control.Wbpagview table, select the **Edit** button that is next to **Non-pages**. The Control.Wbpagview table contains all MIME types that it recognizes. By entering *Y* or *N* beside the associated MIME type, you instruct SAS e-Data ETL software to count or exclude that MIME type during processing. In the following display, for example, three MIME types—ABC, ACGI, and AIP—will be counted as page views.

Display 3.14 Control.Wbpagview Table



	Extension (Part of Level After the Dot)	MIME Type	Count as Page View (Y/N)?
1	123	application/vnd.lotus-1-2-3	N
2	a	application/octet-stream	N
3	aab	application/x-authorware-bin	N
4	aam	application/x-authorware-map	N
5	aas	application/x-authorware-seg	N
6	abc	text/vnd.abc	Y
7	abs	audio/x-mpeg	N
8	acgi	text/html	Y
9	afl	video/animaflex	N
10	ai	application/postscript	N
11	aif	audio/x-aiff	N
12	aifc	audio/x-aiff	N
13	aiff	audio/x-aiff	N
14	aim	application/x-aim	N
15	aip	text/x-audio-software-intra	Y
16	ani	application/x-navi-animation	N

Bad Status Codes

The bad status codes property enables you to customize the action that SAS e-Data ETL software takes when it encounters records that contain an error status code in the incoming data from the Web server log file. For example, status code 404 indicates that the Web server could not locate the requested resource.

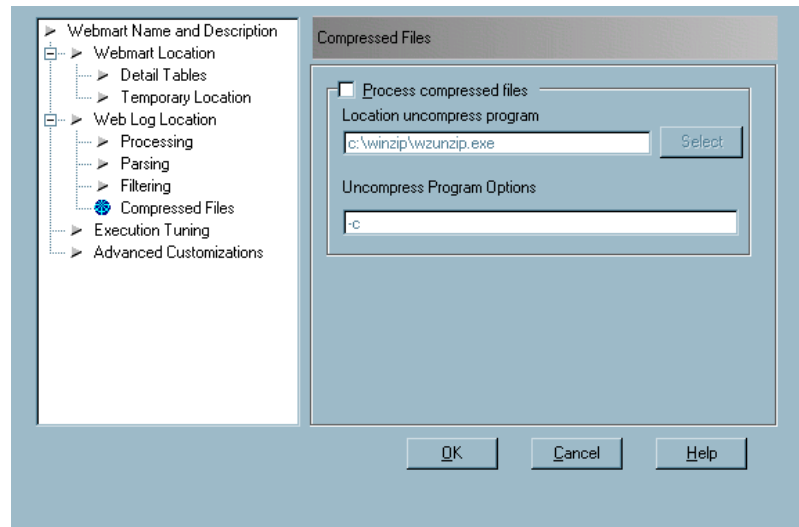
If you want your detail tables to contain data about bad status code pages, then the records must be available in your table. **ForceNonPageView**, which is the default

action, instructs the software to keep the information about the resources that produced error codes in the table, but not to count bad status codes as pages.

The Compressed Files Properties

In the Compressed Files frame of the Properties window, you specify whether your Web server log files are compressed, where the uncompressing executable file is located, and, if applicable, what uncompress command options to use.

Display 3.15 Compressed Files Frame of the Properties Window



Navigating to the Compressed Files Frame

From the tree view in the Properties - <Web mart name> window, select **Compressed Files** under **Web Log Location**. The Compressed Files frame appears to the right of the tree view.

Processing Compressed Web Server Log Files

Selecting the **Process compressed files** option instructs SAS e-Data ETL software to process Web server log files that have been compressed with either a ZIP utility or the UNIX **compress** command. If the option is selected, then SAS e-Data ETL software calls the specified uncompress program to uncompress the Web server log files during processing, but it does not physically place the uncompressed files onto the disk of the processing system.

CAUTION:

Performance metrics have shown that selecting this option results in poor performance in the Windows operating environment. For better this performance, use a separate utility to uncompress your Web server log files before processing them with SAS e-Data ETL software in the Windows environment. △

If the **Process compressed files** option is selected, then specify the name and location of the program that will be used to uncompress the Web server log files. Use

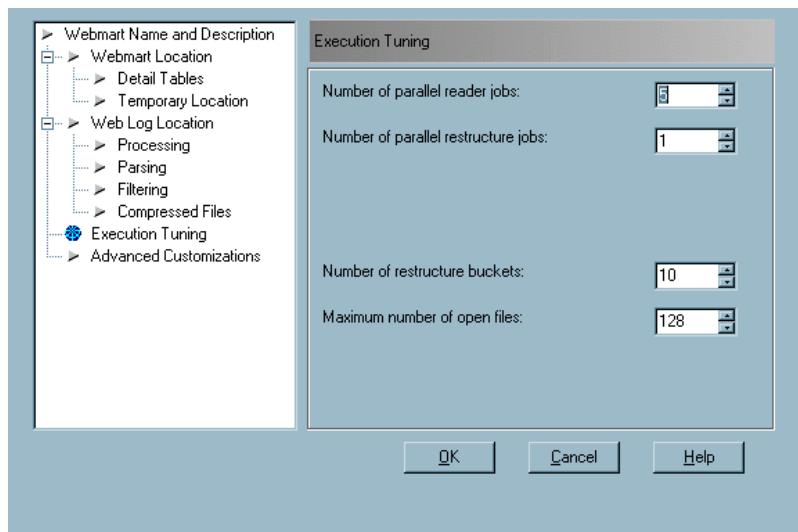
the **Select** button to locate the appropriate program, or type the path and name of the uncompress program in the text field.

If you use the uncompress command in UNIX, then enter **-c** in the **Uncompress Program Options** field (see Display 3.15 on page 50). The **-c** option converts the compressed file to the standard output that SAS e-Data ETL software reads.

The Execution Tuning Properties

In the Execution Tuning frame, you can specify the maximum number of sessions that you want to allow for different types of SAS e-Data ETL software tasks in order to minimize the bottlenecks that are caused by intensive processes.

Display 3.16 Execution Tuning Frame of the Properties Window



Navigating to the Execution Tuning Frame

From the tree view in the Properties - *<Web mart name>* window, select **Execution Tuning**. The Execution Tuning frame appears to the right of the tree view.

Description of Number of Parallel Jobs Properties

SAS e-Data ETL software uses parallel processing during the Extract process. The number of parallel jobs that are created depends on the number you specify for **Number of parallel reader jobs** and for **Number of parallel restructure jobs**. The following table describes the default values for these properties.

Table 3.7 Description of Parallel Jobs Properties

Property Name	Default Value	Description
Number of parallel reader jobs	5	Controls how many concurrent SAS sessions are started when SAS e-Data ETL software is reading Web server log data
Number of parallel restructure jobs	1	Controls how many concurrent SAS sessions are started when SAS e-Data ETL is restructuring Web server log data

The default value for this property might not be the optimal value for your site.

Number of Restructure Buckets

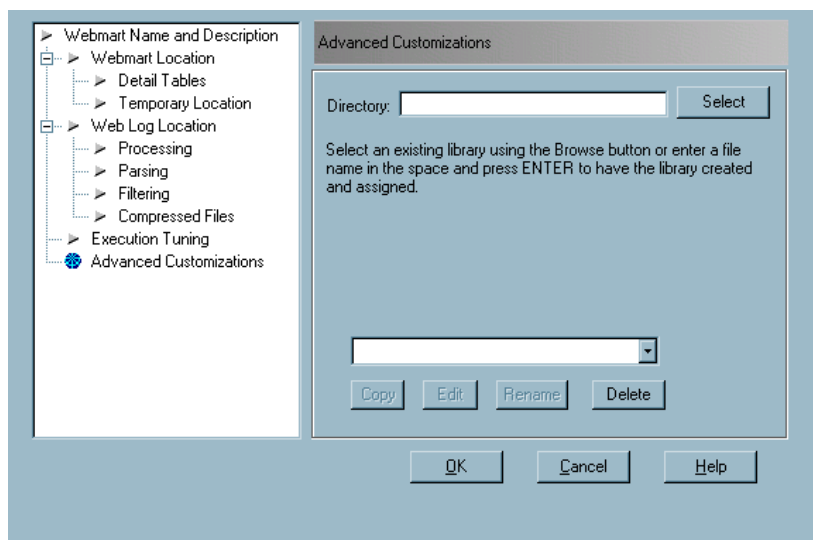
The **Number of restructure buckets** value indicates the number of divisions that are applied to the data that is being processed. By dividing the incoming data into restructure buckets, SAS e-Data ETL software sorts the data more efficiently. The default number of restructure buckets is 10.

Maximum Number of Open Files

The **Maximum number of open files** value indicates the operating environment limit. Some operating environments limit the number of files that can be opened simultaneously; some environments base this number on the account of the user. The value of this property might need to be adjusted if the programs fail or if a program has problems with opening tables due to operating environment limits. The default value is 128.

The Advanced Customizations Frame

The Advanced Customizations frame of the Properties window enables you to select, copy, and modify source entries called custom access modules (CAMs), such as `User_assignments_after_input`.

Display 3.17 Advanced Customizations Frame of the Properties Window

Navigating to the Advanced Customizations Frame

From the tree view in the Properties - <Web mart name> window, select **Advanced Customizations**. The Customizations frame appears to the right of the tree view.

About Custom Access Modules (CAMs)

The source entries that have been created specifically to enable you to customize the SAS e-Data ETL processes are called custom access modules (CAMs). The CAMs that are supplied with the SAS e-Data ETL program are located in the Sashelp.Wbetl catalog (within your SAS 9.1 directory structure). The comments within a CAM provide instructions for modifying the CAM.

CAUTION:

Do not modify any source entry that is not a CAM. Modifying a source entry other than a CAM will produce unexpected results and is strongly discouraged. △

The code from a CAM is run at the beginning or end of the Extract process, depending on the function of the CAM.

The following list contains the source entries that you can modify:

Custom_log_format

Executes any custom code for creating SAS formats that are used by the SAS e-Data ETL Extract process. This code will be executed after all of the other formats that are used by SAS e-Data ETL program have been created. Therefore, this code can be used to add new formats or to change the definition of existing formats.

Custom_log_input

Builds the input statement that will read the custom log records as specified in the call to RXMATCH. It also performs any calculations that handle the formatting for a specific Web log in order to set up values that are expected by subsequent ETL processes.

Custom_log_rxfree

Executes the call to RXFREE that deallocates the memory that is used by RXPARSE.

Custom_log_rxmatch

Executes the call to RXMATCH that will detect the new Web log format that is specified by the pattern defined in Custom_log_rxparse.

Custom_log_rxparse

Executes the call to RXPARSE that detects the new Web log format.

Custom_log_set_server_type

Sets the server type as specified by the results of the call to RXMATCH in Custom_log_rxmatch.

Custom_log_test_harness

Used to test the support of custom Web logs in an environment that is separate from the SAS e-Data ETL environment. The comment blocks located at the top of each of the custom_log_ source entries (with names that begin with **custom_log_**) contain example code that reads Common Log Format (CLF) log files. For your first custom log, run the code in this example as it is written, and make sure that the program runs successfully before you make any changes to the code. You can use the example data called **clf.log** located in SASMISC to test the example source entries.

Determine_server_and_sitename

Executes any code that is needed to parse the name of the file being processed in order to set any variables that are not available in the data itself. For example, the sitename might be used as the name of a directory in which the file is located.

User_assignments_after_input

Used in one of two ways: to set variables that are not available until after the Web log is read, or to populate any fields that are not recorded directly in the Web log. The statements in this module are executed immediately after the SAS INPUT statement.

User_assignments_before_input

Used to customize certain variables before a record is written to the SAS tables. While the Extract process is running, this module is positioned to run after the URL is parsed into all relevant pieces and after all other processing of the incoming record (within a Web log) is completed.

User_assignments_by_session_id

Used to customize certain variables after the Web log records are sorted by Session_ID and Date_Time. The User_assignments_by_session_id module is invoked during the execution of the Execute_the_analysis_jobs module, which is immediately before each observation is output.

User_assignments_for_data_mining

Used to transform the fields of your Web log to make them more effective for data mining functions. Statements that are added here will be executed automatically at the end of Load processing.

Visitor_id_logic

Calculates the visitor ID from other columns in the Control.Wbconfig data set. See also “How to Set Visitor ID Values from Cookies or CGI Parameters” on page 75.

For a complete list of the source entries that are executed by the Extract process, see “Overview: The Modules in the Extract Process” on page 80. For a complete list of the source entries that are executed by the Load process, see “Overview: The Modules of the Load Process” on page 99.

How to Modify a CAM

The Advanced Customization frame enables you to select and modify CAM source entries in order to customize specific ETL processes. When you select a catalog (such as Wbetl) that contains the CAM source entry that you want to modify, a temporary copy of the catalog is created in your Work library. Any modified source entry is saved (with the same name) in a duplicate catalog in your Usermods library, and the original source entry in the Sashelp library always remains unmodified.

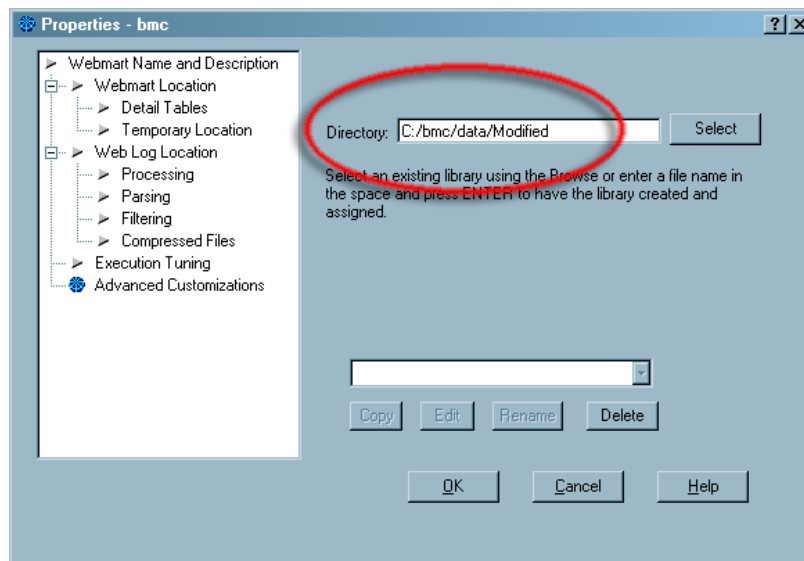
CAUTION:

The original source entries in the Sashelp library should never be modified directly. The Advanced Customization frame enables you to modify and save a copy of the source entry in the Usermods library. △

The following sections use the User_assignments_after_input CAM (one of many CAMs that are provided with the SAS e-Data ETL software) as an example to illustrate how to use the Advanced Customizations frame for modifying CAMs in the SAS e-Data ETL Administrator.


Note: You can use these steps to modify any CAM source entry by substituting the source entry name for User_assignments_after_input. △

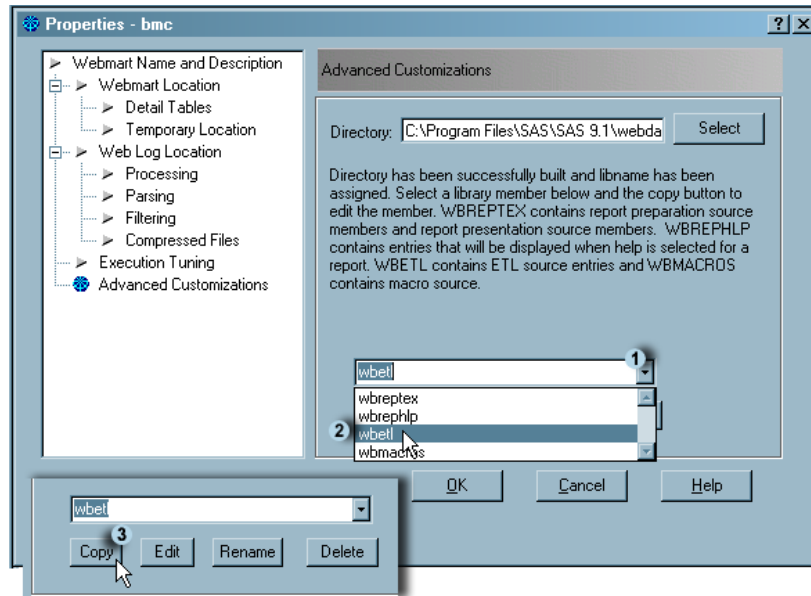
- 1 Navigate to the Advanced Customizations frame of the Properties window of your Web mart.
- 2 In the **Directory** text box, type the path to a destination library (or the location of your modified source entries). If you have not assigned your Usermods library, then type the path, including the new name, and press ENTER. The new library is created and assigned. In the following display, for example, a new library named Modified is assigned to the Usermods library:



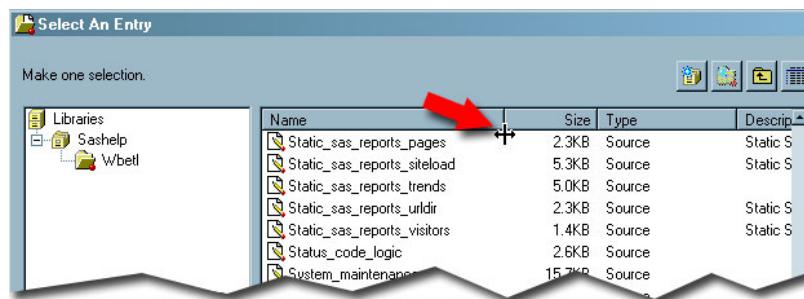
- 3 Click **Select**. The text below the **Select** button changes to prompt you to select the catalog that contains the source entry to copy (in order to modify the copy in a later step).

Note: All of the CAMs for SAS Web Analytics ETL processing are located in the Wbetl catalog of the Sashelp library. △

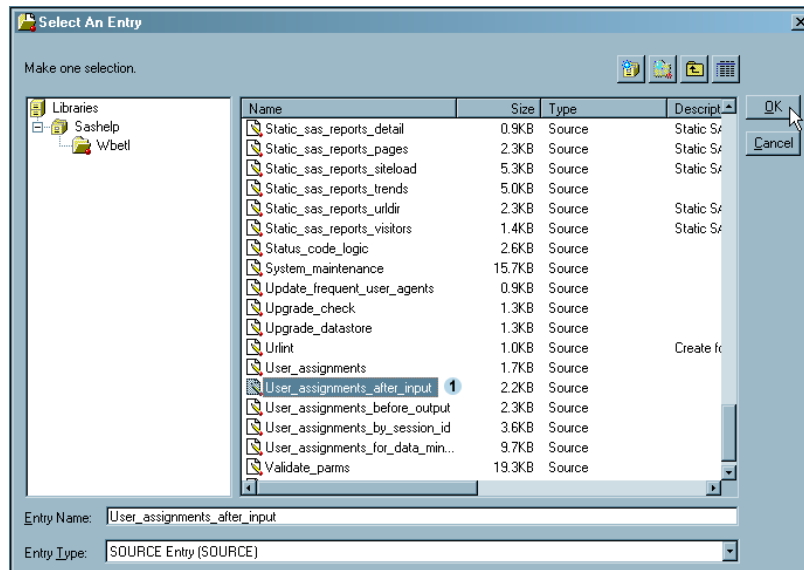
- 4 Click  (1 in the following display) to display the drop-down list.
- 5 Highlight the catalog that contains the source entry that you want to copy. The User_assignments_after_input source entry is in the Wbetl catalog, so **wbetl** is selected (2 in the following display). The selected catalog name appears in the text box.
- 6 Click **Copy** (3 in the insert of the following display). The Select An Entry window appears.



- 7 To widen the **Name** column, move the cursor over the line between the Name column and the Size column until the cursor changes (see the following display):

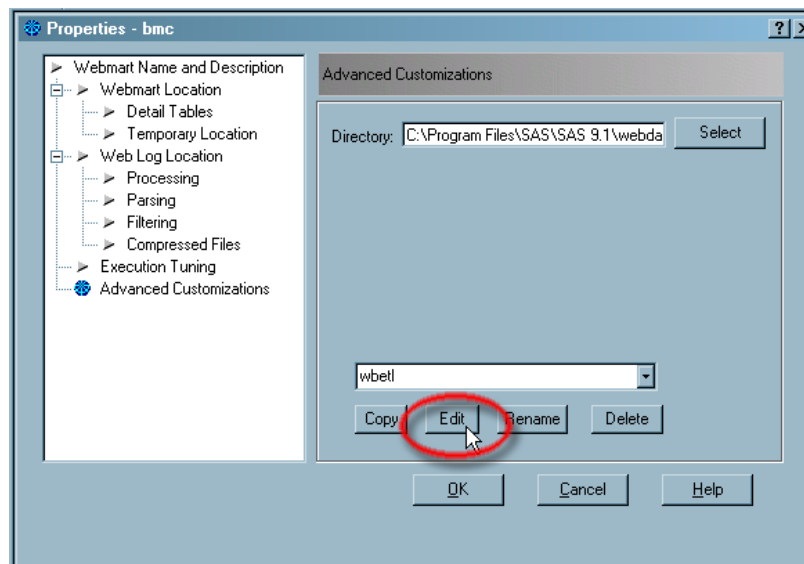


Drag the cursor to the right until you can identify the source entry names in the Name column. Highlight the source entry that you want to modify (in this example, User_assignments_after_input), and select **OK** (see the following display):



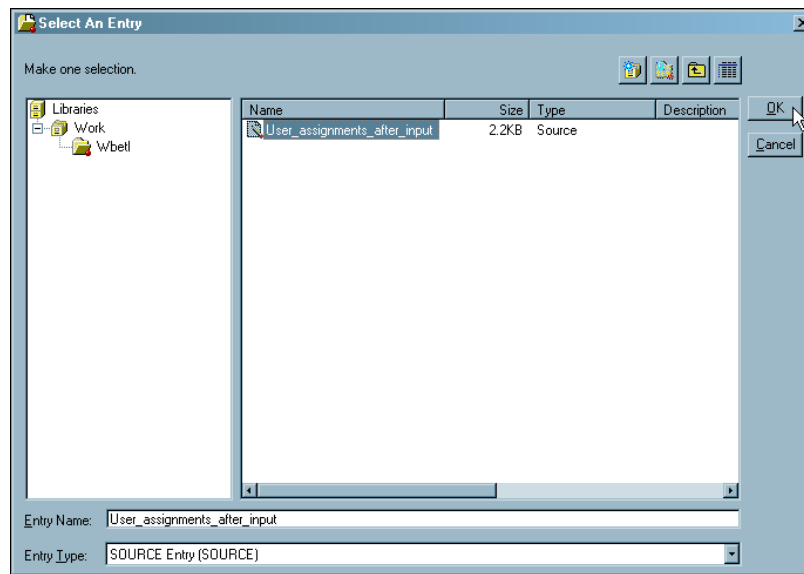
The CAM source entry is copied to your Usermods library, and the Advanced Customization frame re-appears.

- 8 Select the copy of the CAM source entry from the catalog in the Work library in order to modify the CAM source entry. Click **Edit** (see the following display):

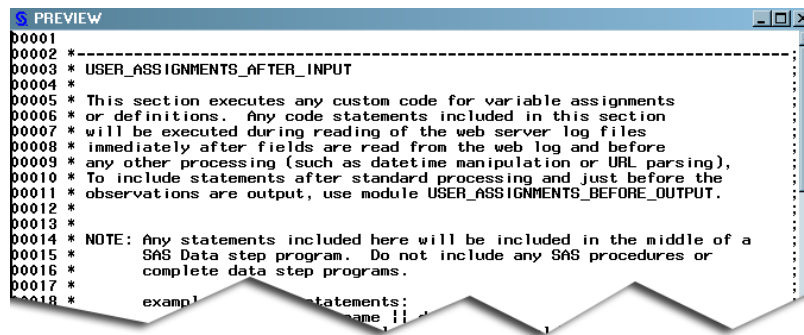


The Select An Entry window displays all of the source entries in your Work library.

- 9 Select User_assignments_after_input and click **OK** (see the following display):



A SAS Program Editor window opens (see the following display):

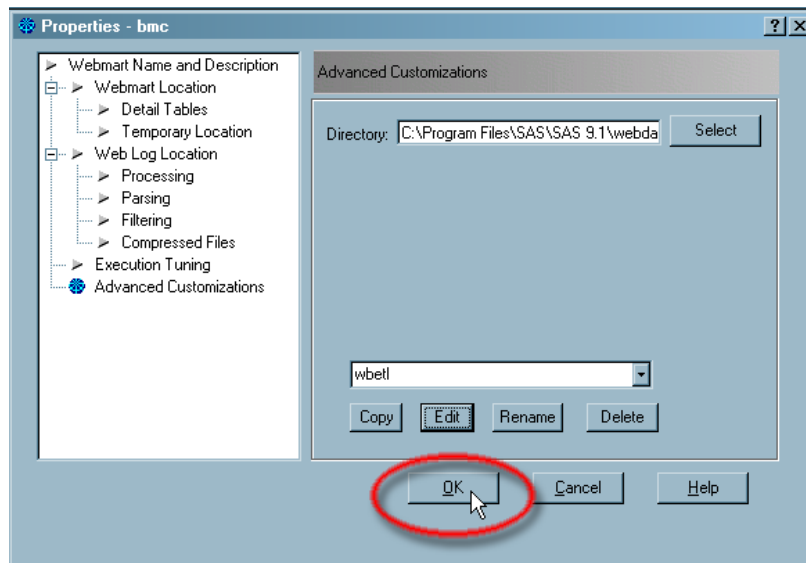


- 10 Enter your changes to the source entry code. The instructions for modifying a CAM are located in the comment lines of the source entry. When you are finished modifying the code, close the SAS Program Editor window. Click **Yes** in the small dialog box that appears to save your modifications.

Note: Click **Cancel** to return to the SAS Program Editor window where you can continue making modifications, or click **No** to lose any modifications that you made to the code and to return to the Advanced Customization frame. △

Your changes are temporarily saved to the Work library.

- 11 To modify additional source entries in the same catalog, repeat steps 7–10.
- 12 To save all of the modified source entries to your Usermods.Wbetl catalog, click **OK** in the Advanced Customization frame (see the following display):




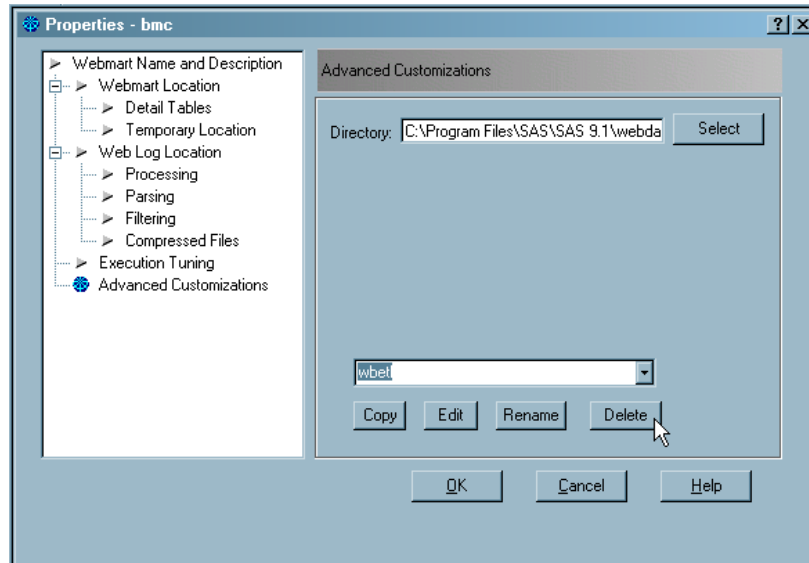
Note: If you click **Cancel** in the Advanced Customizations frame, then the modifications that you made to any source entries are lost, and no modified source entries are copied to the Usermods library. However, temporary copies of any modified source entries might exist in the Work library for as long as your SAS session remains open. See the following section to delete a source entry from the Work library. △

Deleting a CAM from the Work Library

Note: You can delete only the CAM source entries in your temporary Work library. △
To delete a CAM source entry:

- 1 In the SAS e-Data ETL Administrator, go to the Advanced Customizations frame of the Properties window of a selected Web mart.

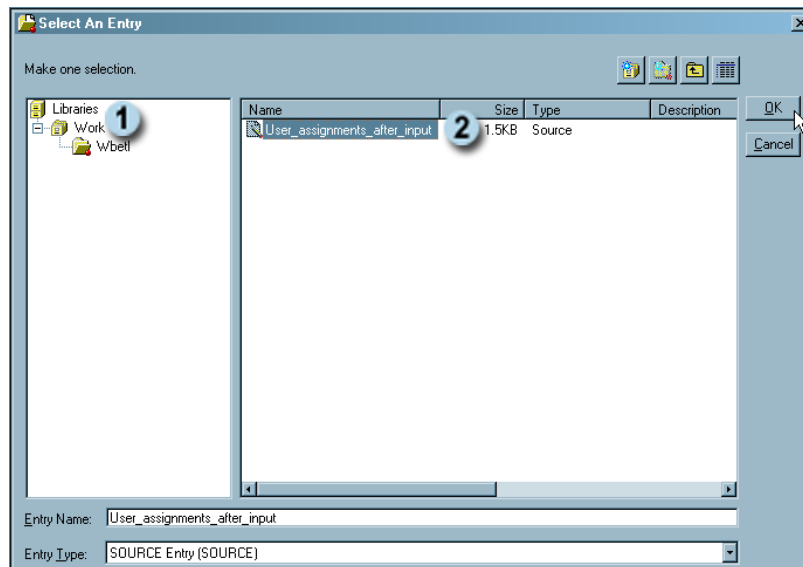
- 2 Click  near the bottom of the Advanced Customizations frame to display the drop-down list. Select the catalog that contains the CAM source entry that you want to delete and click **Delete** (see the following display):



The Select an Entry window appears, showing a list of the CAM source entries in your Work library (1 in the following display).

Note: Only the source entries that you did not save to the Usermods library will appear in the temporary Work library. △

- 3 Select the CAM that you want to delete (2 in the following display) and click **OK**



SAS e-Data ETL software deletes the source entry without confirmation and returns you to the Advanced Customizations frame.

How the Customizing Process Works in SAS e-Data ETL Software

The ETL processes read metadata tables and execute source entries, both of which are located in the Sashelp library. To customize the ETL processes, you use the SAS e-Data ETL Administrator to modify the underlying metadata, instead of modifying the metadata tables directly. You can also modify the ETL processing by customizing the selected source entries called CAMs.

The Control and Usermods libraries are important for enabling you to customize the SAS e-Data ETL process. These libraries are associated with a specific Web mart. Therefore, every time you create a new Web mart, you need to create and assign the Control and Usermods libraries for the Web mart.

The Control Library for Your Web Mart

The Control library enables you to customize the metadata tables in the Sashelp library. You can use the SAS e-Data ETL Administrator to customize most of these tables. For more information about customizing a Web mart, see “Overview: The Properties of a SAS Web Analytics Web Mart” on page 28.

Description of Specific Control Tables

The SAS metadata tables that are listed in the following table are included in the Control library.

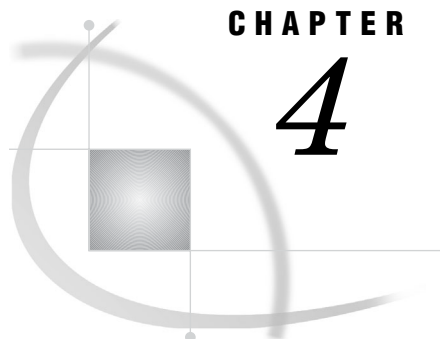
Table 3.8 Tables in the Control Library

Table Name	Contents
Wbconfig	The description and the default values for SAS e-Data ETL program parameters. Any changes that you make to the SAS e-Data ETL metadata, such as changing a default value, are reflected in this table.
Wbfields	The fields that are extracted from the Web logs and are used to create the Weblog-Detail table. To make modifications to this table, see “Adding or Deleting Fields in Your Web Server Log File” on page 38.
Wbinfile	The INFILE options for processing Web logs. In this table, you can specify the desired options for the SAS INFILE statement for each Web log. For more information, see “INFILE Options for Various Web Log Types” on page 37.
Wbpagview	All MIME types that SAS e-Data ETL software recognizes during processing. This table can be edited to exclude particular file types. For more information, see “Non-Pages” on page 48.
Wbspecl	All the host names or IP addresses that SAS e-Data ETL software should exclude during processing. For more information, see “Special Client List” on page 47.
Wbspider	A current list of all recognized spiders and robots. To learn how to update this list, see “Spiders” on page 47.

Usermods Library

The purpose of the Usermods library is to contain the catalogs of all of your modified CAM source entries and modified SAS data sets. You should create and assign a Usermods library for each Web mart.

When you use the Advanced Customizations frame of the SAS e-Data ETL Administrator, modified CAM source entries are automatically saved to the Usermods library that you either create at that time or which already exists (in a path that you specify). During ETL processing, if modified source entries exist in the Usermods library of your Web mart, then the SAS e-Data ETL Extract and Load processes automatically execute those modified source entries instead of executing the corresponding default source entries (located in the Sashelp library). For more information, see “The Advanced Customizations Frame” on page 52.



CHAPTER

4

Setting Up ETL Processing for SAS e-Data ETL Software

<i>Working with Web Logs</i>	63
<i>Guidelines for Managing Web Logs</i>	63
<i>Overview: Custom Web Logs</i>	64
<i>How to Add Support for a Custom Web Log</i>	64
<i>Correcting Data Reader Errors</i>	66
<i>Skipped Records or Fields</i>	67
<i>How to Add the Date to a Web Log</i>	67
<i>Adjusting Your Local Time Zone</i>	67
<i>Using the Daily.sas Program to Perform the ETL Processes</i>	69
<i>How to Load Data into Custom Directory Locations</i>	70
<i>Custom Invocation of the %WAETL Macro: Example 1</i>	70
<i>Custom Invocation of the %WAETL Macro: Example 2</i>	71
<i>Custom Invocation of the %WAETL Macro: Example 3</i>	71
<i>Using the %EDATAETL Macro to Extract and Load Web Log Data</i>	71
<i>When to Use the %EDATAETL Macro</i>	71
<i>How to Run the Extract and Load Processes Individually</i>	71
<i>Allocating Libraries for a Web Mart</i>	72

Working with Web Logs

Guidelines for Managing Web Logs

Because of variations in the configuration of Web servers and the security policies that are used by corporate Web sites, you will need to customize the daily maintenance of the Web log files that are processed by the SAS Web Analytics solution.

However, some Web log management tasks are common to nearly all Web sites. Following are some programming methods that you can use to automate some of these routine but critical tasks as you track and maintain your Web logs:

Close off your Web logs.

Close off the Web server log files that are retained by each of your Web servers in order to avoid losing or duplicating any information about the use of your Web site. Perform this operation at least once a day at the same time on all servers.

Your Web server should include utility software that provides instructions for automatically closing off its Web logs at specified intervals.

Rename each Web log.

Change the name of the Web log file before you process it. For example, include the Web server name and the datetime that the Web log was closed off to create a unique Web log file name.

Resolve the client IP addresses to their host names.

If your Web server does not resolve the client IP addresses to their host names for you, look up the client IP addresses in the Web log to resolve them to their host names.

Compress the Web logs.

Compress the Web server log files by using a ZIP utility program (in Windows) or by using the **compress** command (in UNIX). Be sure to specify the corresponding file options in the SAS e-Data ETL Administrator (see “The Compressed Files Properties” on page 50).

Transfer the Web logs to the SAS Web Analytics host.

Transfer the Web server log files from the Web server host(s) to the server where SAS e-Data ETL is installed in order to analyze and archive the Web log data.

Overview: Custom Web Logs

To enable the SAS e-Data ETL processes to read custom Web logs, use the source entries (located in the Wbctl catalog) that begin with Custom_log_. These Custom_log_ source entries contain examples in their code comments that you can place in each custom Web log. Refer to these examples along with the following instructions.

How to Add Support for a Custom Web Log

- 1 Create a **Usermods** directory with a Wbctl catalog that contains all the Custom_log_* source entries (both the entries that you have updated and the entries that remain unchanged).

If you want to run the example in the source entries, then copy the statements from the comment block at the top of each Custom_log_* source and paste them into a SAS Program Editor window. Remove the comment delimiters at the beginning and end of each line. You can run these examples in the test harness that is provided. A test harness is a program that replicates the SAS e-Data ETL environment. Because the test harness enables you test your code outside SAS e-Data ETL software, it is faster and simpler to use.

- 2 Add entries to the Control.Wbfields table for any of the new fields that you are defining. If you are running the example, then you do not need to add any fields. If you forget to add a field to Wbfields, then the test harness will work, but SAS e-Data ETL software will not keep the variable in the Web mart when it reads the custom log.
- 3 Add an entry for your custom log format to the Control.Wbinfile table. This entry gives your new log format a name (in all uppercase) and supplies options for the INFILE statement that is used to read the data. If you are running the example, then duplicate the observation for CLF and update the Web server name to "EXAMPLE."
- 4 Set the macro variables described in the following table.

Table 4.1 Macro Variables to Set in the Custom_log_test_harness Module

Macro Variable	Definition
Custom_Weblog_Filename	The directory path to the location of your Web log.
Control_Location	The directory path of your Web mart's permanent results with /control appended to the end. On UNIX, it is the directory for your Web mart with /control appended to the end. The /control specifies the Web mart's control directory.
Custom_Log_Name	The unique name that is used to identify your custom Web log. Web log names that the SAS e-Data ETL process recognizes are CLF, IISOrig, MSPProxy, ELF, and Netscape. When naming your custom Web log, do not use any of these values. The name that you choose should be a valid SAS variable name.

- 5 Make any modifications to the Custom_log_format module that are needed. The Custom_log_format module contains any custom code necessary to create SAS formats that the Extract process uses. This code is executed after all of the other formats for SAS e-Data ETL processing have been created.
- 6 Make any needed modifications to the Custom_log_rxpath module. This module contains a call to the RXPARSE SAS function. Follow the example below. You can find this example in the comment block within the source code of the Custom_log_rxpath module.

Detect a log by using the following format. (Note that this is a single line of code. It has been broken here to fit on the page.)

```
client-ip remote_login cs-username [datetime] req-type
cs-usr-stem sc-status sc-bytes
```

Note that the cs(Referrer), cs(User-Agent), and Cookie fields are optional.

- 7 Make any changes to the Custom_log_rxmatch module that are needed. To make these changes, use the following format (continuing with the example):

```
match_example = rxmatch(example, _infile_);
```

- 8 Make any changes to the Custom_log_input module that are needed. The code in this section builds the input statement that will read the custom log records as specified in the call to the RXMATCH function. The code also performs any calculations that are specific to the log format in order to set up values that are expected by additional SAS e-Data ETL processing. These values include the following:

Combined_URL	Should contain the URL from the input record so that it will be parsed according to RFC 2396.
Query_String	Should contain the query string so that it will be parsed into name-value pairs. (The query string can also be included as part of the Combined_URL. In that case it will be separated from the requested file by a question mark.)
Username	Should contain the authorized user ID that a user entered if he was prompted by the Web server.
Datetime	Should contain the datetime of the Web log record so that the datetime derivative variables can be calculated from it.

GMT_Offset Should contain the GMT offset of the timestamp in the Web log record, though the processing that occurs later by default does not reference this field. You also need to modify the Datetime_logic program to use the GMT offset.

- 9 Make any changes to the Custom_log_set_server_type module that are needed. This module sets the server type as specified in the results of the call to RXMATCH in the Custom_log_rxmatch module.
- 10 Change the Custom_log_rxfree module to include the call to the RXFREE call routine.
- 11 Run the SAS e-Data ETL Extract process and check for errors before continuing. After the Extract process runs successfully, proceed to the SAS e-Data ETL Load process.

For more information about the RXPARSE or RXMATCH functions or the RXFREE call routine, see SAS Help and Documentation.

Correcting Data Reader Errors

Data reader errors may occur during the execution of the SAS e-Data ETL Extract process. As long as the Extract process can read some parts of the Web log, it continues to run. However, if the Extract process can read no part of the Web log, then a fatal error occurs, and the process stops.

The following table lists the most common causes of data reader errors and offers possible solutions:

Table 4.2

Cause of a Data Reader Error	Solution
A blank that is not enclosed in double quotes exists in a field.	Using the SAS e-Data ETL Administrator, change the INPUT and INFORMAT modifiers so that blanks are enclosed in double quotes.
The datetime field is missing from a record.	Modify your Web log to include the datetime field (see “How to Add the Date to a Web Log” on page 67).
The process is backloading large volumes of data.	To minimize problems that might occur when you backload data, break the compressed data into smaller pieces before processing it. For example, process data for only one day within a single execution of the ETL process. In the Execution Tuning frame of the SAS e-Data ETL Administrator, you can also adjust the settings of the parallel job properties in order to minimize these errors (see “The Execution Tuning Properties” on page 51).

Cause of a Data Reader Error	Solution
A field value is expressed as a name-value pair.	Change the INFORMAT modifier of the field in the Control.Wbfields table to include the name string.
The process does not recognize the format of the Web server log file.	Modify the Web log or add the Web log as a custom log so that SAS e-Data ETL software will recognize and process the file.

Skipped Records or Fields

You might discover that the total number of records that are reported in a SAS Web Analytics report are not equal to the number of records in the processed Web log. To determine the reason(s) for any discrepancies of this nature, check the following factors that affect whether the ETL process either counts or skips a record in a Web log:

Table 4.3 Factors that Cause the ETL Process to Ignore a Web Log Record

Cause	Description
Incorrect syntax	The ETL process skips a record if a line in a Web log contains incorrect syntax. For example, if no blank space exists between the keyword GET and the URL, then the record is skipped. Examine the syntax of the Web log. If necessary, make the correction to the syntax (see “Overview: Custom Web Logs” on page 64).
URL contains a GIF	If a URL contains a GIF parameter, then the ETL process interprets the record as a non-page.
How you defined spider clients	If you have set the action for spider clients to skip , then the ETL process will not count any records that correspond to requests that come from a recognized spider client.

How to Add the Date to a Web Log

If your Web log does not contain the date (within the datetime field), then ask a system administrator to turn on the date option so that the Web server will add a datetime field to the Web log. For accuracy in reporting, you want the date when each record was written rather than the date that comes from a file header or some other location.

If you need to work with the dates in the Web log, then add code to the first subsection, Detect Header, of the Job_build__inputs CAM (see “Job_build__inputs” on page 83) in order to use the date from the header record of the Web log as the date for the session. The Job_build__inputs CAM is used to determine the layout of the Web server log file.

Adjusting Your Local Time Zone

To adjust for your time zone in relation to Greenwich Mean Time (GMT), modify the appropriate variable in the Sashelp.Wbetl.Datetime_logic CAM. To modify a CAM, see “How to Modify a CAM” on page 55. Here is the code for the Datetime_logic CAM:

```

=====;
* section: datetime logic ;
* ;
* descrip: this section calculates datetime and its derivatives. ;
* ;
=====;
* ;
%put ##### ;
%put # Datetime logic # ;
%put ##### ;

%macro Datetime_Logic;
    if datetime=. then delete ;
/*-----*
* Adjust datetime stamp for Daylight Savings Time and for GMT Offset
* replacing the value surfaced in the web server log file.
*
* NOTE: Most servers report local time and most GMT offsets have already
* been adjusted for daylight savings time. These calculations are
* provided as examples of the adjustments that may be necessary to
* convert the reported time to GMT, or Coordinated Universal Time.
*
*
* * Example adjustment to datetime for Daylight Savings Time;
* dstime = dhms(mdy(4,8-weekday(mdy(4,7,year(datepart(datetime)))),
* year(datepart(datetime))),3,0,0)
* < datetime <
* dhms(mdy(10,32-weekday(mdy(10,31,year(datepart(datetime)))),
* year(datepart(datetime))),2,0,0);
* if dstime then datetime = datetime - ((GMT_Offset + 1) * 3600);
* else datetime = datetime - (GMT_Offset * 3600);
*
*-----*/
*-----;
* Calculate derivatives of Datetime as rounded dates. ;
* ;
* Each DATETIME derivative variable (date, week, month, quarter, and year);
* must be an actual SAS date value for the PROC SUMMARY at the ;
* end of Extract to identify the data properly. That is, these variables ;
* must contain a fullyfledged SAS date value, not a shortened form ;
* like 1-4 for quarter. The other datetime-related variables like ;
* HOUR and DAY_OF_WEEK are not used to pin an observation down in time ;
* and so do not have this restriction. ;
* ;
* Note also that every WebHound table *must* contain at least one ;
* recognizable DATETIME derivative variable so that generations of the ;
* data can be "managed over time" successfully. ;
*-----;
Date = datepart(datetime) ;
Week = (Date - weekday(Date)+1);
Month = mdy(month(Date),1,year(Date)) ;
Quarter = mdy(month(date) - mod(month(date)-1,3),1,year(date));
Year = mdy(1,1,year(Date));
* These variables do not need to be SAS date values;

```

```

Hour = hour(timepart(Datetime)) ;
Day_of_Week = weekday(Date) + 1; * 1 = Sunday, 2 = Monday ... ;
%mend Datetime_Logic;
%Datetime_Logic;
=====;
* END of DATETIME LOGIC ;
=====;

```

Using the Daily.sas Program to Perform the ETL Processes

When you run the %WAETL macro to initialize your new Web mart, the Daily.sas program (see the sample below) is created in the **sas** directory of your SAS Web Analytics Web mart. The Daily.sas program automates the ETL processes by invoking the following programs through the associated macros:

- 1 the Extract process (%EDATAETL macro)
- 2 the Load process (%EDATAETL macro)
- 3 the Warehouse process (%WAETL macro)

Before you can use the Daily.sas program, you need to modify it according the instructions in the header block of the program. If you have a customized directory structure, you also need to modify the locations of the custom directories in the corresponding arguments within the %WAETL macro.

```

/**
 * Use this program to process weblog data into the SAS Web Analytics warehouse.
 *
 * IMPORTANT: You must supply the location of the weblogs to process.
 *
 *      Either
 *
 *      A. Specify the location in the WEBLOGS macro variable below by replacing
 *
 *          with the location of your weblogs. WEBLOGS can
 *
 *          be the full path either for a single weblog or for a directory that
 *
 *          contains multiple weblogs.
 *
 *      Or
 *
 *      B. Launch the e-Data ETL administrative GUI from an interactive SAS session
 *
 *          using the "edataetl" command and specify the location of your weblog(s)
 *
 *          in the appropriate field.
 *
 *      If you use method B. then you must remove the "weblog_path=%weblogs" parameter
 *
 *      from the %edataetl call below.
 */

%MACRO SWA_ETL;

*** TO DO: specify the location of your weblogs;
%let WEBLOGS=;

%let webmart=d:\swa\prod\bmc\swamart-pub;

```

```

%let wbrc=0;

/* Extract raw weblog data into staging datasets. */
%dataetl( data_store   = &webmart\e-data-etl,
          weblog_path  = &weblogs,
          program      = extract
        );

/* Load web data into final detail dataset. */
%dataetl( data_store   = &webmart\e-data-etl,
          program      = load
        );

/* Load web detail data into warehouse. */
%if &wbrc = 0 %then %do;

    %waetl( swamart    = &webmart,
            program    = warehouse
          );

%end;

%MEND SWA_ETL;
%SWA_ETL;

```

How to Load Data into Custom Directory Locations

If you have a customized directory structure, then you need to modify the locations of the custom directories in the corresponding arguments within the %WAETL macro. The following examples offer some suggestions for setting up your invocation of the %WAETL macro.

Custom Invocation of the %WAETL Macro: Example 1

This example loads the standard parsed Web log from the standard SAS e-Data ETL application directory for the **swamart-pub** Web mart, storing the intermediate data sets in the existing **d:\swa\prod\bmc\swamart-pub\tempstore** directory.

```

%waetl(name=swamart-pub,
        temp_store=d:\swa\prod\bmc\swamart-pub\tempstore,
        program=warehouse
      );

```

Custom Invocation of the %WAETL Macro: Example 2

This example loads the customized parsed Web log called **custom_log_detail** from the customized SAS e-Data ETL application directory (**d:\custom\e_data_etl**) for the Web mart whose root is at **d:\swa\prod\bmc\swamart-pub**, and stores the intermediate data sets in the default temporary directory.

```
libname custom 'd:\custom\e-data-etl\detail';
%waetl(detail=custom.custom_log_detail,
       swamart=d:\swa\prod\bmc\swamart-pub,
       program=warehouse
);
```

Custom Invocation of the %WAETL Macro: Example 3

This example loads the customized parsed Web log called **custom_log_detail** from the standard SAS e-Data ETL application library (Detail) for the **swamart** Web mart, storing intermediate data sets in the default temporary directory.

```
libname detail 'd:\swa\prod\bmc\swamart\e-data-etl\detail';
%waetl(name=swamart-pub,
       detail=detail.custom_log_detail,
       program=warehouse
);
```

For more about using the %WAETL macro, see “The %WAETL Macro” on page 184.

Using the %EDATAETL Macro to Extract and Load Web Log Data

When to Use the %EDATAETL Macro

If you customize either the Extract or Load processes by editing one or more of their customer accessible modules (CAMs), then you might want to invoke these programs individually rather than as part of the Daily.sas program.

How to Run the Extract and Load Processes Individually

To run the Extract or Load processes individually, submit the %EDATAETL macro in a SAS interactive session (SAS Program Editor window). Run these processes in the following order:

- 1 Extract
- 2 Load

Execute the SAS e-Data ETL Extract and Load processes by submitting the %EDATAETL macro in the SAS Program Editor window as follows:

```
%edataetl(name=webmart-name, program=program-name);
```

For the NAME argument, type the name of your Web mart exactly as it appears in the SAS e-Data ETL Administrator main window. For the PROGRAM argument, valid arguments include EXTRACT or LOAD.

For example, to run the Extract process for the BMC Web mart (the name of the Web mart for the BetterManagement.com Web site as registered in the SAS e-Data ETL Administrator), submit the following code:

```
%edataetl (name=bmc, program=extract);
```

Rarely would you run only one program without running the other program. To run both the Extract and the Load processes, type the %EDATAETL macro twice. For example, to run both the Extract and the Load processes for the BMC Web mart, submit the following code:

```
%edataetl (name=bmc, program=extract);
%edataetl (name=bmc, program=load);
```

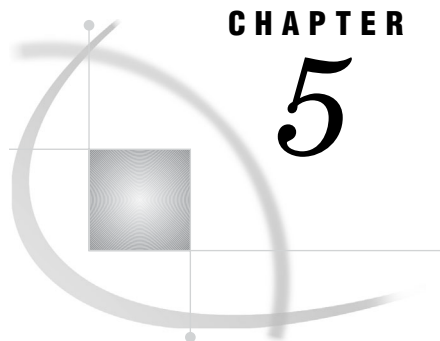
However, if an error existed in the Extract process, then the Load process would probably also fail. By executing only the Extract process, then you might check the SAS log for errors before executing the Load process, thereby possibly saving some time.

Allocating Libraries for a Web Mart

If you want only to allocate your libraries for a Web mart, then submit the %EDATAETL macro in the SAS Program Editor window by using the following code:

```
%edataetl (name=<webmartname>);
```

The %EDATAETL macro establishes the required libraries, copies the necessary control tables, and reads the Web mart properties for the Web mart as specified in the SAS e-Data ETL Administrator.



CHAPTER

5

The SAS e-Data ETL Extract Process

<i>Overview of the Extract Process</i>	73
<i>Functions of the Extract Process</i>	73
<i>Identifying Available Web Logs</i>	73
<i>Determining Web Server Type</i>	74
<i>Creating SAS Reader Programs</i>	74
<i>Sorting the Data</i>	74
<i>Determining Visitors and Sessions</i>	74
<i>Overview: The Visitor ID</i>	74
<i>How to Set Visitor ID Values from Cookies or CGI Parameters</i>	75
<i>How a Session Is Defined</i>	75
<i>How Session Duration Is Defined</i>	75
<i>Dates in a Web Log</i>	76
<i>Modules That You Can Modify in the Extract Process</i>	76
<i>User_assignments_after_input CAM</i>	76
<i>User_assignments_before_output CAM</i>	76
<i>User_assignments_by_session_ID CAM</i>	76
<i>Running the Extract Process</i>	77

Overview of the Extract Process

The purpose of the Extract process is to read the Web server log files (Web logs) and to create SAS detail data sets. The Extract process parses the Web log data into a temporary list of detail data sets in preparation for the Load process.

As it reads each Web log, the Extract process performs the following actions:

- identifies visitor sessions and the clickstream path that each visitor followed
- discovers cookie information if it is present
- identifies requests for executable programs such as CGI parameters
- filters the Web log data, identifying non-page items, requests from special clients, requests from robots and spiders, and requests that generate bad status codes

Functions of the Extract Process

Identifying Available Web Logs

When the Extract process starts, it searches the directories that are specified in the SAS e-Data ETL Administrator and makes a list of the files in each directory. The Extract process processes all files that it identifies as valid Web server log files in these directories and any subdirectories.

Determining Web Server Type

The Extract process then determines the type of Web server that created the Web log. The following types of Web logs are recognized by default:

- Common Log Format (CLF), also known as Combined Log Format or extended Common Log Format, most often associated with Apache Web servers
- Extended Log Format (ELF), most often associated with Microsoft Internet Information Server (IIS) Web servers
- Netscape/iPlanet, used by Netscape and iPlanet Web servers
- Microsoft Proxy, used by Microsoft Proxy servers
- IIS (original), used exclusively by Microsoft's personal Web Servers

You can add new or custom Web log formats to the list of formats that can be read by modifying a default definition for one of the recognized Web log formats.

Creating SAS Reader Programs

After the Web logs are located, SAS e-Data ETL software generates SAS programs that read the Web logs. User specifications (such as filtering processes that ignore page requests from spiders or identify non-page requests) that have been defined for the Web mart are included in the code.

Sorting the Data

The Extract process concurrently executes the programs that read the Web logs. Then, the data that is extracted from the Web logs is sorted and combined with the existing data that is stored in detail tables. The detail tables are located in the permanent results directory.

Determining Visitors and Sessions

Overview: The Visitor ID

The task of tracking visitors and sessions has inherent complications that affect all Web log analysis programs. If a visitor accesses a Web server through an Internet gateway, then Internet Protocol (IP) addresses can be assigned dynamically to the visitor from a range of available IP addresses. This situation can arise when the client uses an Internet service provider such as America Online (AOL) or Microsoft Network (MSN). Two individual AOL visitors, for example, might be considered as a single visitor because the AOL server repeatedly uses IP addresses from the same pool of available addresses. Proxy servers can create similar problems.

Many Web sites use cookies to better determine the identity of a visitor and the visitor's session.

In this example of a field from a Web log record, the visitor ID is a string that is set by the SHOPPERID parameter in a cookie:

```
SHOPPERID= FBBICEFBGLAGEIJDCLDGGJL
```

In the following example, the visitor ID is a combination of the visitor's IP address and the string that describes the visitor's browser and browser version:

```
24.142.56.13Mozilla/4.61+(Win98;+I)
```

How to Set Visitor ID Values from Cookies or CGI Parameters

To determine the value for visitor ID from a cookie or CGI parameter, you must specify the cookie or the CGI parameter in the Value column for the UBIQUITOUS_IDENTIFIERS parameter (located in the Wbconfig.sas7bdat data set) by performing the following steps:

- 1 Using the SAS Table Editor, select and open the Wbconfig data set (Wbconfig.sas7bdat) in the Control library.
- 2 Scroll down to the 44th row of the Wbconfig table. The parameter called UBIQUITOUS_IDENTIFIERS is in the Parameter_Name column.
- 3 In the Value column of this row, enter a value that corresponds to the cookie or CGI parameter. See the Description column in the same row for a list of the options that you can use to specify the value(s) for the UBIQUITOUS_IDENTIFIERS parameter.

Note: For more about the Wbconfig data set, see “How the Customizing Process Works in SAS e-Data ETL Software” on page 61. △

It is also useful (although not required) to use the COOKIES_TO_COPY or CGIPARMS_TO_COPY parameters (also located in the Wbconfig data set) in conjunction with the UBIQUITOUS_IDENTIFIERS parameter. When you enter **cookies_to_copy** or **cgiparms_to_copy** as the value for the UBIQUITOUS_IDENTIFIERS parameter, the column name that is specified in the COOKIES_TO_COPY or CGIPARMS_TO_COPY parameter is also used as the value for the UBIQUITOUS_IDENTIFIERS parameter.

For example, if you have a cookie that you assign to the User_Site_Logon column using the COOKIES_TO_COPY parameter, and then you set **cookies_to_copy** as the value of the UBIQUITOUS_IDENTIFIERS parameter, then the value in the User_Site_Logon column will be prefixed with the letter "U" and will be assigned to the output column Visitor_ID in the Weblog_Detail table.

The e-Data ETL Extract process determines the visitor ID in the following order:

- 1 If a value for the UBIQUITOUS_IDENTIFIERS parameter exists in the Wbconfig table, then this value is used and is prefixed with the letter "U" by the Extract process.
- 2 If no value for the UBIQUITOUS_IDENTIFIERS parameter exists, then the standard server cookie parameter, SITESERVER ID, is used and is prefixed with the letter "U" by the Extract process. The SITESERVER ID parameter is located in the cookie parameters (in the Web log) that are prefixed with **SITESERVER=ID=**.
- 3 If the SITESERVER ID cookie parameter is not available, then the value for the Visitor_ID column is generated by using the two output fields Client_ID and User_Agent. This value is prefixed with the letter "N" by the Extract process.

How a Session Is Defined

SAS e-Data ETL software uses the following method to define a session when a more accurate ID source (cookie, authentication, or registration) is not available.

SAS e-Data ETL software counts only sessions that contain at least one page view. The Entry_Point value is a count of the number of sessions in which at least one page was viewed. When the first page is viewed, Entry_Point is set to 1. Non-page views, such as bad status codes or image files, are ignored if they precede the first page. If a session contains no page views, then neither the Entry_Point nor Exit_Point values are set.

How Session Duration Is Defined

By default, if a visitor does not make a new request for a page within 30 minutes of the last request, SAS e-Data ETL software considers the current session ended. SAS

e-Data ETL software interprets any additional time that a visitor remains at a page (beyond this 30-minute timeout threshold) as part of a new session for that visitor. To modify the default timeout threshold of a session for your Web mart, see “The Processing Properties” on page 35.

Dates in a Web Log

If the date is not specified for each record within a Microsoft Web log, then the Extract process uses the date in the heading of the log for processing those records.

If a date is not present in the heading of a log that has originated from other Web servers, then the Extract process skips this log.

Modules That You Can Modify in the Extract Process

The modules of the Extract process are described in “Overview: The Modules in the Extract Process” on page 80. The modules of the Extract process that you can modify are called custom access modules (CAMs). The CAMs for the Extract process include the following modules:

- User_assignments_after_input
- User_assignments_before_output
- User_assignments_by_session_id

CAUTION:

Aside from the CAMs listed in this section, you should not modify, customize or override any other modules of the Extract process. △

For the general instructions on modifying any CAM, see “About Custom Access Modules (CAMs)” on page 53. To see the code in any of the following CAMs, see “Overview: The Modules in the Extract Process” on page 80.

User_assignments_after_input CAM

Edit the User_assignments_after_input module to accomplish either of the following operations:

- to set variables that are not available until after the Web log is read
- to populate any fields that are not recorded directly in the Web log

The statements in this module are executed immediately following the SAS INPUT statement.

User_assignments_before_output CAM

Edit the User_assignments_before_output module to customize selected variables before a record is written to the SAS data sets. This module runs after the URL is parsed into all relevant pieces and after all other processing of the incoming record (within a Web log) is completed.

User_assignments_by_session_ID CAM

Use the User_assignments_by_session_ID module to customize selected variables after the Web log records are sorted by Session_ID and Datetime.

Running the Extract Process

The Extract process is normally executed automatically within the Daily.sas program as part of the ETL processing of your Web log data. To run the Extract process separately from the other ETL programs, invoke the %EDATAETL macro by typing **extract** (in the SAS Program Editor window) as the argument of the PROGRAM parameter as follows:

```
%edataetl(name=mybigcompany, program=extract);
```

The following example demonstrates how to use macro variables when calling the %EDATAETL macro:

```
/* Set up macro variables */

%let weblog_path=C:\MyData\WebLogs;
%let data_store=C:\My Data\MyWebMart\&work;
%let work_area=C:\My Data\MyWebMart\&work\TEMP;
/* Initialize the Web mart */

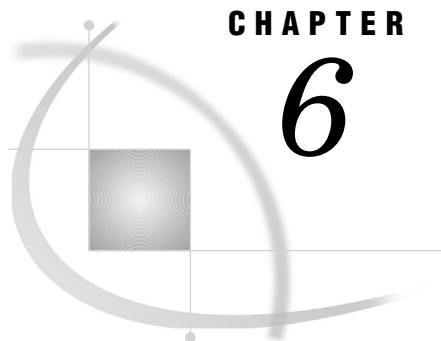
%edataetl(
  weblog_path=&weblog_path,
  data_store=&data_store,
  work_area=&work_area
);

/* Run Extract program */

%edataetl(
  weblog_path=&weblog_path,
  data_store=&data_store,
  work_area=&work_area,
  program=extract
);

/* Run Load program */

%edataetl(
  weblog_path=&weblog_path,
  data_store=&data_store,
  work_area=&work_area,
  program=load
);
```

CHAPTER

6

Reference Notes for SAS e-Data ETL Extract Process

Overview: The Modules in the Extract Process 80

The Modules in the Extract Process 80

Create_formats 80

Custom_log_format 81

Custom_log_input 81

Custom_log_rxfree 81

Custom_log_rxmatch 81

Custom_log_rxpathse 81

Custom_log_set_server_type 81

Custom_log_test_harness 81

Determine_available_weblogs 81

User_assignments_for_data_mining 82

Determine_weblog_type 82

Extract_build_filenames 82

Save_macro_variable_settings 82

Job_build__main 82

Job_build__inputs 83

Job_build__data_attributes 83

User_assignments_after_input 83

Determine_server_and_sitename 85

Main_variable_assignments 85

Datetime_logic 85

Special_client_logic 85

Spider_client_logic 85

Visitor_ID_logic 86

Determine_executable_program 86

Parse_main_url 86

Parse_referrer 87

Page_logic 87

Status_code_logic 87

Determine_browser_and_platform 87

Determine_organization 87

Determine_unique_url 88

User_assignments_before_output 88

Output_referrer_query_parms 90

Output_referrer_pathinfo_parms 90

Output_cookie 90

Output_query_string_parms 90

Output_pathinfo_parms 90

Job_build__output_logic 91

Sort_buckets 91

<i>Extract_initialization</i>	91
<i>Execute_the_data_read_jobs</i>	91
<i>Aggregate_job_datasets</i>	92
<i>Execute_the_analysis_jobs</i>	92
<i>Aggregate_analysis_stats</i>	92
<i>User_assignments_by_session_ID</i>	92
<i>Check_sas_logs_for_metrics</i>	94
<i>The Temporary Working Area</i>	95
<i>Overview: The Temporary Working Area</i>	95
<i>Read_jobn.sas</i>	95
<i>Analysis_jobn.sas</i>	95

Overview: The Modules in the Extract Process

The section entitled “The Modules in the Extract Process” on page 80 briefly describes the source entries in the Extract process so that you can identify their general functionality. The source entries are listed in order of data flow.

CAUTION:

The only source entries in the SAS e-Data ETL Extract process that you can modify are the custom access modules (CAMs) that are listed in “About Custom Access Modules (CAMs)” on page 53. △

To modify a CAM, see “About Custom Access Modules (CAMs)” on page 53.

For descriptions of the CAMs in the Extract process and instructions for modifying them, see the following sections:

- “User_assignments_after_input” on page 83.
- “User_assignments_before_output” on page 88.
- “User_assignments_by_session_ID” on page 92.

Note: For the module(s) in the Load process that you can edit, see “User_assignments_for_data_mining” on page 100. △

The section entitled “The Temporary Working Area” on page 95 describes the significance of the temporary working area and describes the outputs that are created in the temporary working area.

The Modules in the Extract Process

Create_formats

The `Create_formats` module defines the SAS formats that the Extract process uses. Because the data upon which the formats are built changes often (as in the case of the current list of the spiders or robots), the Extract process defines these formats at every invocation to ensure that the most recent changes are included when the logs are read. The formats are created from SAS tables that are supplied with SAS e-Data ETL software. These tables are located in the Sashelp library, and their names begin with **WB**.

Custom_log_format

This module executes any custom code for creating SAS formats that are used by the SAS e-Data ETL Extract process. This code is executed after all of the other formats that are used by SAS e-Data ETL program have been created. Therefore, this code can be used to add new formats or to change the definition of existing formats.

Custom_log_input

This module builds the input statement that reads the custom log records as specified in the call to RXMATCH. It also performs any calculations that handle the formatting for a specific Web log in order to set up values that are expected by subsequent ETL processes.

Custom_log_rxfree

This module executes the call to RXFREE that deallocates the memory that is used by RXPARSE.

Custom_log_rxmatch

This module executes the call to RXMATCH that detects the new Web log format that is specified by the pattern defined in Custom_log_rxpars.

Custom_log_rxpars

This module executes the call to RXPARSE that detects the new Web log format.

Custom_log_set_server_type

This module sets the server type as specified by the results of the call to RXMATCH in Custom_log_rxmatch.

Custom_log_test_harness

This module tests the support of custom Web logs in an environment that is separate from the SAS e-Data ETL environment. The comment blocks located at the top of each of the Custom_log_ source entries (with names that begin with **Custom_log_**) contain example code that reads Common Log Format (CLF) log files. For your first custom log, run the code in this example as it is written, and make sure that the program runs successfully before you make any changes to the code. You can use the example data file called **clf.log**, which is located in Sasmisc, to test the example source entries.

Determine_available_weblogs

The Determine_available_weblogs module searches for the Web log location that the user specified during the Web mart setup in order to find all Web logs that need to be

processed. If the specified location is a directory, then the module searches all files and all subdirectories within that directory for potential Web logs.

User_assignments_for_data_mining

This module transforms the fields of your Web log to make them more effective for data mining functions. Statements that are added here are executed automatically at the end of Load processing.

Determine_weblog_type

The Determine_weblog_type module examines the list of Web logs and determines the type of Web server that created these logs. This module provides useful feedback early in the module execution about the types of files that are being processed. It does not examine compressed Web log files; it only recognizes them as files to be processed.

If a Web log file is in an unknown format, then the module displays a warning in the SAS log and continues to the next Web log. The possible corrective action depends upon whether this file is in a format that SAS e-Data ETL software cannot recognize but should process anyway, or whether the unrecognized file is not actually a Web log and should be removed from the directory.

Extract_build_filenames

The Extract_build_filenames module constructs the proper SAS FILENAME statement that is required for processing each Web log file. The type of FILENAME statement that is generated might vary depending on whether or not the file is compressed and whether or not DNS resolution is specified.

Save_macro_variable_settings

The Save_macro_variable_settings module creates a file that contains all the properties of the Web mart so that these properties can be accessed by the concurrent SAS programs.

Job_build__main

The Job_build__main module (note the double underscore after *build*) builds the SAS reader programs with other modules, applying the parameters of the currently selected Web mart.

The logic modules (Special_client_logic, Spider_client_logic, Visitor_ID_logic, and Status_code_logic) apply filtering criteria in the following order:

- 1 special clients
- 2 spider clients
- 3 non-page views
- 4 bad status codes

Incoming records are divided into groups, or buckets, for more efficient processing by the system. All the concurrent SAS programs use the same algorithm for dividing the records into buckets. All of the information that pertains to a particular visitor can be

merged again, even if the records that constitute the visit to the Web site are distributed among different servers or Web logs.

The Job_build__main module includes the rest of the modules in this subsection, from Job_build__inputs through Sort_buckets.

Job_build__inputs

The Job_build__inputs source entry (a double underscore exists between “build” and “inputs” in the name of this source entry) generates the SAS code that is required for reading each specific Web log. This module opens each log, determines the type of Web server that created the log, and determines the fields that are available in the Web log.

The SAS INPUT statement is created by comparing the variables in a Web log with the information in the Control.Wbfields table.

If a compressed Web log contains other Web logs, then all the files must be of the same log format and must contain the same variables. If the Web server type cannot be determined by comparing the first 30 lines of input, then the Web log is not processed, and the program skips to the next Web log.

You can modify the Job_build__inputs module in order to accommodate any Web log formats that the Extract process might encounter but would not recognize.

Job_build__data_attributes

The Job_build__data_attributes module generates the SAS ATTRIB statements based on the information that is contained in the Control.Wbfields table.

User_assignments_after_input

The User_assignments_after_input module can be used in one of two ways: to set variables that are not available until after the Web log is read, or to populate any fields that are not recorded directly in the Web log.

The statements in this module are executed immediately after the SAS INPUT statement.

For the general instructions on modifying any CAM, see “About Custom Access Modules (CAMs)” on page 53.

The default code in the User_assignments_after_input CAM follows:

```
*-----;
* USER_ASSIGNMENTS_AFTER_INPUT                ;
*                                              ;
* This section executes any custom code for variable assignments      ;
* or definitions. Any code statements included in this section        ;
* will be executed during reading of the web server log files         ;
* immediately after fields are read from the web log and before      ;
* any other processing (such as datetime manipulation or URL parsing), ;
* To include statements after standard processing and just before the  ;
* observations are output, use module USER_ASSIGNMENTS_BEFORE_OUTPUT. ;
*                                              ;
*                                              ;
* NOTE: Any statements included here will be included in the middle of a ;
*       SAS Data step program. Do not include any SAS procedures or   ;
*       complete data step programs.                                   ;
*                                              ;
*       example of valid statements:                                   ;
*           my_variable = username || date;                            ;
*           if upcase(combined_url) = "/my/special/file.htm" then      ;
```

```

*          pagecnt = sum(pagecnt, 1);                                ;
*          if client_id eq '0.0.0.0' then delete;                    ;
*                                                                    ;
*-----;
* CHANGES:                                                         ;
*                                                                    ;
* 20021029 cabahl Added GOMEZAGENT                                C001 ;
* 20021114 cabahl added CONTYPE                                  C002 ;
* 20021204 cabahl consolidated the user agent exclusion          C003 ;
* 20030103 cabahl added logic to rename gif files for auto quote C004 ;
*          mapping                                              ;
* 2003019 cabahl added logic to rename gif files used for auto   C005 ;
*          autoquote mapping 2/4/03 redesign                    ;
*=====;

/* Make any custom definitions and assignments here */

/* C003--remove known agents that are not wanted: */
%itv_remove_agents;

if user_cookie = '-' then user_cookie='';

/* C004 - renaming gifs
PreScreen Page = images/jspnames/prescreen.gif
PreScreen Knockout - Thank You Major Violation = images/jspnames/thankYouMajorVio.gif
PreScreen Knockout - Thank you no Driver 50 or over = images/jspnames/thankyouNoDriver50.gif
PreScreen Knockout - Thank You_rfq = images/jspnames/thankyou_rfq.gif
PreScreen Knockout - Thank You - Mass. = images/jspnames/thankyou_ma.gif
PreScreen Knockout - Thank You - NJ = images/jspnames/thankyou_nj.gif
Start (State Select) = images/jspnames/Welcome.gif
Premium Page = images/jspnames/premiumSummary.gif
*/

*combined_url = lowercase(combined_url);

if index(combined_url,'images/jspnames/') > 0 then do;
  select(lowercase(combined_url));
    when('images/jspnames/prescreen.gif')      combined_url='images/jspnames/prescreen.html';
    when('images/jspnames/thankyoumajorvio.gif') combined_url='images/jspnames/thankyoumajorvio.html';
    when('images/jspnames/thankyounodriver50.gif') combined_url='images/jspnames/thankyounodriver50.html';
    when('images/jspnames/thankyou_rfq.gif')     combined_url='images/jspnames/thankyou_rfq.html';
    when('images/jspnames/thankyou_ma.gif')      combined_url='images/jspnames/thankyou_ma.html';
    when('images/jspnames/thankyou_nj.gif')      combined_url='images/jspnames/thankyou_nj.html';
    when('images/jspnames/welcome.gif')          combined_url='images/jspnames/welcome.html';
    when('images/jspnames/premiumsummary.gif')   combined_url='images/jspnames/premiumsummary.html';
    otherwise;
  end;
end;

/* C005 - 2/04 redesign */
if index(lowercase(combined_url),'/sales/images/jsptacker.gif') > 0 then combined_url=tranwrd(combined_url,'gif','jsp');
if index(lowercase(combined_url),'/service/images/jsptacker.gif') > 0 then combined_url=tranwrd(combined_url,'gif','jsp');
```

```
*-----;
* END of USER ASSIGNMENT AFTER INPUT      ;
*-----;
```

Determine_server_and_sitename

The Determine_server_and_sitename module populates the Server field and Sitename field from information that is contained in the Web log if that log contains these fields.

The Server field indicates the name of the physical system on which the Web server is running. The Sitename field indicates the name of the virtual Web site. One server can host many Web sites. Conversely, one Web site can use many physical Web servers.

Main_variable_assignments

The Main_variable_assignments module translates the variables as they are read from the Web log into the set of known variables.

Datetime_logic

The Datetime_logic module sets the following variables:

```
Date                = datepart(datetime);
Week               = (Date-weekday(Date+1));
Month              = mdy(Month(Date),1,Year(Date));
Quarter            = mdy(Month(Date)-mod(Month(Date)-1,3),1,Year(Date));
Year               = mdy(1,1,Year(Date));
Hour              = Hour(timepart(Datetime));
Day_Of_Week       = weekday(Date) + 1;
```

Where 1=Sunday, 2=Monday, 3=Tuesday, and so on. These variables do not need to be SAS date values.

Special_client_logic

Log records from clients that are listed in the Control.Wbspecl table are processed by the Special_client_logic module, based on the setting of the Special_Client_Action property. Here are the possible values of the Special_Client_Action property:

- **Skip** (discard and ignore the records from this client)
- **Keep** (process the records from this client normally)
- **Tally** (count the number of records that are received from this client, but do not process them further)

Spider_client_logic

SAS e-Data ETL software automatically detects spiders and robots when they access the **robots.txt** file. If the value of the Number_Of_Auto_Spiders field in the Control.Wbspider table is greater than zero, then any Web log records that contain clients that are designated as spider or robot programs are processed by the Spider_client_logic module. This module processes the spiders and robots based on the setting of the Spider_Client_Action property.

This property is set by using the SAS e-Data ETL Administrator, and the value is saved in the Control.Wbconfig table. Clients that are identified as robots or spiders are stored in the Control.Wbspider table.

See “Special_client_logic” on page 85 for the possible values of Spider_Client_Action property.

For information about enabling SAS e-Data ETL software to recognize new spider clients, see “Spiders” on page 47.

Visitor_ID_logic

SAS e-Data ETL software tracks a visitor through a Web site by using the Visitor_ID variable. The Visitor_ID variable is set as the value of the Ubiquitous Identifier if this option is set. Otherwise, the Visitor_ID variable is set as the value of the cookie SITESEVER=ID (if this cookie is present) or as the value of the cookie ASPSESSIONID (if this cookie is present). If neither of these cookies is present in the Web log, then the value of Visitor_ID is set as the combination of the Client_ID field and the User_Agent field. For more about Visitor_ID, see “Determining Visitors and Sessions” under “Overview of the Extract Process” on page 73.

SAS e-Data ETL software does not use the Username (or authenticated user) field as the Visitor_ID, because the Username field in a Web log is often left blank. The server can determine the Username only for those URLs that have established authentication procedures. For example, if a visitor logs on to a site (realm) that requires authentication such as a user ID from the visitor, the server can record that ID label as the Username for as long as the visitor remains within that authenticated realm. If all the sessions of interest exist within the authenticated realm that is served by the Web server, then the Visitor_ID variable can be set to equal the value of the Username field.

Determine_executable_program

The Determine_executable_program module examines the current request to determine whether the request is for an executable program, such as a CGI program. The module makes this determination by checking for CGI parameters, by checking the file extension of the requested file, and by checking the URL against the list of files and directories that are specified in the Control.Wbpgms table.

To add new types of executable programs to the Control.Wbpgms table, modify the list of files and directories by following the directions under “CGI Program Locations” on page 43.

Parse_main_url

The Parse_main_url module examines the requested URL and parses it into constituent pieces, based on the standard that is provided by RFC 2396 and by the Internet Draft of the WWW Common Gateway Protocol Version 1.1.

The Parse_main_url module also contains the logic that determines whether a request should be counted as a hit or a page view. The determination is based on the information in the Control.Wbpagview table and the file extension of the requested object. For example, if a requested file has an extension of HTM, then this request is counted as a page view. The File_Type field, which represents the MIME type of the requested object, is also set in this module.

Note: SAS e-Data ETL software counts all executable programs as page views, regardless of their file extensions. △

Parse_referrer

The Parse_referrer module examines the Referrer field and parses it into its constituent pieces, based on the standard that is provided by RFC 2396 and by the Internet Draft of the WWW Common Gateway Protocol Version 1.1.

Page_logic

The Page_logic module acts on the request as specified in the setting of the Non_Page_Views property, which you can set by using the SAS e-Data ETL Administrator. Items that are typically considered to be non-page views include graphics files. Most graphics files for Web display have a GIF or JPG extension. See “Non-Pages” on page 48 for more information about customizing a Web mart to recognize additional types of non-pages.

Here are the possible values of the Non_Page_Views property:

- **Skip** (ignore and discard the request for the non-page object)
- **Keep** (process the request for the non-page object normally)
- **Tally** (count the number of non-page object requests, but do not save detailed information about the request)

Status_code_logic

The Status_code_logic module responds to the page request as specified in the settings of the Status_Code_Action property, which you can set by using the SAS e-Data ETL Administrator. This module bases its action on whether the server indicates an error when it attempts to fulfill a page request. See “Bad Status Codes” on page 49 for more information about how SAS e-Data ETL software reports the page requests that return errors.

Here are the possible values of the Status_Code_Action property:

- **Skip** (ignore and discard the request for the page)
- **Keep** (process the request for the page normally)
- **Tally** (count the number of page requests, but do not save detailed information about the request)
- **ForceNonPageView** (treat the request the same as any other request for a non-page object)

Determine_browser_and_platform

The Determine_browser_and_platform module parses the User_Agent variable into variables that represent the following information:

- the browser program that is being used by the visitor
- the browser version
- the operating system on which the visitor’s machine is running

Because no formatting standards currently exist for specifying a browser and its version, the Determine_browser_and_platform module consults a list of known values in order to determine the visitor’s browser and platform. The list of known values is kept in the Control.Wbbrwsr and Control.Wbpltfm tables.

Determine_organization

If the visitor’s IP address was resolved into a host name by using DNS lookup, then the Determine_organization module parses the host name into organization and

organization type. If the host name contains the country and state, then the visitor's country and U.S. state are also retrieved.

Determine_unique_url

The Determine_unique_url module is used to build the path or sequence of page requests as a visitor travels through a Web site. For advanced usage, you can customize this module in order to assign meaningful phrases to this sequence, such as a name for the path of a purchase activity by a customer.

User_assignments_before_output

The User_assignments_before_output module is supplied if you want to customize selected variables before a record is written to the SAS tables. While the Extract process is running, this module is positioned to run after the URL is parsed into all relevant pieces and after all other processing of the incoming record (within a Web log) is completed.

The default code in the User_assignments_before_output CAM follows:

```
*-----;
* USER_ASSIGNMENTS_BEFORE_OUTPUT                                ;
*                                                                 ;
* This section executes any custom code for variable assignments ;
* or definitions. Any code statements included in this section   ;
* will be executed during reading of the web server log files    ;
* immediately before data is written out to SAS data sets and after ;
* any other processing (such as datetime manipulation or URL parsing), ;
* To include statements before standard processing and just after the ;
* data is read from the web log file, use module                 ;
* USER_ASSIGNMENTS_AFTER_INPUT.                                  ;
*                                                                 ;
*                                                                 ;
* NOTE: Any statements included here will be included in the middle of a ;
*       SAS Data step program. Do not include any SAS procedures or ;
*       complete data step programs.                                ;
*                                                                 ;
*       example of valid statements:                                ;
*           my_variable = username || date;                        ;
*           if upcase(requested_file) = "/my/special/file.htm" then ;
*               pagecnt = sum(pagecnt, 1);                        ;
*           if client_id eq '0.0.0.0' then delete;                 ;
*                                                                 ;
* 20020820 ccb add logic to create a spider flag                  ;
*=====;

/* Make any custom definitions and assignments here */
*-----;
* Check for the existence of name/value pair for module=        ;
* cgi parameter. If a value exists, add it to the                ;
* requested_file variable. This will enhance the page           ;
* view reports.                                                  ;
*-----;
requested_file=lowercase(requested_file);
```

```

*-----;
* Create variables for business segments.  If a session has ;
* a hit to the url level below, tag that session as being ;
* part of the segment.  The next three variables will be ;
* propagated throughout the session via wbconfig field ;
* ;
* Webhound won't allow us [yet] to propagate numeric fields ;
* so use character values.  Each segment variable must ;
* also be defined in wbfields. ;
*-----;
url_level_1=lowcase(url_level_1);
url_level_2=lowcase(url_level_2);
url_level_3=lowcase(url_level_3);
url_level_4=lowcase(url_level_4);
url_level_5=lowcase(url_level_5);
url_level_6=lowcase(url_level_6);
url_level_7=lowcase(url_level_7);
url_level_8=lowcase(url_level_8);

*-----;
* Set spider flag - uses the spider flag and other ;
* indicators to determine whether or not a detail record ;
* is a spider ;
* variables created - ;
*   spider: 0 (not a spider) ;
*           1 (spider) ;
*           .5 (potentially a spider) ;
*-----;
spider=0;
/* determine if client_id exists in WBSPIDER if so then set spider to 1 */
if input(client_id,spider.) then spider=1;

*-----;
* if client_id not in wbspider determine if other client ;
* id or requested_file meet spider criteria ;
*-----;
if spider = 0 then
  if
    /* check requested_file to see if robot */
    index(lowcase(requested_file),'robot') > 0 then spider = 1;

/* check user_agent to see if contains suspicious text */
if spider = 0 then
  if index(lowcase(user_agent),'seeker') > 0
    or index(lowcase(user_agent),'finder') > 0
    or index(lowcase(user_agent),'index') > 0
    or index(lowcase(user_agent),'crawler') > 0
    or index(lowcase(user_agent),'lwp') > 0
    or index(lowcase(user_agent),'arach') > 0
    or index(lowcase(user_agent),'spider') > 0
    or index(lowcase(user_agent),'bot') > 0
    or index(lowcase(user_agent),'borg') > 0

```

```

        then spider=.5;

*-----;
* END of USER ASSIGNMENTS BEFORE OUTPUT      ;
*-----;

```

Output_referrer_query_parms

The Output_referrer_query_parms module creates the ReferrerParms table. This table contains the CGI parameter information from the Referrer field in the Web log. This module parses the CGI parameters into name-value pairs. Then the module writes each name-value pair to a SAS table that contains the Parameter_Name, Parameter_Value, and Record_ID fields. These fields can be used to map the name-value pair back to the original record of the Web log.

Output_referrer_pathinfo_parms

The Output_referrer_pathinfo_parms module creates the ReferrerParms table. This table contains the PATH_INFO parameter from the Referrer field in the Web log. This module parses the PATH_INFO parameters into name-value pairs. Each name-value pair is written to a SAS table that contains the Parameter_Name, Parameter_Value, and Record_ID fields. These fields can be used to map the name-value pair back to the original record of the Web log.

Output_cookie

If cookie parsing is requested, then the Output_cookie module builds the SAS statements necessary to create the tables that will contain a visitor's cookie information. The cookie information is obtained from the Web log.

The Output_cookie module parses a cookie string into its name-value pairs. Then the module writes each name-value pair to a SAS table that contains the Parameter_Name, Parameter_Value, and Record_ID fields. These fields can be used to map the name-value pair back to the original record of the Web log.

See “The Detail Tables Properties” on page 31 for information on modifying the Number_of_cookie_datasets property.

Output_query_string_parms

The Output_query_string_parms module creates the CGIParms table. This table contains CGI parameter information that is discovered in the Web log. This module parses any CGI parameter (also referred to as the query string) or PATH_INFO parameters that are contained in the Web log records into name-value pairs.

Then the module writes each name-value pair to a SAS table that contains the Parameter_Name, Parameter_Value, and Record_ID fields. These fields can be used to map the name-value pair back to the original record of the Web log.

Note: The CGI parameter information for records that have been filtered by using the **Skip** or **Tally** value (of the Number_Of_CGI_Parms_Datasets property) is not written to the SAS tables. △

Output_pathinfo_parms

The Output_pathinfo_parms module creates the CGIParms table. This table contains the PATH_INFO parameter from the URL field of the Web log. This module parses the PATH_INFO parameter into name-value pairs. Each name-value pair is written to a

SAS table that contains the `Parameter_Name`, `Parameter_Value`, and `Record_ID` fields. These fields can be used to map the name-value pair back to the original record of the Web log.

Job_build__output_logic

The `Job_build__output_logic` module contains the logic to divide the requests into separate buckets based on the IP address or host name of the visitor. The incoming records are divided into the number of groups that are specified as the number of restructure buckets in SAS e-Data ETL Administrator. See “Number of Restructure Buckets” on page 52 for information about changing the number of restructure buckets.

Note: If you choose to edit the tables directly rather than using the SAS e-Data ETL Administrator interface, the number of restructure buckets is defined in the `Number_Of_Analysis_Buckets` property. △

Sort_buckets

Web server data is divided into SAS tables according to the value specified as the number of restructure buckets. See “Number of Restructure Buckets” on page 52 for information about changing the number of restructure buckets. The `Sort_buckets` module sorts each of these tables so that they can be merged with the corresponding buckets or tables that are created by any other concurrent processes.

Note: If you choose to edit the tables directly rather than using the SAS e-Data ETL Administrator interface, the number of restructure buckets is defined in the `Number_Of_Analysis_Buckets` property. △

Extract_initialization

The main function of the `Extract_initialization` module is to initialize the parallel SAS sessions. This preparation includes starting the parallel SAS sessions and copying any necessary programs or formats, such as the macros that are used to parse a visitor’s URL into its constituent pieces.

Execute_the_data_read_jobs

The `Execute_the_data_read_jobs` module starts the number of concurrent SAS sessions for reading the Web logs. The `Number_Of_Parallel_Reader_Jobs` property, which is set by using the SAS e-Data ETL Administrator, specifies the number of parallel SAS sessions (jobs) to be opened, unless there are fewer Web logs to be read than the specified number of parallel jobs.

Each session runs the generated SAS programs to read the Web logs. Each SAS session or reader job has a corresponding SAS program, `Read_jobn.sas` (which is located in a directory that is used for storing temporary files), and a corresponding log file, `Read_jobn.flex.log` (which is contained in the **saslogs** directory, a directory that is used for storing permanent results). The tables and the information that are particular to each job (1, 2, 3...*n*) are located in a directory named **Read_Jobn**, which is located in the temporary file directory (see “Overview: The Temporary Working Area” on page 95).

When each concurrent SAS session is completed, SAS logs are searched for any errors or warnings. Any condition that results in an error causes the Extract process to stop processing.

Aggregate_job_datasets

The `Aggregate_job_datasets` module joins all the restructure buckets together so that session information can be determined by the analysis jobs. This module also joins the `Job_Statistics` tables from each parallel SAS session and adds this information to the `Job_Statistics` table.

Execute_the_analysis_jobs

The `Execute_the_analysis_jobs` module operates similarly to the `Execute_the_data_read_jobs` module. However, in the `Execute_the_analysis_jobs` module, the number of concurrent SAS sessions is controlled by the `Number_Of_Parallel_Analysis_Jobs` property (rather than the `Number_Of_Parallel_Reader_Jobs` property). Unless the system that is running SAS e-Data ETL software has an extraordinary amount of memory, the value of this property will be less than the value of the `Number_Of_Parallel_Reader_Jobs` property.

The concurrent SAS sessions analyze the groups of SAS tables that contain the Web server information. Each SAS session that is run from this module is designated as an analysis job. Each session has a corresponding log file, **`Analysis_jobn.log`**, which is contained in the **`saslogs`** directory.

Each analysis job has its own directory by bucket. The tables are contained in the temporary file location directories named **`Analysis_Job1`** through **`Analysis_Job10`** (see “`Analysis_jobn.sas`” on page 95), assuming that the default setting for the `Number_Of_Analysis_Buckets` property is 10.

Note: In the SAS e-Data ETL Administrator, the `Number_of_analysis_buckets` property is specified as the number of restructure buckets. See “Number of Restructure Buckets” on page 52 for information about changing the number of restructure buckets that are specified in the SAS e-Data ETL Administrator. △

The `User_assignments_by_session_id` module is invoked during the execution of the `Execute_the_analysis_jobs` module immediately before each observation is output.

Aggregate_analysis_stats

The `Aggregate_analysis_stats` module joins job statistics information from each analysis job.

User_assignments_by_session_ID

The `User_assignments_by_session_ID` module is supplied in the event that you want to customize certain variables after the Web log records are sorted by `Session_ID` and `Datetime`.

The `User_assignments_by_session_id` module is invoked during the execution of the `Execute_the_analysis_jobs` module, which is immediately before each observation is written to the data sets.

The default code in the User_assignments_by_session_ID module follows:

```

*-----;
* User_Assignments_By_session_id ;
* ;
* Set variables based on SESSION_ID value and on the fact ;
* that the data is sorted by SESSION_ID at this point. ;
* This macro is invoked during the analysis jobs of Extract ;
* just before each observation is output. If pathing data sets ;
* are being created, this logic can also access the value ;
* of the variables in the PATH variable as it is being ;
* constructed. The observations at this point are grouped by ;
* Session_ID but the BY statement cannot be used, so use the ;
* the internal variables _lastSessionNum and _firstSessionNum ;
* to detect the start or end of a group of ;
* observations in the same session. For example: ;
* ;
* if _firstSesssionNum then do; ;
* * First click of session processing goes here; ;
* end; ;
* ;
* if _lastSesssionNum then do; ;
* * Last click of session processing goes here; ;
* end; ;
* ;
* Note that any variables that you have declared in ;
* WBFIELDS cannot be RETAINED in this code (they will have ;
* been established in earlier processing and will inherit a ;
* missing value in every observation), so if you ;
* want to retain values of one of these variables between ;
* successive observations, use a temporary variable name ;
* to retain the value and assign the WBFIELDS-defined ;
* name directly. For example, to make a sequence counter ;
* of the clicks in a session, you might code something ;
* like this (SESSION_SEQUENCE has been defined in WBFIELDS ;
* but TEMP_SESSION_SEQUENCE has not): ;
* ;
* retain temp_session_sequence 0; ;
* drop temp_session_sequence; ;
* if _firstSessionNum then ;
* temp_session_sequence = 0; ;
* temp_session_sequence = temp_session_sequence + 1; ;
* session_sequence = temp_session_sequence; ;
* ;
*-----+-----;
* History: | ;
* 01Jan2002 |S0127106 Update comment for reheading (df) ;
* 19Nov2003 |Added logic to handle pdf pages appropriately ;
* |cabahl C0001 ;
* 03Dec2003 |Added file size check C0002 ;
*-----;

/*-----REHEADING-----*/
/* Code statements that detect end or beginning of session here */

```

```

retain temp_session_sequence 0;
drop    temp_session_sequence;
if _firstSessionNum then
    temp_session_sequence = 0;
temp_session_sequence = temp_session_sequence + 1;
session_sequence = temp_session_sequence;
/*-----*/

*-----;
* C001: Many of the PDF files were coming up with multiple page_counts due ;
*     to partial loading (status code 206) or caching (status code 304-- ;
*     not modified). Typically the first hit would have ;
*     status code 200 and subsequent hits would have status code 206 ;
*     partial load) or status code 304 (cached/not modified). However, ;
*     sometimes the first hit would have status code 304 or 206. ;
* ;
*     To remedy this, set page count to 0 for PDF files with status code ;
*     304 or 206 if the previous record in the session is that same PDF ;
*     file. This will remove the subsequent hit from page view reports ;
*     but not from status code reports. C001 ;
*-----;
if length(requested_file) > 4 then do;
    if compress(session_id) = compress(lag(session_id)) and
        lowercase(compress(lag(requested_file)))=lowercase(compress(requested_file)) and
        substr(lowercase(requested_file),length(requested_file)- 3)='.pdf' and
        status_code in (206,304)
    then page_count=0;
end;

%mend user_assignments_by_session_id;

```

Check_sas_logs_for_metrics

The Check_sas_logs_for_metrics module examines the SAS log files from the concurrent SAS sessions and searches for information that is related to the amount of data that is processed during each SAS session. This information is then added to the performance information that is contained in the Job_Statistics table, which is located in the permanent results directory.

The Temporary Working Area

Overview: The Temporary Working Area

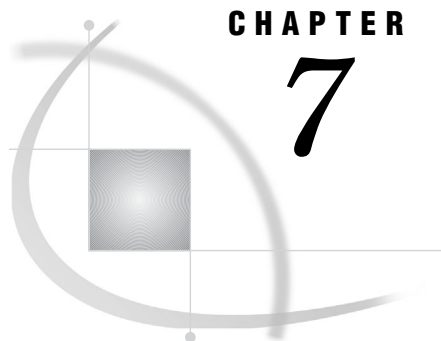
The temporary working area is populated by the Extract process and is used as input to the Load process. The temporary working area is used to store SAS tables and programs that are created and used during the execution of the Extract and the Load processes.

Read_jobn.sas

The Read_jobn.sas modules contain the SAS statements that are generated by the Extract process. They are used to read the Web logs. These programs are executed by the concurrent SAS sessions that have been started in the Execute_the_data_read_jobs module. The concurrent SAS sessions are started by using SAS MP/CONNECT. The concurrent SAS sessions are sometimes referred to as *child processes*. The Number_Of_Parallel_Reader_Jobs property determines the number of Read_jobn files.

Analysis_jobn.sas

The Analysis_jobn.sas modules contain the SAS statements that are generated by the Extract process in the analysis phase. These programs are executed by the concurrent SAS sessions that are started in the Execute_the_analysis_jobs module (see “Execute_the_analysis_jobs” on page 92). The concurrent SAS sessions are started by using SAS MP/CONNECT. The Number_Of_Parallel_Analysis_jobs property determines the number of Analysis_Jobn files.



CHAPTER

7

The SAS e-Data ETL Load Process

Overview of the Load Process 97

The Modules of the Load Process 97

The User_assignments_for_data_mining CAM 97

Running the Load Process 98

Overview of the Load Process

The Load process combines the Web server data that the Extract process reads with the existing data in the Web mart.

Note: The Load process permanently updates the Web mart. Make sure that the Web mart is backed up on a regular basis. △

When the Load process runs, it performs the following actions:

- creates new generations of the Weblog_Detail data sets and Pathing data sets
- creates the Unique_URLs data set

Note: By default, the number of generations specified for the Cookies data sets, the CGIParams data sets, and the ReferrerParams data sets is 0. Unless you change this default value, the Load process does not create generations of these data sets. △

With each invocation, the Load process updates some of the files in the **detail** directory.

Note: Save the contents of the subdirectories in the **detail** directory before each run of the Load process. If the Load process fails, then you can recover your detail information from your backup. △

Your Web mart's **checkpt** directory lists the contents of these directories from the last Load process. This information can help you determine which files were updated during a failed run of the Load process. However, you might prefer to use the backup directories to restore the entire contents of your **detail** directory.

The Modules of the Load Process

The modules of the Load process are described in “The Modules in the Load Process” on page 100. Except for the User_assignments_for_data_mining custom access module (CAM), you should not modify, customize, or override the modules in the Load process.

The User_assignments_for_data_mining CAM

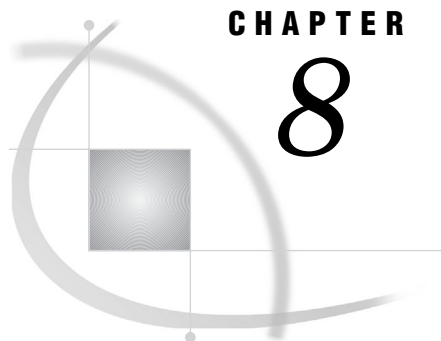
The User_assignments_for_data_mining CAM is specifically created to enable you to modify its code. This module prepares your Web log data for data mining by creating a

view called Detail.Web_Mine that contains all the current Web log detail data for page views. This module is executed automatically at the end of the Load process. In order to use data mining effectively, customize the code of this module to meet the needs of your organization.

Running the Load Process

The Load process is normally executed automatically within the Daily.sas program as part of the ETL processing of your Web log data. To run the Load process separately from the other ETL processes, invoke the %EDATAETL macro by typing **load** (in the SAS Program Editor window) as the argument of the PROGRAM parameter as follows:

```
%edataetl(  
  weblog_path=C:\Weblogs,  
  data_store=C:\Webmart,  
  work_area=C:\Webmart\Temp,  
  program=load);
```



CHAPTER

8

The Modules in the SAS e-Data ETL Load Process

<i>Overview: The Modules of the Load Process</i>	99
<i>The Modules in the Load Process</i>	100
<i>Checkpoint_load_init</i>	100
<i>Create_generations</i>	100
<i>Create_unique_urls</i>	100
<i>Clean_up_working_areas</i>	100
<i>User_assignments_for_data_mining</i>	100
<i>Checkpoint_load_term</i>	105
<i>Contents of the Detail Library</i>	105
<i>Overview: The Detail Library and Its Detail Tables</i>	105
<i>Weblog_Detail_n</i>	106
<i>CGIParms_n</i>	107
<i>ReferrerParms_n</i>	107
<i>Cookies_n</i>	107
<i>Pathing_n</i>	108
<i>\$ID2URL. Format Example</i>	109
<i>Unique_URL</i>	109

Overview: The Modules of the Load Process

The section entitled “The Modules in the Load Process” on page 100 describes the operation of each module in the Load process. Except for the *User_assignments_for_data_mining* module, you should not modify, customize, or override the modules in the Load process. Understanding the operation of each module is not necessary for general use, but it might be valuable when you troubleshoot the SAS e-Data ETL processes.

CAUTION:

The only module in the Load process that you can modify is the *User_assignments_for_data_mining* custom access module (CAM). △

For a description of the *User_assignments_for_data_mining* CAM, see “*User_assignments_for_data_mining*” on page 100. To modify the *User_assignments_for_data_mining* CAM, see “About Custom Access Modules (CAMs)” on page 53.

Note: For the modules that you can modify in the SAS e-Data ETL Extract process, see Chapter 6, “Reference Notes for SAS e-Data ETL Extract Process,” on page 79. △

The section “Contents of the Detail Library” on page 105 describes the detail tables that are created by the Load process.

The Modules in the Load Process

Checkpoint_load_init

The Checkpoint_load_init module creates a list of all the files in the Summary and Detail libraries as well as the contents of the Control library. This list is stored in the Web mart's **checkpt** directory before Load processing begins. You can use these lists to help recover your data if a system failure occurs during Load processing.

Create_generations

The Create_generations module creates new generations of the Weblog_Detail, CGIParams, Cookies, Pathing, and ReferrerParams tables in the Detail library. The generations of these tables are not bound to specific time spans. Each generation of a detail table corresponds to the amount of data that is processed during a single execution of the Load process.

Create_unique_urls

The Create_unique_urls module assigns a value to the Unique_URL property. This value is used only for the Pathing table. By default, the Requested_File URLs (with any parameters stripped off) are recorded for all page views in the Pathing table.

Clean_up_working_areas

The Clean_up_working_areas module removes all the temporary tables that are created in your Web mart's temporary working area. These tables are created during Extract processing to be used during the Load process.

User_assignments_for_data_mining

The User_assignments_for_data_mining module is executed automatically at the end of the Load processing. This module prepares your Web log data for data mining. It creates a view called Detail.Web_Mine that contains all the current Web log detail data for page views. To be effective for data mining, the code in this module needs to be customized to meet the needs of your site.

For the general instructions for modifying any CAM, see “About Custom Access Modules (CAMs)” on page 53.

The default code in the User_assignments_for_data_mining CAM follows:

```
/*=====
* User_Assignments_For_Data_Mining
*
* This section does setup of the weblog data for data mining.
* It is used to transform the fields of the input
* web log to make them most effective for data mining.
* Statements added here will be executed automatically at the end
* of Load processing.
*
*/
```

```

* The example coded here creates a view of all the current web log
* detail data for page views, including CGI parms and
* cookies and surfaces them in separate columns so they are ready
* for easy selection via SAS Enterprise Miner.
*
* You will probably want to change this behavior somewhat at your site.
* For example, you might want to use only a part of the REQUESTED_FILE
* or you might want to exclude some or all of the CGI parms or cookies
* (for best performance, if you do not intend to use any cookie or
* CGI parm values, you should comment out the code here that reads
* the CGI parm and/or cookie data set). If the number of CGI parms
* or cookies for any single line of web log exceeds the array size
* declared here, update the cgiparm_count_max, cookie_count_max,
* or referrer_parm_max values that appear below.
*
*-----*
*
* History:
* 27Aug2001 S0123412 Creation (df)
*
*=====*/

```

```
%Section_Header(Section=User_Assignments_For_Data_Mining) ;
```

```
* Create a view of weblog detail data with parms and cookies;
```

```
data    DETAIL.WEB_MINE
/ view=DETAIL.WEB_MINE ;
```

```

*-----;
* This view contains all the columns in WEBLOG_DETAIL plus arrays for      ;
* CGI parms, referrer parms, and cookies. Change the max values            ;
* for each of these if you have more values than will fit in the default.  ;
*-----;

%let  cgiparm_count_max      = 32;
%let  cookie_count_max      = 32;
%let  referrerparm_count_max = 32;

%put %qcmpres(%wbnote(USER_ASSIGNMENTS_FOR_DATA_MINING) A maximum of
           &cgiparm_count_max CGI parms for each request will be used for data mining.);
%put %qcmpres(%wbnote(USER_ASSIGNMENTS_FOR_DATA_MINING) A maximum of
           &referrerparm_count_max referrer parms for each request will be used for data mining.);
%put %qcmpres(%wbnote(USER_ASSIGNMENTS_FOR_DATA_MINING) A maximum of
           &cookie_count_max cookies for each request will be used for data mining.);

attrib cookies_read      label="Cookies read for this record";
attrib cookies_kept      label="Cookies kept for this record";
attrib cgiparms_read     label="CGI/PathInfo parms read for this record";
attrib cgiparms_kept     label="CGI/PathInfo parms kept for this record";
attrib referrerparms_read label="CGI/PathInfo parms read for this record";

```

```

attrib referrerparms_kept label="CGI/PathInfo parms kept for this record";

array cgiparm_name { &cgiparm_count_max } $64;
array cgiparm_value { &cgiparm_count_max } $1000;
array cgiparm_source{ &cgiparm_count_max } $1000;          * Valid values are
                                                           "Query_String" or
                                                           "Path_Info";

array referrerparm_name { &referrerparm_count_max } $64;
array referrerparm_value { &referrerparm_count_max } $1000;
array referrerparm_source{ &referrerparm_count_max } $1000; * Valid values are
                                                           "Referrer_Query_String" or
                                                           "Referrer_Path_Info";

array cookie_name { &cookie_count_max } $64;
array cookie_value { &cookie_count_max } $1000;

goto SKIP;
    * Supply declaration to avoid type conflict in case no parms or cookies;
    parameter_name = cgiparm_name{1};
    parameter_source = cgiparm_source{1};
    parameter_value = cgiparm_value{1};
SKIP:

*-----;
* Select_Table_by_Date reads all generations of WEBLOG_DETAIL      ;
*-----;

set %Select_Tables_by_Date( table          = weblog_detail
                             ,begin_date   = 1984-03-20
                             ,end_date     = 2084-03-20
                             ,in_variable_prefix = _in_detail_
                             ,where        = page_count gt 0 ) ;

*-----;
* Read CGI parms (names, values, and source) into arrays          ;
*-----;

_iorc_ = 0;
cgiparms_read = 0;
do while(_iorc_ eq 0);

    * Get_Parms sets PARAMETER_NAME, PARAMETER_VALUE, PARAMETER_SOURCE, and _IORC_;
    %Get_Parms( cgiparms, in_variable_prefix=_in_detail_ );

    * If _IORC_ is zero, parms were found;
    if _iorc_ eq 0 then do;
        cgiparms_read = cgiparms_read + 1;

    * Do not overflow array;
    if cgiparms_read gt hbound(cgiparm_name) then do;

```

```

        put "WARNING:(USER_ASSIGNMENTS_FOR_DATA_MINING) Unable to hold "
            parameter_source "parm " cgiparms_read
            parameter_name +(-1) "=" parameter_value
            " for Record_ID " record_id +(-1)
            ". Increase CGIPARM_COUNT_MAX from &cgiparm_count_max..";
    end;

    else do;
        * Save the values we just read;
        cgiparm_name{cgiparms_read} = parameter_name;
        cgiparm_value{cgiparms_read} = parameter_value;
        cgiparm_source{cgiparms_read} = parameter_source;
    end;
end;

_error_ = 0;

*-----;
* Read Referrer parms (names, values, and source) into arrays      ;
*-----;

_iorc_ = 0;
referrerparms_read = 0;
do while(_iorc_ eq 0);

    * Get_Parms sets PARAMETER_NAME, PARAMETER_VALUE, PARAMETER_SOURCE, and _IORC_;
    %Get_Parms( referrerparms, in_variable_prefix=_in_detail_ );

    * If _IORC_ is zero, parms were found;
    if _iorc_ eq 0 then do;
        referrerparms_read = referrerparms_read + 1;

        * Do not overflow array;
        if referrerparms_read gt hbound(referrerparm_name) then do;
            put "WARNING:(USER_ASSIGNMENTS_FOR_DATA_MINING) Unable to hold "
                parameter_source "parm " referrerparms_read
                parameter_name +(-1) "=" parameter_value
                " for Record_ID " record_id +(-1)
                ". Increase REFERRERPARM_COUNT_MAX from &referrerparm_count_max..";
        end;

    else do;
        * Save the values we just read;
        referrerparm_name{referrerparms_read} = parameter_name;
        referrerparm_value{referrerparms_read} = parameter_value;
        referrerparm_source{referrerparms_read} = parameter_source;
    end;
end;

end;

```

```

_error_ = 0;

*-----;
* Read cookies (names and values) into arrays          ;
*-----;

_iorc_ = 0;
cookies_read = 0;
do while(_iorc_ eq 0);

    * Get_Parms sets PARAMETER_NAME, PARAMETER_VALUE, PARAMETER_SOURCE, and _IORC_;
    %Get_Parms( cookies, in_variable_prefix=_in_detail_ );

    * If _IORC_ is zero, parms were found;
    if _iorc_ eq 0 then do;
        cookies_read = cookies_read + 1;

        * Do not overflow array;
        if cookies_read gt hbound(cookie_name) then do;
            put "WARNING:(USER_ASSIGNMENTS_FOR_DATA_MINING) Unable to hold "
                parameter_source "parm " cookies_read
                parameter_name +(-1) "=" parameter_value
                " for Record_ID " record_id +(-1)
                ". Increase COOKIE_COUNT_MAX from &cookie_count_max..";
            end;

        else do;
            * Save the values we just read;
            cookie_name{cookies_read} = parameter_name;
            cookie_value{cookies_read} = parameter_value;
        end;
    end;

end;
_error_ = 0;

*-----;
* Show how many cookies and parms we were able to keep      ;
*-----;

cgiparms_kept      = min(cgiparms_read,hbound(cgiparm_name));
referrerparms_kept = min(referrerparms_read,hbound(referrerparm_name));
cookies_kept       = min(cookies_read,hbound(cookie_name));

run;

%Section_Footer(Section=User_Assignments_For_Data_Mining) ;

```

Checkpoint_load_term

The Checkpoint_load_term module records the time of completion of the Load step. This information is recorded in the Job_Statistics table, which can be used to produce reports.

Contents of the Detail Library

Overview: The Detail Library and Its Detail Tables

If the Generation_Count property for each detail table (Weblog_Detail, CGIParams, Cookies, Pathing, and ReferrerParams) is set to keep one or more generations, then each time the Load process is executed, it creates a new generation of the table. When the Detail library contains the number of each type of table that is specified its associated Generation_Count property, the oldest table is deleted at the next execution of the Load process.

For more information about setting the number of generations for a detail table, see “The Detail Tables Properties” on page 31.

The following table shows the contents of the Detail library after the Load step has successfully run. The Load process creates the Weblog_Detail, CGIParams, ReferrerParams, Cookies, and Pathing tables only when the Generation_Count property is greater than zero. The Unique_URLs table exists only if the metadata value for the number of Pathing tables is greater than zero.

Note: The default value in the Generation_Count property for the CGIParams, ReferrerParams, and Cookies tables is zero. Therefore, by default, these tables will not be generated unless you change the number of generations to keep to a non-zero value. For information about how to change the number of history tables to keep, see “Calculating the Number of History Tables to Keep” on page 34. △

Table 8.1 Contents of the Detail Library after Running the Load Process

View	Definition
Weblog_Detail	Combines Weblog_Detail_1, Weblog_Detail_2, etc.
CGIParams	Combines CGI Params_1, CGI Params_2, etc.
ReferrerParams	Combines ReferrerParams_1, ReferrerParams_2, etc.
Cookies	Combines Cookies_1, Cookies_2, etc.
Pathing	Combines Pathing_1, Pathing_2, etc.
Unique_URLs	Used with the Pathing table to translate numbers to URLs.
Web_Mine	Combines Weblog_Detail for page views.

Weblog_Detail_n

After each run, the Load process creates a Weblog_Detail_n table that contains one observation for each line of Web log input. Those items that are specified in the Filters frame of the SAS e-Data ETL Administrator are excluded. (See “The Filtering Properties” on page 45 for more information.)

Weblog_Detail_1 is the newest table, Weblog_Detail_2 is the next newest table, and so on. The Weblog_Detail_n tables are sorted by Session_ID and Date_Time. A similarly named view with no numeric suffix, Detail.Weblog_Detail, interleaves all of the existing tables together, maintaining their sorted order.

The following fields can be included in the Weblog_Detail table.

Browser	Org	Session_Duration
Browser_Version	Org_Type	Session_ID
Bytes_Received	Page_Count	Session_Start
Bytes_Sent	Path	Sitename
Client_ID	Path_Info	State
Combined_URL	Platform	Status_Code
Cookie_Jar	Protocol	Time
Cookie_Present	Quarter	Time_Taken
Content_Group	Query_String	Total_Bytes
Country	Record_ID	Unique_URL
Date	Referrer	URL_Fragment
Date_Time	Referrer_Domain	URL_Level_1
Day_Of_Week	Referrer_Path_Info	URL_Level_2
Domain	Referrer_Query_String	URL_Level_3
Entry_Point	Referrer_URL_Level_1	URL_Level_4
Exit_Point	Referrer_URL_Level_2	URL_Level_5
Eyeball_Time	Referrer_URL_Level_3	URL_Level_6
File_Count	Referrer_URL_Level_4	URL_Level_7
File_Type	Referrer_URL_Level_5	URL_Level_8
GMT_Offset	Referrer_URL_Level_6	User_Agent
Hour	Referrer_URL_Level_7	Username
HTTP_Version	Referrer_URL_Level_8	View_EyeballTime
Initial_Reference_By_Session	Requested_File	Visitor_ID
Method	Scheme	WebLogFileName
MIME_Type	Server	Week
Month		

Some of these fields are not kept in the Weblog_Detail table, but are defined so that they can be added by changing their kept status from NO to YES in the Control.Wbfields

table. The Control.Wbfields table is accessed from the Processing frame of the SAS e-Data ETL Administrator. To learn how to view and edit the Control.Wbfields table, see “Adding or Deleting Fields in Your Web Server Log File” on page 38.

CGIParms_1

The CGIParms_1 table contains the name-value pairs that have been parsed from the query string in the Web log record. CGIParms_1 is the newest table, CGIParms_2 is the next newest table, and so on. The metadata for the table determines the number of tables that are created. The Detail.CGIParms view interleaves all the existing tables together, maintaining their sorted order.

You can use the Record_ID field to join an observation in the CGIParms_1 table with an observation in the corresponding Weblog_Detail_1 table. Combining these tables will give you a complete picture of the Web log record in which the parameter appeared. An example of the code that will join the CGIParms_1 table with the Weblog_Detail_1 table appears in the User_assignments_for_data_mining CAM.

The following fields are included in the CGIParms table:

Record_ID
Parameter
Value Indexed by Record_ID

ReferrerParms_1

The ReferrerParms_1 table contains the name-value pairs that are parsed from the Referrer field in the Web log record. ReferrerParms_1 is the newest table, ReferrerParms_2 is the next newest table, and so on. The metadata for the table determines the number of tables that are created. The Detail.ReferrerParms view interleaves all the existing tables together, maintaining their sorted order.

You can use the Record_ID field to join an observation in the ReferrerParms_1 table with an observation in the corresponding Weblog_Detail_1 table. Combining these tables will give you a complete picture of the Web log record in which the parameter appeared. An example of the code that will join the ReferrerParms_1 table with the Weblog_Detail_1 table appears in the User_assignments_for_data_mining CAM.

The following fields are included in the ReferrerParms table:

Record_ID
Parameter
Value Indexed by Record_ID

Cookies_1

The Load process creates a Cookies_1 table that contains Cookie name-value pairs and a record identifier. Cookies_1 is the newest table, Cookies_2 is the next newest table, and so on. The metadata for the table determines the number of tables that are

created. The Detail.Cookies view interleaves all the existing tables together, maintaining their sorted order.

You can use the Record_ID field to join an observation in the Cookies_1 table with an observation in the corresponding Weblog_Detail_1 table. Combining these tables gives you a complete picture of the Web log record in which the cookies appeared. An example of the code that joins the Cookies_1 table with the Weblog_Detail_1 table appears in the User_assignments_for_data_mining CAM.

The following fields are included in the Cookies table:

Record_ID
 Parameter
 Value Indexed by Record_ID

Pathing_1

The Pathing_1 table contains a session's path, duration, and datetime stamp. Pathing_1 is the newest table, Pathing_2 is the next newest table, and so on. These tables are sorted by Session_ID and Date_Time. The metadata for this table determines the number of tables that are created. The Detail.Pathing view interleaves all the existing tables together, maintaining their sorted order.

The value of the Path field is determined by the code in the Determine_unique_url module during the Extract process. (For more information about this module, see "Determine_unique_url" on page 88.) Each observation in the Path field contains a series of numbers that are separated by colons. These numbers can be mapped into their corresponding URL by using the ID2URL format or the \$ID2URL. format. The ID2URL format is defined for the number of Pathing tables that are returned. ID2URL is in Control.Formats, which is part of the FMTSEARCH path that the %EDATAETL macro establishes. You can use a similar format, \$URL2ID., to determine the ID of any URL in the Weblog_Detail table. (The variable name in that table is Requested_File.)

The following fields are included in the Pathing table:

Date_Time
 Session_ID
 Path
 Session_Duration

\$ID2URL. Format Example

In the Pathing table, the Path field consists of a string of numbers that are separated by colons. An example of an entry in the Path field is 42:95:3:101. Each of these numbers represents a page that is requested by the visitor to your Web site. You can use the \$ID2URL. format to map the number (or requested page) to the full pathname of the requested page.

The following code removes the right-most number from your Path field entry and converts it to the full pathname of the requested page:

```
node_label=put(trim(left(scan(node,-1,':'))), $id2url.);
```

By using this code in the entire Path string, you can resolve all these numbers to see the path that each visitor is traversing on your Web site.

Unique_URL

The Unique_URL tables are maintained by the Load step. They contain all the URLs that are referenced in the retained Pathing tables. The value of each unique URL is determined by the code in the Determine_unique_url module of the Extract process. For more information about this module, see “Determine_unique_url” on page 88.

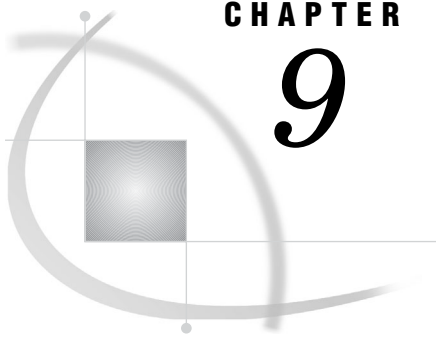
The Unique_URL tables are used to create the formats that map the numeric ID of the URL (in the Pathing table’s Path field) to an actual URL name, and vice versa. These formats, \$ID2URL. and \$URL2ID., are contained in Control.Formats.

The following fields are included in the Unique_URL tables:

Requested_File

Generation

URL_Guid



CHAPTER

9

Using the SAS Web Analytics Administrator to Manage Your Web Mart(s)

<i>Navigating in the SAS Web Analytics Administrator</i>	112
<i>The Interface of the SAS Web Analytics Administrator</i>	113
<i>Understanding Date Ranges in the Web Mart Status Table</i>	113
<i>The Web Marts Page</i>	115
<i>Editing a Web Mart in the SAS Web Analytics Administrator</i>	116
<i>Deleting a Web Mart in the SAS Web Analytics Administrator</i>	117
<i>Copying a Web Mart in the SAS Web Analytics Administrator</i>	117
<i>Updating a Web Mart</i>	118
<i>The Data Page</i>	119
<i>Customizing the Parameters that Control Summarization (Web Analytics Configuration)</i>	120
<i>Customizing the Graphical Components of Reports (User Interface Settings)</i>	121
<i>The Traffic Page</i>	123
<i>Creating an Item in the Traffic Page</i>	123
<i>Modifying, Copying or Deleting an Item in the Traffic Page</i>	124
<i>The Summaries Page</i>	125
<i>Working with Summaries</i>	125
<i>Creating a New Summarization</i>	126
<i>Modifying or Deleting a Summarization</i>	127
<i>The Scorecards Page</i>	127
<i>The Dashboards Page</i>	128
<i>The Segmentations Page</i>	129

Navigating in the SAS Web Analytics Administrator

The SAS Web Analytics Administrator is a Web-based interface that is designed to enable you to do the following:

- customize how the data is summarized into the summary data sets during ETL processing of your Web mart data
- customize the operation of the SAS Web Analytics Report Viewer in order to control how data appears in reports

The main menu of the Administrator consists of graphic links to the Web pages described in the following table:

Table 9.1 Main Menu Bar in the SAS Web Analytics Administrator

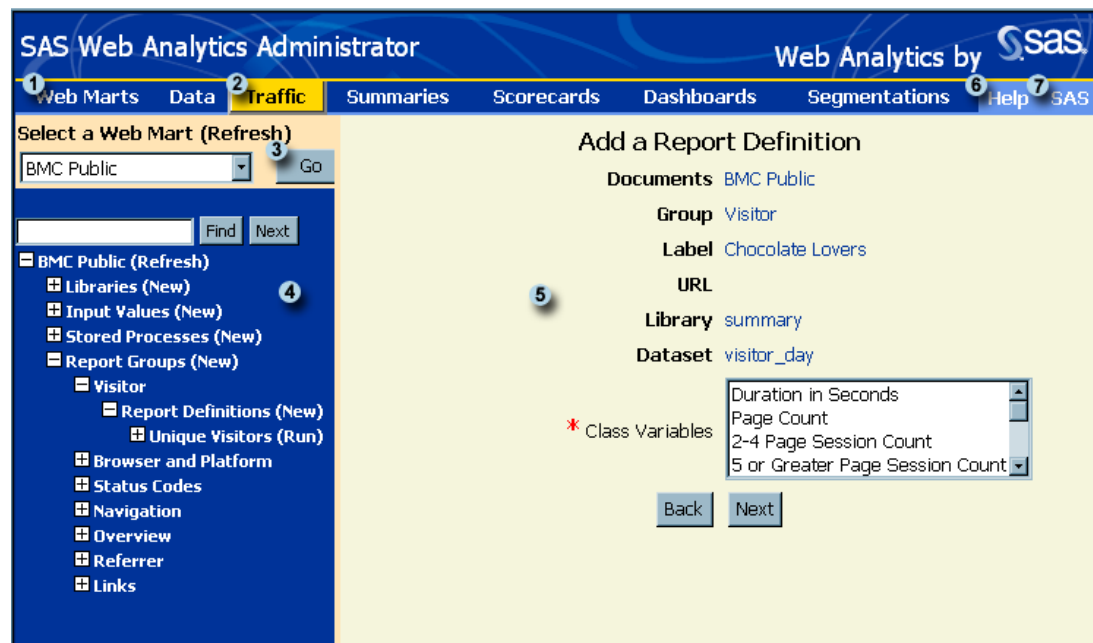
Page	Action
Web Marts	Register, edit or copy a Web mart. View the basic SAS server connections and the root of each of your Web marts.
Data	Edit the summarization and other processing parameters for Web site analysis (Web Analysis Configuration). Customize the appearance of reports (User Interface Settings).
Traffic	Add, modify, or delete libref assignments. Add, modify, or delete the data sets that are used to populate drop-down menus in reports. Add, modify, or delete the features of stored processes. Add, modify, or delete the features of report groups and individual reports.
Summaries	Customize the rules that control the summarization of the metrics into the summary data sets.
Scorecards	Create, edit, or delete a Scorecard.
Dashboards	Create, edit, or delete a Dashboard.
Segmentations	Create, edit, or delete a segmentation report.

The Interface of the SAS Web Analytics Administrator

The following display shows the components of the SAS Web Analytics Administrator. These components operate as follows:

- ❶ The main menu bar displays the links to the pages of the SAS Web Analytics Administrator.
- ❷ The title of the current selected page is yellow.
- ❸ A specific Web mart must be selected in order to view tree view items on any page of the SAS Web Analytics Administrator.
- ❹ The blue navigation area on the left displays a tree view of items.
- ❺ The right side of the page displays a form, a table, or other informational display that results from selecting an item in the tree view.
- ❻ **Help** is a link to the *SAS Web Analytics 5.1: Administrator's Guide*, a PDF document that you can view online, use for keyword searches, or print by using Adobe Reader 6.0 or a later version.
- ❼ **SAS** is a link to the SAS Home Page.

Display 9.1 Components of the SAS Web Analytics Administrator Interface



If you do not see a navigation tree, then select a Web mart from the drop-down list (❸ in the previous display).

Understanding Date Ranges in the Web Mart Status Table

As a convenient reference, the Web Mart Status table is initially displayed on the right of all of the pages of the SAS Web Analytics Administrator (except the Web Marts page). The Web Mart Status table appears after a Web mart is initially selected and remains until you select an item or action in the navigation tree. In the following display, the Web Mart Status table is shown in the Data page, for example.

Display 9.2 Web Mart Status Table on the Data Page of the SAS Web Analytics Administrator

Group	Id	Summary Level	First Date	Last Date
Visitor	Unique Visitors	Day	01/01/04	05/28/04
		Week	12/28/03	05/23/04
		Month	01/01/04	05/01/04
		Quarter	01/01/04	04/01/04
		Year	01/01/04	01/01/04
	Browsers	Day	01/01/04	05/28/04
		Week	12/28/03	05/23/04
		Month	01/01/04	05/01/04
		Quarter	01/01/04	04/01/04
		Year	01/01/04	01/01/04
Browser and Platform	Browser Versions	Day	01/01/04	05/28/04
		Week	12/28/03	05/23/04
		Month	01/01/04	05/01/04
		Quarter	01/01/04	04/01/04
		Year	01/01/04	01/01/04

The Web Mart Status table shows the date ranges that have been processed for each of the corresponding summary levels (day, week, month, quarter, and year). The first date and last date for the different summary levels are not identical, because each summary level represents a unit of time that cannot be subdivided. Each date, regardless of whether it is listed in the First Date or the Last Date column, represents the beginning date of its corresponding interval. For example, the month interval for May 2004 is labeled as 05/01/04.

Examine the first date and last date of the summary levels for the Unique Visitors report in the example Web Mart Status table. The first date of the **Day** summary level is 01/01/2004, and the last date is 05/28/2004. These dates indicate that Web mart data has been processed for the 24-hour intervals (days) starting on January 1, 2004 and ending on (and including) May 28, 2004.

The first date of the **Week** summary level for Unique Visitors is 12/28/2003, assuming that the first day of a week begins on Sunday. December 28th is the start day of the specific week that includes the first day that data was actually processed (01/01/2004). Data was processed through Friday, May 28th, 2004, but the last date of the Week summary level appears as 05/23/04 because the first day (Sunday) of that week interval occurs on 05/23/04.










Note: The summary of the data for the last week in May 2004 is a partial summary because the Web log(s) for Saturday, May 29, 2004 have not been processed. \triangle

The first date of the **Month** summary level is 01/01/04. The last date for the **Month** summary level is 05/01/04 because May 1, 2004 is the first day of the last month for which Web logs were processed.

The Web Mart Status table specifies the dates for the **Quarter** and the **Year** summary levels in a similar way.

The Web Marts Page

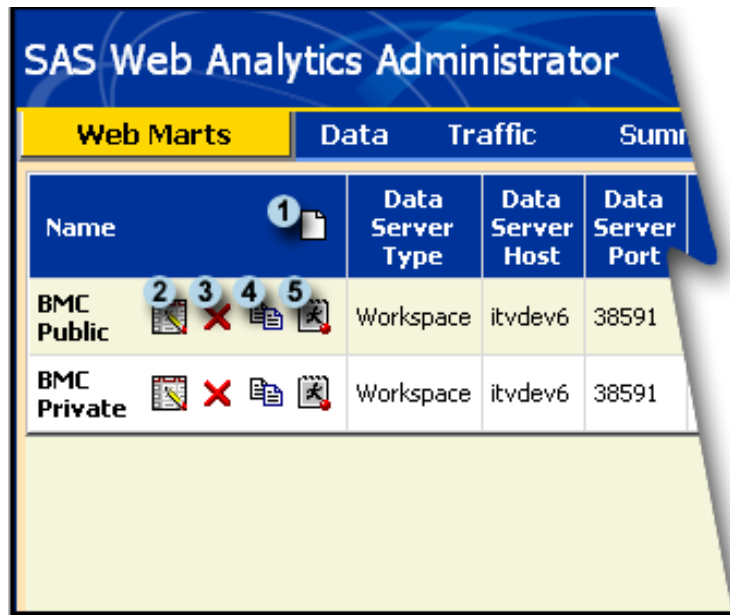
Display 9.3 The Web Marts Page of the SAS Web Analytics Administrator

SAS Web Analytics Administrator										Web Analytics by 	
Web Marts		Data	Traffic	Summaries	Scorecards	Dashboards	Segmentations	Help	SAS		
Name		Data Server Type	Data Server Host	Data Server Port	Web Mart Root	Reports Path	User	Stored Process Host	Stored Process Port	Stored Process User	
BMC Public	   	Workspace	itvdev6	38591	d:\swa\prod\bmc\swamart-pub	d:\swa\prod\bmc\swamart-pub\data\documents	sasguest	itvdev6	38601	sasguest	
BMC Private	   	Workspace	itvdev6	38591	d:\swa\prod\bmc\swamart	d:\swa\prod\bmc\swamart\data\documents	sasguest	itvdev6	38601	sasguest	

The Web Marts page of the SAS Web Analytics Administrator (see the previous display) displays system information about all of the Web marts that have been registered. You can also do the following tasks (see Display 9.4 on page 116):

- ❶ register a new Web mart (for details, see “Registering a New Web Mart” on page 24)
- ❷ edit the Web mart (that is in the same row as the icon)
- ❸ delete the Web mart
- ❹ copy the Web Mart
- ❺ generate the status of the Web mart

Display 9.4 Task Icons in the Web Marts Page



Editing a Web Mart in the SAS Web Analytics Administrator

The Edit a Web Mart form (Figure 9.1 on page 117) enables you to edit the environmental characteristics of a Web mart.

Note: To edit all of the other properties of a Web mart, use the SAS e-Data ETL Administrator (see “Overview: The Properties of a SAS Web Analytics Web Mart” on page 28). △

To edit the environmental characteristics of a Web mart, follow these steps:


- 1 In the Web Marts page, click  next to the Web mart you want to edit. The Edit a Web Mart form appears.
- 2 Enter changes to any fields as necessary. For assistance, consult with the authorities in your organization who are familiar with the configuration of the SAS solution on your server(s). When you are finished, click **Submit**.

Figure 9.1 The Edit a Web Mart Form

SAS Web Analytics Administrator Web Analytics by SAS

Web Marts Data Traffic Summaries Scorecards Dashboards Segmentations Help SAS

Edit a Web Mart

Name

Data Server Type

Data Server Host

Data Server Port

Web Mart Root

Reports Path

User

Password


Stored Process Host

Stored Process Port

Stored Process User

Stored Process Password

Deleting a Web Mart in the SAS Web Analytics Administrator


To delete a Web Mart, select the Web Marts page of the SAS Web Analytics Administrator. Click  in the first column next to the name of the Web mart to delete it.

Copying a Web Mart in the SAS Web Analytics Administrator

You might want to copy an existing Web mart for one of the following reasons:

- As an administrator, you want to enable different communities of end users to quickly access the specific SAS Web Analytics reports that relate to their work.
- You have a highly customized Web mart and you want to use these settings in a new Web mart.


In the first scenario, you might copy a Web mart in order to provide a custom subset of reports. You would provide this subset of reports as a separate Web mart in the SAS Web Analytics Report Viewer. In this way, the data for the customized subset of reports is available without duplicating the ETL processing of the original Web mart data.

The copy () action in the Web Marts page of the SAS Web Analytics Administrator duplicates the values for the fields of an existing Web mart, but does not create new directories.

CAUTION:

Before you copy a Web mart in the SAS Web Analytics Administrator, all of the destination directory structures must previously exist. Either manually copy and paste the Web mart directory and all its subdirectories to a new location in the Windows environment, or run the %WAETL macro with the INITIALIZE argument for the PROGRAM parameter. △


To make a copy of a Web mart (assuming that a directory structure already exists for the new Web mart copy), follow these steps:

- 1 In the Web Marts page, click  next to the Web mart that you want to copy. The Copy a Web Mart form appears. Except for the title, the Copy a Web Mart form is identical to the Edit a Web Mart form (Figure 9.1 on page 117).
- 2 In the Name field, the words “Copy of” are automatically added to the old Web mart name. Modify the name as you desire.
- 3 Modify the other values as necessary. For assistance, consult with the authorities in your organization who are familiar with the configuration of the SAS solution on your server(s). When you are finished, click **Submit**.


Updating a Web Mart

You might need to update a Web mart to create a provisional Web Mart Status table or to re-establish a Web Mart Status table that cannot be read.

When you select a Web mart, the interface (of either the SAS Web Analytics Administrator or Report Viewer) must access the Web Mart Status table. This table is routinely created during ETL processing by the Daily.sas program. However, if you want to select a Web mart in the SAS Web Analytics interface before you run the ETL

processes, then the  icon enables you to create a provisional version of the Web Mart Status table for the interface to access.

To create a provisional Web Mart Status table or to re-establish a Web Mart Status

table, click  next to the Web mart that you want to update. After a few moments, a message verifies that the Web Mart Status table was successfully generated. Click **Continue**. You are returned to the Web Marts page.

The Data Page

Display 9.5 The Data Page of the SAS Web Analytics Administrator

Group	Id	Summary Level	First Date	Last Date
Visitor	Unique Visitors	Day	01/01/04	05/28/04
		Week	12/28/03	05/23/04
		Month	01/01/04	05/01/04
		Quarter	01/01/04	04/01/04
		Year	01/01/04	01/01/04
	Browsers	Day	01/01/04	05/28/04
		Week	12/28/03	05/23/04
		Month	01/01/04	05/01/04
		Quarter	01/01/04	04/01/04
		Year	01/01/04	01/01/04
Browser and Platform	Browser Versions	Day	01/01/04	05/28/04
		Week	12/28/03	05/23/04
		Month	01/01/04	05/01/04
		Quarter	01/01/04	04/01/04
		Year	01/01/04	01/01/04

The Data page enables you to customize the parameters in the following areas:

Web Analytics Configuration	configures the settings for summarizing the data for a selected Web mart.
User Interface Settings	sets the graphical components for decision support reports and a parameter for the funnel report for a selected Web mart.

Customizing the Parameters that Control Summarization (Web Analytics Configuration)

Display 9.6 The Table of Parameters that Control Summarization (WaconfigTable)


ID for metadata maintenance	Value	Description	Parameter Name
1	10	The minimum count of total sessions for path for 1 day.	wab_r1_support
2	40	The minimum count of total sessions for path for 7 day.	wab_r7_support
3	100	The minimum count of total sessions for path for 30 days.	wab_r30_support
4	7	The maximum number of URLs/Pages for a path.	wab_itemset_size
5	&temp_lib..wasumeng	Summary Engine WASUMENG metadata which drives the creation of the proc summary code to build the descriptive summaries in the warehouse.	wab_metadsn
6	config.wadmsum	Summary Engine Administration WAADMSUM metadata which is the source for the WASUMENG metadata.	wab_admsum
7	90	The number of days available in daily summaries	wab_days_in_history
8	52	The number of weeks available in weekly summaries	wab_weeks_in_history
9	36	The number of months available in monthly summaries	wab_months_in_history
10	12	The number of quarters available in quarterly summaries	wab_quarters_in_history
11	3	The number of years available in yearly summaries	wab_years_in_history
12	7	The number of separate date partitioned detail data sets to keep.	wab_num_detail_data_sets
13	30	The number of separate date partitioned session data sets to keep.	wab_num_session_data_sets
14	90	The number of days of history that is retained in SUMMARY.VISITOR.DAY.	wab_visitor_days_in_history

The parameters that control the summarization of the detail data sets into summary data sets are located in the Waconfig table. Explore the fields in the Waconfig table to become familiar with the ways that you can modify these parameters to affect how information is summarized in SAS Web Analytics reports.

To access the Waconfig table, follow these steps:

- 1 Select **Data** in the main menu bar in the SAS Web Analytics Administrator. If a Web mart is not already selected, then select a Web mart and click **Go**. The **Configuration Tables** option appears in the navigation area.
- 2 Select **Configuration Tables ► Web Analytics Configuration**. The Waconfig table appears on the right.

To edit a parameter in the Waconfig table, follow these steps:

- 1 Click  in the row of the parameter that you want to edit. A Metadata Administration form appears on the right, similar to the following display:

The screenshot shows the SAS Web Analytics Administrator interface. The top navigation bar includes 'Web Marts', 'Data' (selected), 'Traffic', 'Summaries', 'Scorecards', 'Dashboards', and 'Segmentations'. On the left, there is a 'Select a Web Mart (Refresh)' dropdown menu with 'BMC Public' selected and a 'Go' button. Below this is a 'Configuration Tables' link. The main area is titled 'Metadata Administration Form' and contains three input fields: 'Value' with '40', 'Description' with 'The minimum count of total sessions for path for 7 day.', and 'Parameter Name' with 'wab_r7_support'. At the bottom right are 'Submit' and 'Cancel' buttons.

- 2 Modify the values for one or more fields as you wish. When you are finished, click **Submit**.
- 3 Click **Continue** below the confirming message that appears on the right. The table reappears.
- 4 Repeat these steps to edit additional parameters.

Customizing the Graphical Components of Reports (User Interface Settings)

Display 9.7 Table for Customizing User Interface Settings

The screenshot shows the SAS Web Analytics Administrator interface with the 'Configuration Table: waresrcr' displayed. The table has three columns: 'Web Analytics Configuration', 'Value/Setting of feature', and 'Description of feature'. The 'User Interface Settings' row is highlighted. Below it are several rows for dashboard performance background and foreground colors, and dashboard image icons.


Web Analytics Configuration	Value/Setting of feature	Description of feature
User Interface Settings	5000	# of Obs at which user is prompted to continue a report
dashboard_down_bg	#ff0033	Dashboard negative performance background color
dashboard_down_fg	#ffffff	Dashboard negative performance foreground color
dashboard_up_bg	#33cc00	Dashboard positive performance background color
dashboard_up_fg	#ffffff	Dashboard positive performance foreground color
dashboard_steady_bg	#0000ff	Dashboard steady performance background color
dashboard_steady_fg	#ffffff	Dashboard steady performance foreground color
dashboard_ok_image	../Images/WithinLimits.gif	Dashboard steady performance image icon
dashboard_belowgood_image	../Images/BelowGood.gif	Dashboard trend down - desired trend down image icon
dashboard_belowbad_image	../Images/BelowBad.gif	Dashboard trend down - desired trend up image icon
dashboard_abovebad_image	../Images/AboveBad.gif	Dashboard trend up - desired trend down image icon
dashboard_abovegood_image	../Images/AboveGood.gif	Dashboard trend up - desired trend up image icon

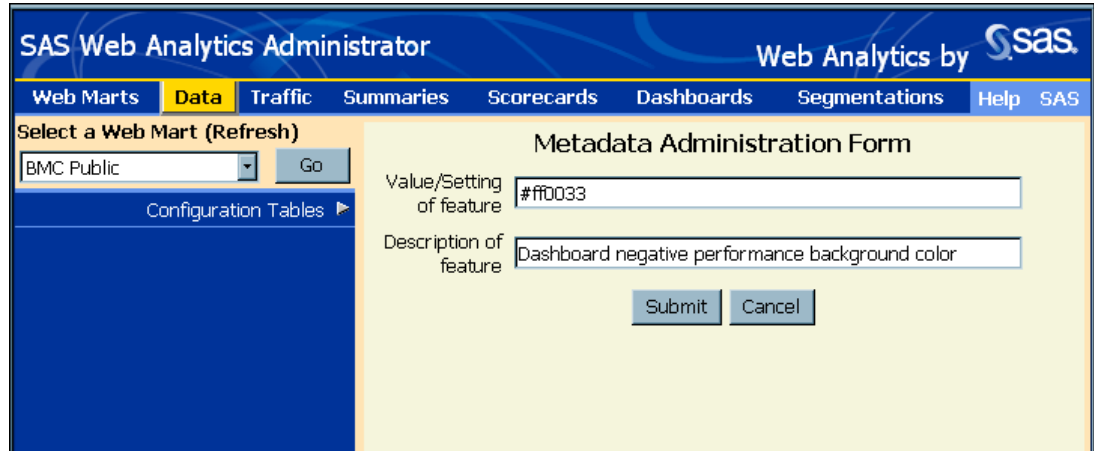
The parameters that control the appearance of graphical components in several types of reports are located in the Waresrcr configuration table (see the previous display). Explore the fields in this configuration table to become familiar with the ways that you can modify how information is displayed in some types of SAS Web Analytics reports.

To access the Waresrcr configuration table, follow these steps:

- 1 Select **Data** in the main menu bar in the SAS Web Analytics Administrator. If a Web mart is not already selected, then select a Web mart and click **Go**. The **Configuration Tables** option appears in the navigation area.
- 2 Select **Configuration Tables ► User Interface Settings**. The Waresrcr table appears on the right.

To edit a parameter in the table, follow these steps:

- 1 Click  in the row of the parameter that you want to edit. A Metadata Administration form appears on the right, similar to the following display:



- 2 Modify the values for one or more fields as you wish. When you are finished, click **Submit**.
- 3 Click **Continue** below the confirming message that appears on the right. The table reappears.
- 4 Repeat these steps to edit additional parameters.

The Traffic Page

Display 9.8 The Traffic Page of the SAS Web Analytics Administrator



The Traffic page of the SAS Web Analytics Administrator enables you to do the following tasks:

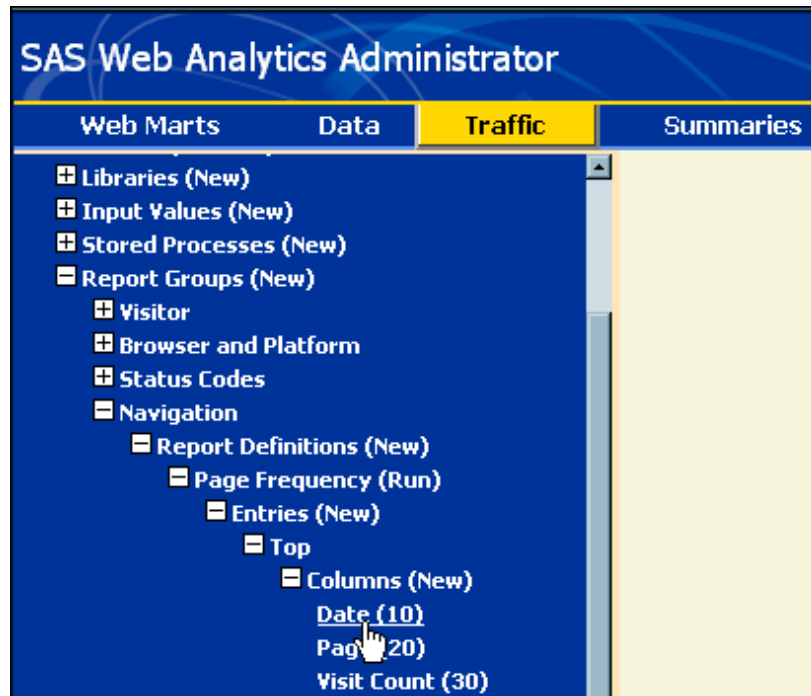
- ☐ assign, modify, copy and delete a library for SAS Web Analytics entries
- ☐ add a column (input value) to a selected traffic report
- ☐ create, modify, or run a stored process for a funnel report
- ☐ create, customize, or delete report groups for viewing by end users in the SAS Web Analytics Report Viewer
- ☐ create, customize, or delete reports (excluding Dashboards, Scorecards, and segmentation reports) for viewing by end users in the SAS Web Analytics Report Viewer

Creating an Item in the Traffic Page

To create an item (library, input value, stored process, report, or report group) in the Traffic page, follow these general steps:

- 1 Expand the navigation tree by clicking **+** next to an item to view a component of the item. Depending on the component, you might need to expand several levels in

order to expose the component you want to create (for example, see the following display):



- 2 Select New (in parentheses) next to the component that you wish to create.
- 3 Edit the fields in the form that appears on the right.
- 4 Click **Submit**, and click **Continue**.

Modifying, Copying or Deleting an Item in the Traffic Page

To modify, delete, or copy an item in the Traffic page, follow these general steps:

- 1 Expand the navigation tree to view the item or the component of the item. Depending on the item, you might need to click several items in order to expose the desired component.
- 2 Select the item or component that you want to modify, delete or copy by clicking it.
- 3 In the table that appears on the right, select **Modify**, **Delete** or **Copy**.
- 4 Provide the information that is required by the action. When you are finished, click **Submit**, and click **Continue**.
- 5 To verify any modifications that you made, select **Refresh** next to **Select a Web Mart**.

The Summaries Page

Display 9.9 The Summaries Page of the SAS Web Analytics Administrator

Web Mart Status					
Group	Id	Summary Level	First Date	Last Date	
Visitor	Unique Visitors	Day	01/01/04	05/28/04	
		Week	12/28/03	05/23/04	
		Month	01/01/04	05/01/04	
		Quarter	01/01/04	04/01/04	
		Year	01/01/04	01/01/04	
Browser and Platform	Browsers	Day	01/01/04	05/28/04	
		Week	12/28/03	05/23/04	
		Month	01/01/04	05/01/04	
		Quarter	01/01/04	04/01/04	
		Year	01/01/04	01/01/04	
	Browser Versions	Day	01/01/04	05/28/04	
		Week	12/28/03	05/23/04	
		Month	01/01/04	05/01/04	
		Quarter	01/01/04	04/01/04	
		Year	01/01/04	01/01/04	
		Day	01/01/04	05/28/04	

In the Summaries page of the SAS Web Analytics Administrator, you can create, modify, or delete data summarizations. The initial display on the right of the Summaries page is the Web Mart Status table. The Web Mart Status table displays the date ranges for which the Web log data has been processed for the selected Web mart. When you select one of the summaries in the navigation tree on the left of the Summaries page, the Summary window opens.

Working with Summaries

The Summary page uses the following conventions:

Required fields

Fields in a form that are labeled with a red asterisk (*) are required fields. You must supply information for a required field in order to continue a task.

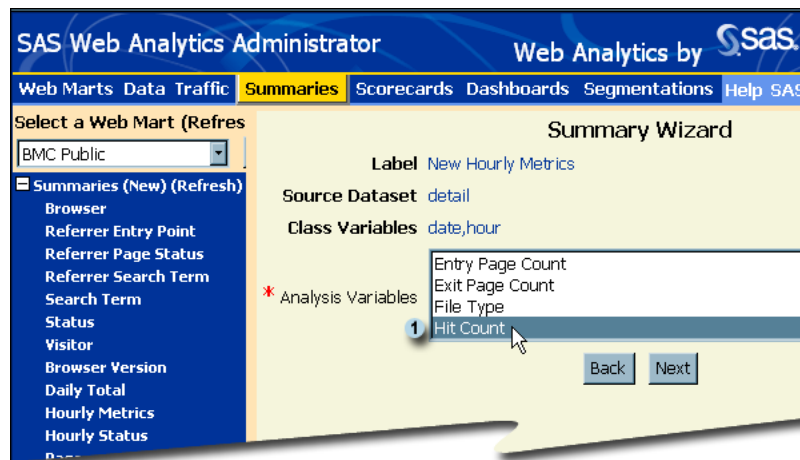
Drop-down menus

Use the standard Windows keyboard conventions to select contiguous or noncontiguous items from a drop-down menu.

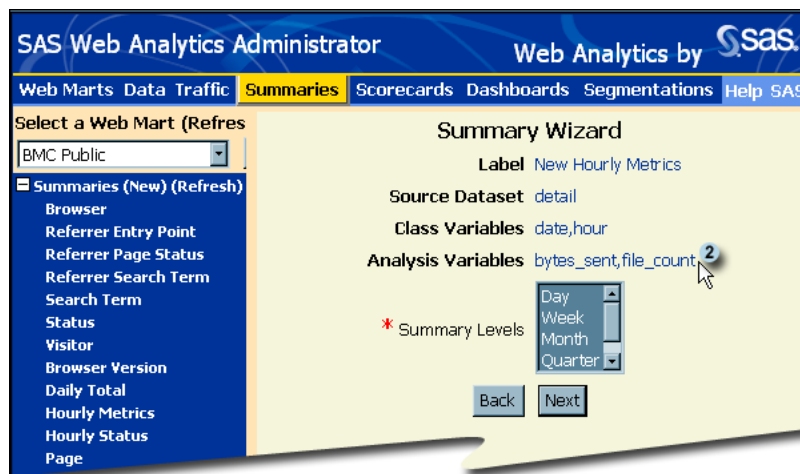
Variable labels and variable names

In the Summary Wizard, a drop-down menu (of the variables to select) shows the variable labels rather than the variable names. When the Summary Wizard confirms your variable choices, however, it displays the variable name(s) that correspond to the variable names in the Waadmsum data set. For example, **Hit**

Count (❶) is an analysis variable label that is selected from the drop-down menu in the following display:



In contrast, the corresponding variable name that is displayed in the confirming text is **file_count** (❷) in the following display:



Creating a New Summarization

Before creating a new summarization family, you need to clearly understand the metrics that you want to summarize and how to implement your new summarization family. It can be useful to write a trial PROC SUMMARY step that is modeled on one of the examples presented in “General Information About SAS Web Analytics Summaries” on page 195. To create a new summarization, follow these general steps:

- 1 Expand the navigation tree to view the item or the component of the item. Depending on the item, you might need to make several clicks in order to expose the desired component.
- 2 Select **New** (in parentheses) next to **Summaries**.
- 3 Supply the fields (at least those fields that are required) in the form that appears on the right.
- 4 Click **Submit**. Click **Continue**.

Modifying or Deleting a Summarization

To modify or delete a summarization, follow these general steps:

- 1 Expand the navigation tree to view the summarization.
- 2 Select the summarization that you want to modify or delete by clicking it.
- 3 In the table that appears on the right, select **Modify** or **Delete**.
- 4 If you are modifying a summarization, then either select the variables you want to modify from a drop-down menu or type the new information into a text box. When you are finished, click **Update**, and click **Continue**. If you are deleting a summarization, then select the appropriate buttons to continue.
- 5 To verify any modifications that you made, select **Refresh** next to **Select a Web Mart**.

The Scorecards Page

Display 9.10 The Scorecards Page of the SAS Web Analytics Administrator

Web Mart Status			
Group	Id	Summary Level	First Date
Visitor	Unique Visitors	Day	01/01/04
		Week	12/28/03
		Month	01/01/04
		Quarter	01/01/04
		Year	01/01/04
	Browsers	Day	01/01/04
		Week	12/28/03
		Month	01/01/04
		Quarter	01/01/04
		Year	01/01/04
Browser and Platform	Browser Versions	Day	01/01/04
		Week	12/28/03
		Month	01/01/04
		Quarter	01/01/04
		Year	01/01/04
		Day	01/01/04

In the Scorecards page of the SAS Web Analytics Administrator (see the previous display), you can create, modify or delete Scorecards. After you select a Web mart, the Scorecards page displays the Web Mart Status Table on the right. The Web Mart Status table displays the date ranges for which the Web log data has been processed for the selected Web mart. For more about the dates in this table, see “Understanding Date Ranges in the Web Mart Status Table” on page 113.

To customize data for Scorecards, see “Overview: Scorecard” on page 149.

The Dashboards Page

Display 9.11 The Dashboards Page of the SAS Web Analytics Administrator

SAS Web Analytics Administrator

Web Analytics by SAS

Web MartsDataTrafficSummariesScorecardsDashboardsSegmentationsHelpSAS

Select a Web Mart (Refresh)

BMC PublicGo

Dashboards (New) (Refresh)

Site Metrics (Run)

Metrics (New) (Category)

Site Health

302 Count: Found

304 Count: Not Modified

400 Count: Bad Request

401 Count: Unauthorized

403 Count: Forbidden

404 Count: Not Found

405 Count: Method Not Allowed

408 Count: Request Time-out

500 Count: Internal Server Error

501 Count: Not Implemented

Traffic

Hit Count

One Hit Visits

Page Count

Visit Count

Duration in Seconds

2-4 Page Visit Count

0-1 Page Visit Count

>= 5 Page Visit Count

Total Bytes Sent

Web Mart Status

Group	Id	Summary Level	First Date	Last Date
Visitor	Unique Visitors	Day	01/01/04	05/28/04
		Week	12/28/03	05/23/04
		Month	01/01/04	05/01/04
		Quarter	01/01/04	04/01/04
		Year	01/01/04	01/01/04
Browser and Platform	Browsers	Day	01/01/04	05/28/04
		Week	12/28/03	05/23/04
		Month	01/01/04	05/01/04
		Quarter	01/01/04	04/01/04
		Year	01/01/04	01/01/04
	Browser Versions	Day	01/01/04	05/28/04
		Week	12/28/03	05/23/04
		Month	01/01/04	05/01/04
		Quarter	01/01/04	04/01/04
		Year	01/01/04	01/01/04
	Platforms	Day	01/01/04	05/28/04
		Week	12/28/03	05/23/04
Month		01/01/04	05/01/04	
Quarter		01/01/04	04/01/04	

In the Dashboards page of the SAS Web Analytics Administrator (see the previous display), you can create, modify or delete Dashboards. After you select a Web mart, the Dashboards page displays the Web Mart Status table on the right. The Web Mart Status table displays the date ranges for which the Web log data has been processed for the selected Web mart. For more about the dates in this table, see “Understanding Date Ranges in the Web Mart Status Table” on page 113.

To customize data for Dashboards, see “Overview: Dashboard” on page 163.

The Segmentations Page

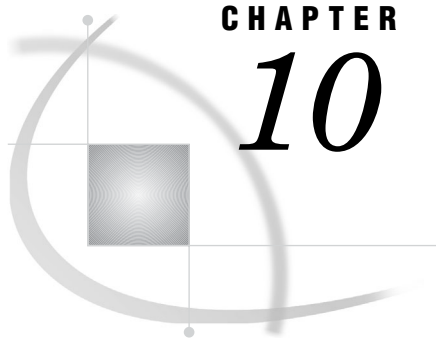
Display 9.12 The Segmentations Page of the SAS Web Analytics Administrator

The screenshot shows the SAS Web Analytics Administrator interface. The top navigation bar includes 'Web Marts', 'Data', 'Traffic', 'Summaries', 'Scorecards', 'Dashboards', 'Segmentations' (highlighted), 'Help', and 'SAS'. Below the navigation bar, there's a 'Select a Web Mart (Refresh)' dropdown menu with 'BMC Public' selected and a 'Go' button. The left sidebar contains a tree view with 'Segmentations (New) (Refresh)' and 'Repeat Visitor - Totals (Run)' expanded, showing various metrics like 'Repeat Visitor (Target)', 'New Visitor (New Visitor)', 'Most Current Visitor (Most Current Visitor)', 'Visitor Id (Visitor Id)', 'Page Count Analytic Period', 'Visit Count Analytic Period', 'Duration Analytic Period', 'Repeat Visitor - Averages (Run)', and 'Metrics (New)'. The main content area displays the 'Web Mart Status' table.

Group	Id	Summary Level	First Date	Last Date
Visitor	Unique Visitors	Day	01/01/04	05/28/04
		Week	12/28/03	05/23/04
		Month	01/01/04	05/01/04
		Quarter	01/01/04	04/01/04
		Year	01/01/04	01/01/04
Browser and Platform	Browsers	Day	01/01/04	05/28/04
		Week	12/28/03	05/23/04
		Month	01/01/04	05/01/04
		Quarter	01/01/04	04/01/04
	Browser Versions	Day	01/01/04	05/28/04
		Week	12/28/03	05/23/04
		Month	01/01/04	05/01/04
		Quarter	01/01/04	04/01/04
	Platforms	Day	01/01/04	05/28/04
		Week	12/28/03	05/23/04

In the Segmentations page of the SAS Web Analytics Administrator (see the previous display), you can create, modify or delete segmentation reports. After you select a Web mart, the Segmentations page displays the Web Mart Status table on the right. The Web Mart Status table displays the date ranges for which the Web log data has been processed for the selected Web mart. For more about the dates in this table, see “Understanding Date Ranges in the Web Mart Status Table” on page 113.

To customize data for segmentation reports, see Chapter 13, “Setting Up a Segmentation Report,” on page 171.



CHAPTER

10

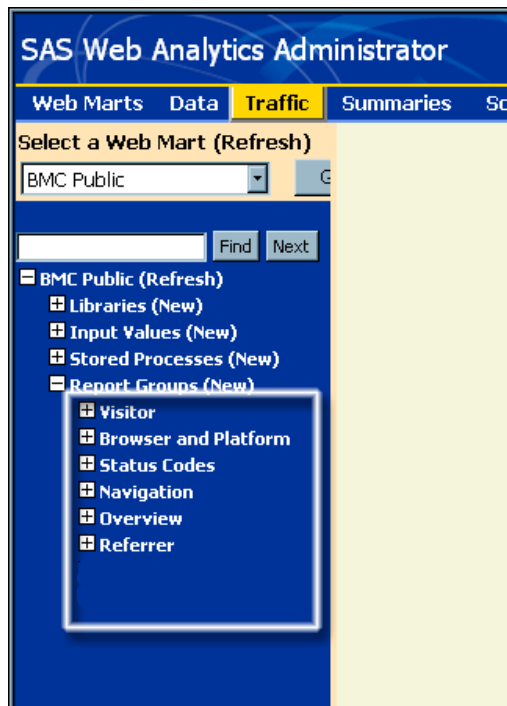
Administration of SAS Web Analytics Reports

<i>Working with Report Groups</i>	131
<i>What Is a Report Group?</i>	131
<i>How to Create a New Report Group</i>	133
<i>How to Modify a Report Group</i>	135
<i>How to Delete a Report Group</i>	136
<i>Creating a New Report</i>	137
<i>Refining a Report</i>	139
<i>Drillable Reports</i>	144
<i>Overview: Drillable Reports</i>	144
<i>Examples of Drillable Reports</i>	144
<i>Specifying the Value for the Link Field for a Drillable Report</i>	145
<i>Adding a Filter to the Drilled Report</i>	146
<i>Variables You Can Set for Reports</i>	146
<i>Using Variables as Filter Values</i>	146
<i>Using Variables as Hyperlinks</i>	147

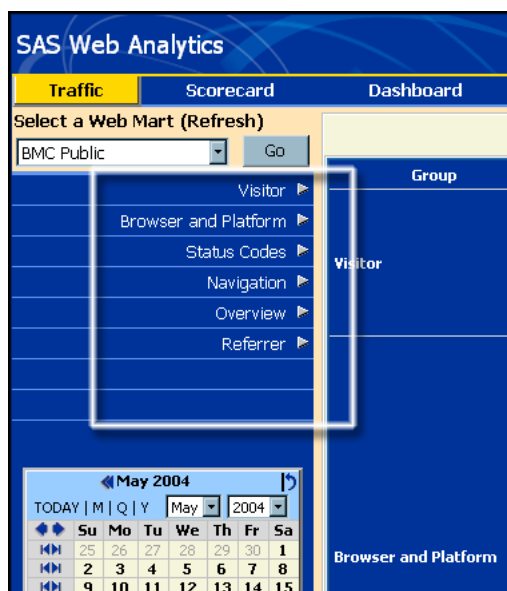
Working with Report Groups

What Is a Report Group?

A report group is a category in the SAS Web Analytics Administrator to which you can assign related reports. The default report groups that are located on the Traffic page of the SAS Web Analytics Administrator are shown in the following display:

Display 10.1 Default Report Groups in the SAS Web Analytics Administrator


The report groups in the SAS Web Analytics Administrator correspond to the report groups that appear in the Traffic page of the the SAS Web Analytics Report Viewer. The default report groups in the SAS Web Analytics Report Viewer are shown in the following display:

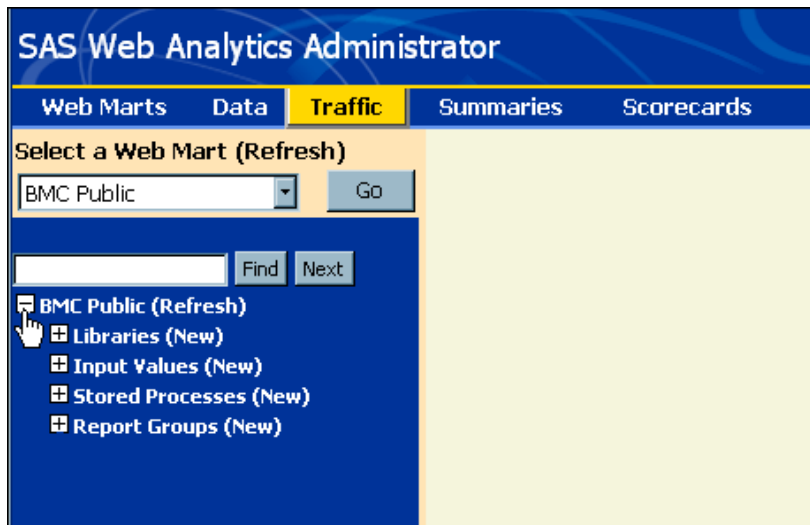
Display 10.2 Default Report Groups in the SAS Web Analytics Report Viewer

How to Create a New Report Group

Here is a stepwise example that shows how to create a new report group using the SAS Web Analytics Administrator:

Note: The SAS Web Analytics Administrator is a Web-based application. To access the SAS Web Analytics Administrator in your Internet browser, select the URL address that you configured during the setup of the SAS Web Analytics solution. △

- 1 Select **Traffic** in the main menu bar of the SAS Web Analytics Administrator.
- 2 If no Web mart is selected, or if the Web mart that is displayed is not correct, then select a Web mart from the **Select a Web Mart** drop-down menu. Click **Go**.
- 3 Click the node icon () next to a Web mart name in the tree view to expand the node. Libraries, Input Values, Stored Processes, and Report Groups nodes appear below the expanded Web mart node. For example, the following display shows the expanded node of the BMC Public Web mart:



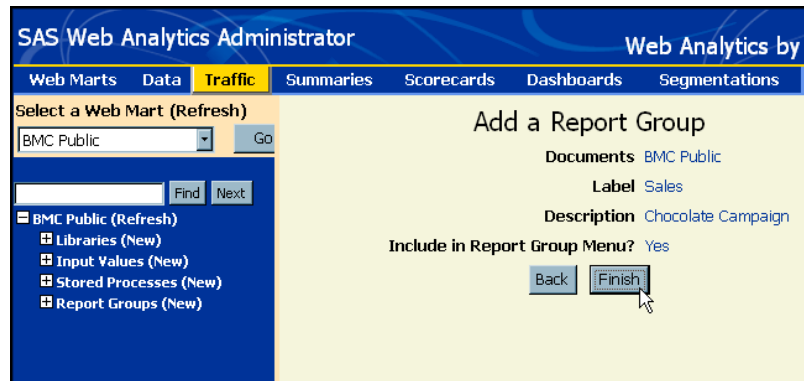
- 4 Click **(New)** next to **Report Groups** to create a new report group.
- 5 The following series of prompts for information about the new report group appear in succession. For each item, type the information in the text box and click **Next**. An asterisk (*) next to the prompt indicates that you must provide information for that item in order to finish the task.

Enter Label: The text that you enter here appears in the navigation tree of the Traffic page in the SAS Web Analytics Report Viewer.

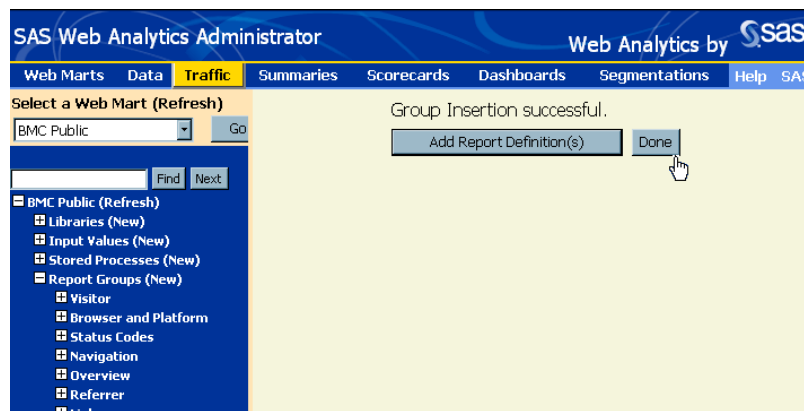
Enter Description: The text that you enter here appears in the SAS Web Analytics Report Viewer as a description of the report group.

Include in Report Group Menu: Leave the default choice as **Yes**, and click **Next**. The new report group appears in the tree view of the Traffic page in the SAS Web Analytics Report Viewer. If you select **No**, then the report group (and any reports that are created under it) are hidden.

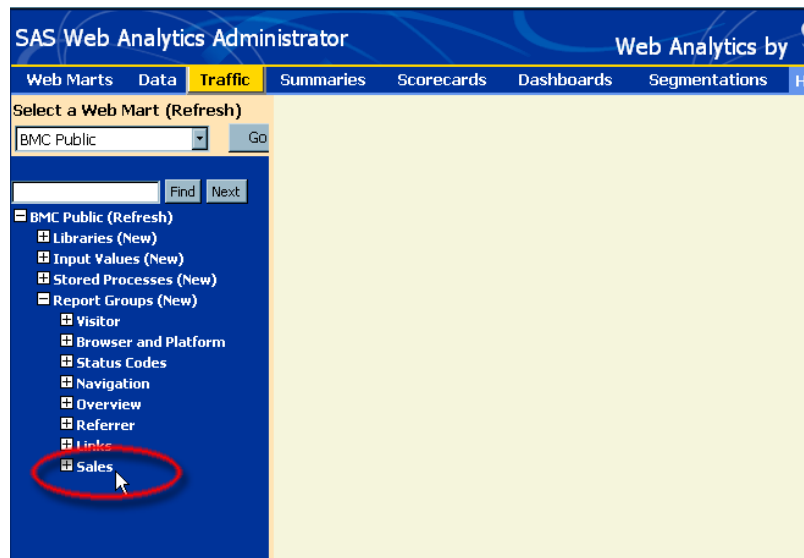
- 6 Click **Finish** (see the following display):



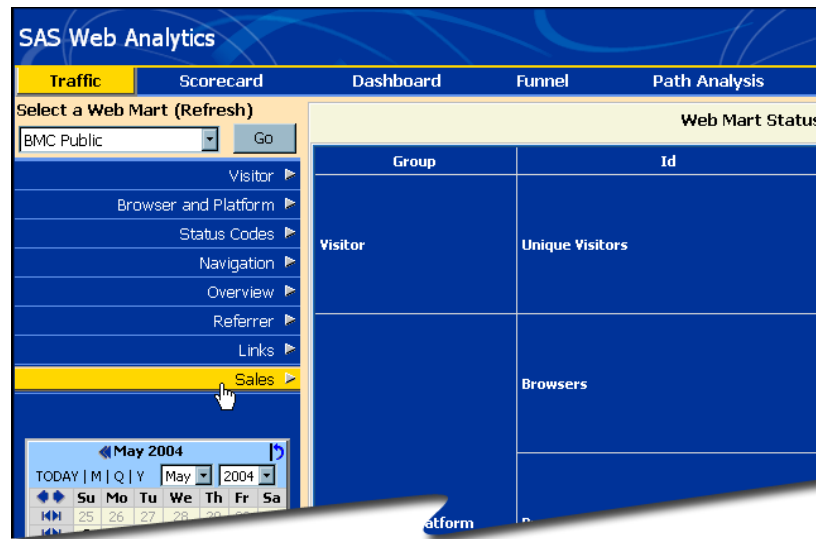
- 7 Click **Done** (see following display):



- 8 To verify the new report group (**Sales** in the following example), click either **Refresh** from the browser menu or **(Refresh)** next to the Web mart name in the tree view of the SAS Web Analytics Administrator.



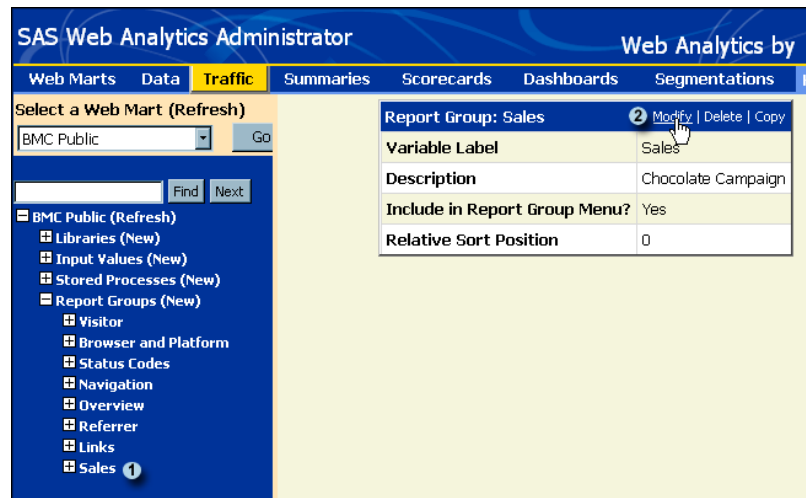
The new report group also appears on the Traffic page of the SAS Web Analytics Report Viewer. The report group Sales, which was created in the previous example, is shown in the following display:



How to Modify a Report Group

To modify the display properties of a report group, follow these steps:

- 1 Verify that you are on the Traffic page of the SAS Web Analytics Administrator (the **Traffic** link in the menu bar is yellow). In the tree view on the left, select the name of the report group below the Web mart that you want to modify. For example, the Sales report group (❶ in the following display) is selected. A table of properties appears at the right of the page.



- 2 Select **Modify** (❷ in the previous display).

- 3 In the Documents Administrator Group form, enter the changes that you want to make to the report group. Click **Submit** (see the following display):

The screenshot shows the 'Documents Administrator Group Form' in the SAS Web Analytics Administrator interface. The left sidebar contains a tree view with 'BMC Public (Refresh)' expanded, showing sub-items like 'Libraries (New)', 'Input Values (New)', 'Stored Processes (New)', and 'Report Groups (New)'. The main form area has the following fields:

- Variable Label:** Sales
- Description:** Chocolate Campaign - March
- Include in Report Group Menu?:** Yes (dropdown menu)
- Relative Sort Position:** 0

At the bottom of the form are 'Submit' and 'Cancel' buttons. A mouse cursor is pointing at the 'Submit' button.

- 4 Below the **Status Group Update successful** message that appears, click **Continue**. The updated table of properties appears (see the following display):

The screenshot shows the same interface as before, but now a table of properties for the 'Report Group: Sales' is displayed. The table has the following data:

Report Group: Sales		Modify Delete Copy
Variable Label	Sales	
Description	Chocolate Campaign - March	
Include in Report Group Menu?	Yes	
Relative Sort Position	0	

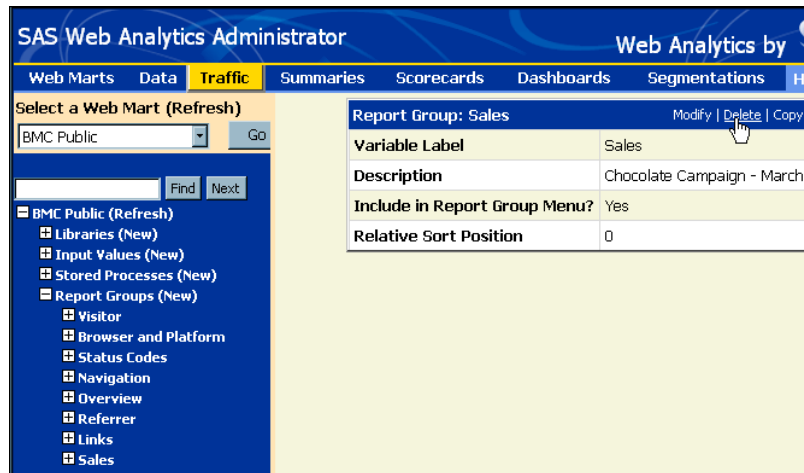
A mouse cursor is pointing at the 'Copy' link in the top right corner of the table.

How to Delete a Report Group

To delete a report group, follow these steps:

- 1 Verify that you are on the Traffic page of the Web Analytics Administrator (the **Traffic** link in the menu bar is yellow). In the tree view on the left, select the name of the report group that you want to delete. A table of properties for that report group appears at the right of the page.

- 2 Select **Delete** in the upper right corner of the table (see the following display):



- 3 Click **OK**. Below the **Status Group Deletion successful** message that appears, click **Continue**. The report group is removed from the tree view.

Creating a New Report

Before you create a new report, determine the report group to which it will belong. If none of the existing report groups is appropriate for the new report, then create a new report group first (see “How to Create a New Report Group” on page 133).

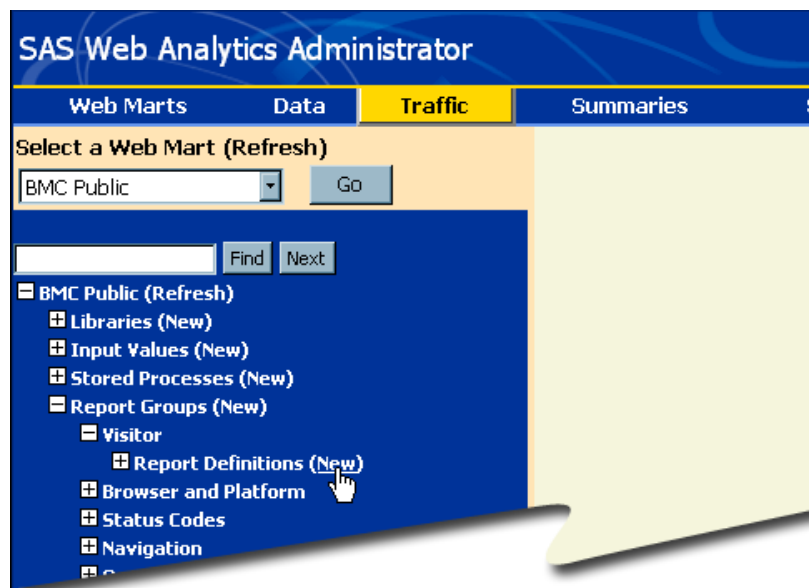
To create a new report for a selected Web mart, follow these steps:

- Select the report group to which the new report will belong.
- Create a new *report definition*.
- Refine the report (if necessary).

To create a new report definition, follow these steps:

- 1 Verify that you are on the Traffic page of the SAS Web Analytics Administrator (the **Traffic** link in the menu bar is yellow).
- 2 Select the report group by expanding the node next to the name of the report group to which you want to add a new report definition. The tree view expands to display a new item, **Report Definitions (New)**, below the selected report group.

- 3 Click **(New)** next to **Report Definitions** (see the following display):



- 4 Several prompts for information about the report definition appear. At each prompt, type the information in the text box, and click **Next**. Your typed information is added and the next prompt appears. An asterisk (*) next to a prompt indicates that you must provide information for that item in order to finish the task. The following list describes the information that you are prompted for:

Label

The text that you enter here is displayed in the SAS Web Analytics Report Viewer as the name of the report in the tree view.

URL

The SAS Web Analytics Report Viewer can display reports or can link to Web sites (for example, <http://www.sas.com>).

Link to a Web site

If you want the report to be a link to a Web site, then type the URL of the Web site in the text box.

Display a report

If you want the report to be a presentation of variables in a data set, then leave this text box empty. The form will request additional information about the report, as described in the fields that follow (**Library**, **Dataset**, **Class Variables**, and **Analysis Variables**).

Library

In the drop-down menu, select the library that contains the data set that will be displayed in the report. The default list contains the Dated and Summary libraries by default. If your data set is not in one of these libraries, then select the **Show All** option to view an expanded list of libraries.

Dataset

In the drop-down menu, select the data set that contains the variables to display in the report.

Class Variables

In the drop-down menu, select the class variables to display in the report. Use the standard Windows keyboard conventions to select contiguous or non-contiguous items from a drop-down menu.

Analysis Variables

In the drop-down menu, select the analysis variables to display in the report. Use the standard Windows keyboard conventions to select contiguous or non-contiguous items from a drop-down menu.

- 5 After you select all of the analysis variables, click **Finish**. Below the **Definition Insertion Successful** message that appears, click **Done**.

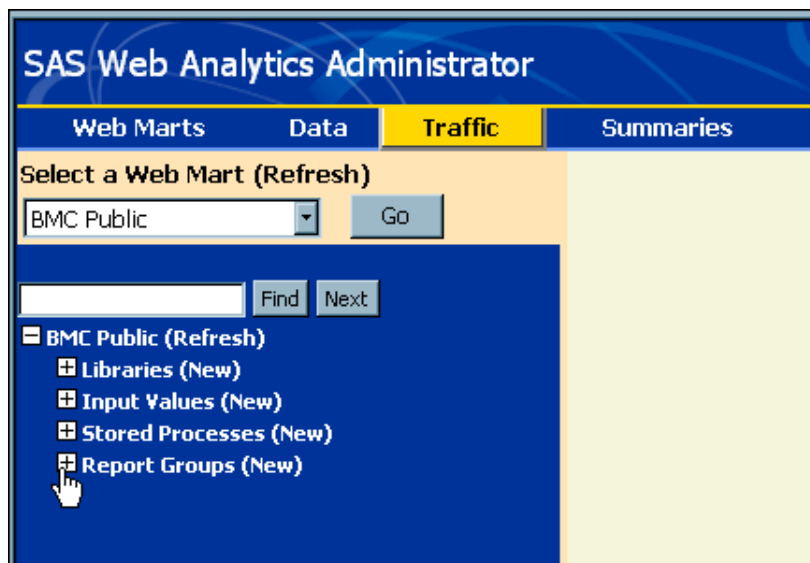
Note: The **Add Report Definitions** button enables you to continue creating another report definition instead of selecting **New** in the navigation tree of the SAS Web Analytics Administrator. △

At this point, you have created the basic definition of a report. If a basic report is sufficient for your report presentation needs, then you are finished with your report definition. However, if you want to customize a report by, for example, changing the report label or specifying the location of a custom Help file for the report, then see “Refining a Report” on page 139.

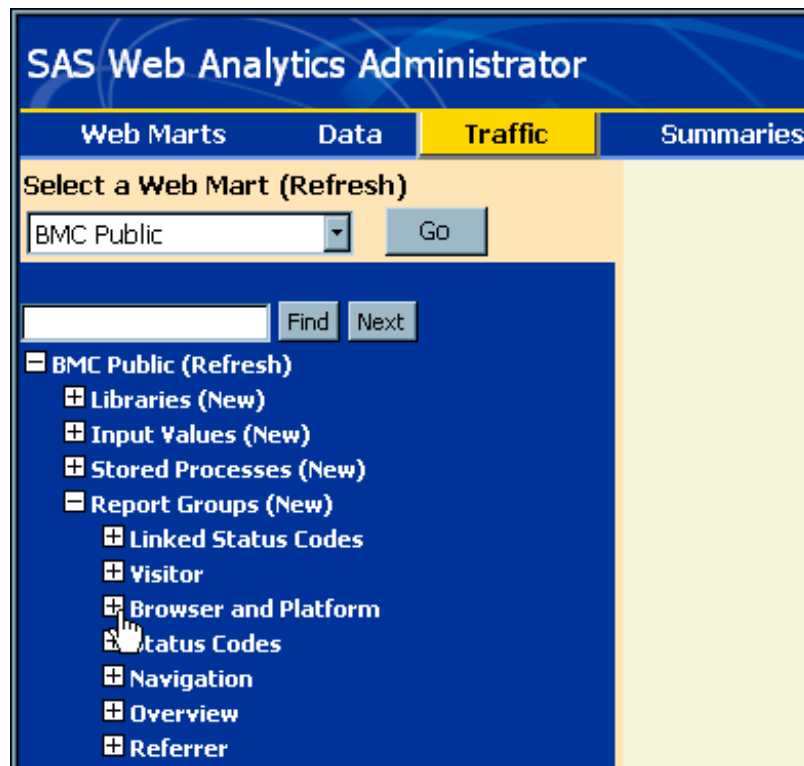
Refining a Report

To add refinements to a report, such as changing the report label or specifying the location of a custom Help file for the report, follow these steps:

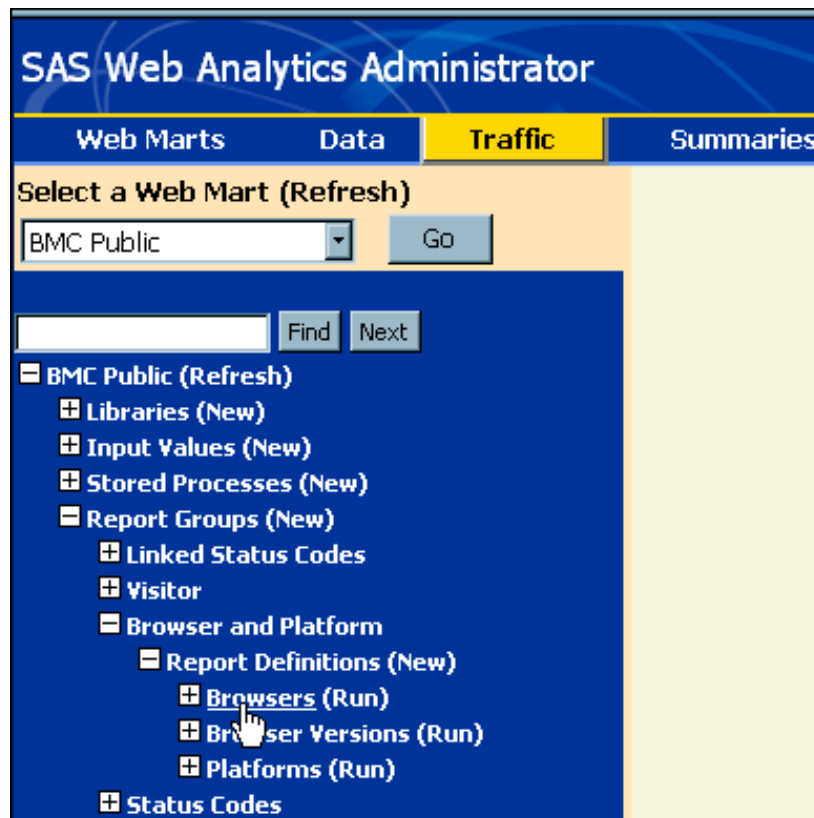
- 1 Verify that you are on the Traffic page of the SAS Web Analytics Administrator (the **Traffic** link of the menu bar in the Traffic page is yellow).
- 2 Select a Web mart (if necessary). If necessary, expand the Web mart node and the **Report Groups (New)** node.



- 3 Expand the report group that contains the report that you want to refine.



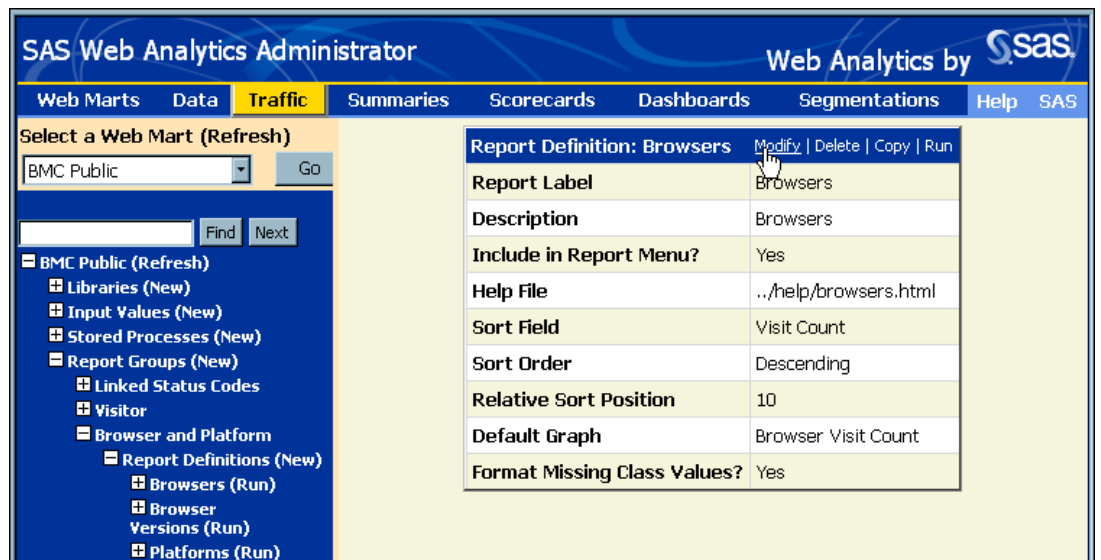
- 4 Expand **Report Definitions (New)** to view the report list. Select the report that you want to refine in the navigation tree.



The metadata table for the report appears on the right of the Traffic page.

Note: The metadata table for the report is also displayed after you finish a basic report definition. △

- 5 Select **Modify** in the upper right corner of the table.



The Documents Administrator Report Definition Form appears.

The screenshot shows the SAS Web Analytics Administrator interface. The top navigation bar includes 'Web Marts', 'Data', 'Traffic' (selected), 'Summaries', 'Scorecards', 'Dashboards', 'Segmentations', 'Help', and 'SAS'. The left sidebar shows a tree structure under 'BMC Public (Refresh)' with options like 'Libraries (New)', 'Input Values (New)', 'Stored Processes (New)', 'Report Groups (New)', 'Linked Status Codes', 'Visitor', 'Browser and Platform', 'Report Definitions (New)', 'Browsers (Run)', 'Browser Versions (Run)', 'Platforms (Run)', 'Status Codes', 'Navigation', 'Overview', and 'Referrer'. The main area is titled 'Documents Administrator Report Definition Form' and contains the following fields:

- Group:** Browser and Platform (dropdown)
- Report Label:** Browsers (text input)
- Description:** Browsers (text input)
- Include in Report Menu?:** Yes (dropdown)
- Help File:** ../help/browsers.html (text input)
- Sort Field:** Visit Count (dropdown)
- Sort Order:** Descending (dropdown)
- Relative Sort Position:** 10 (text input)
- Default Graph:** Browser Visit Count (dropdown)
- Format Missing Class Values?:** Yes (dropdown)

At the bottom right are 'Submit' and 'Cancel' buttons.

This form enables you to access the fields in order to refine the report, as well as to modify the basic report definition.

The fields in the Documents Administrator Report Definition Form are described in the following list:

Group

is the report group to which the report belongs.

Report Label

is the text that is displayed as the name of the report in the Traffic page of the SAS Web Analytics Report Viewer (and in the navigation tree of the Traffic page of the SAS Web Analytics Administrator).

Description

is a short description of the report. By default, the description is the same as the report label.

Include in Report Menu?

determines whether the report appears in the menu of the report group to which it belongs in the Traffic page in the SAS Web Analytics Report Viewer. If you select **Yes**, then the report appears in the menu. If you select **No**, then the report does not appear in the menu.

Help File

consists of the path and file name of the online help for this report. The standard SAS Web Analytics reports include online help. For customized reports, you can provide the location of a corresponding help file (if it exists) in this field.

Sort Field

lists the variables that you selected when you defined the report. If you want the report to be sorted by a specific variable, then select the variable from the drop-down list.

Relative Sort Position

is an integer that corresponds to the desired position of the report in relation to the other reports in the report group as they appear in the SAS Web Analytics Report Viewer. To move the position of a report higher in the list, change the value of its relative sort position to a number that is smaller than that of the other reports.

Note: For the relative sort position, any non-zero value takes precedence over 0. For example, if you set the relative sort position of an item in a list to 2, and if all of the other items in the list have a default relative sort position of 0, then the item with a relative sort position of 2 is placed first in the list. △

Default Graph

specifies the type of graph to display by default for this report.

Format Missing Class Values

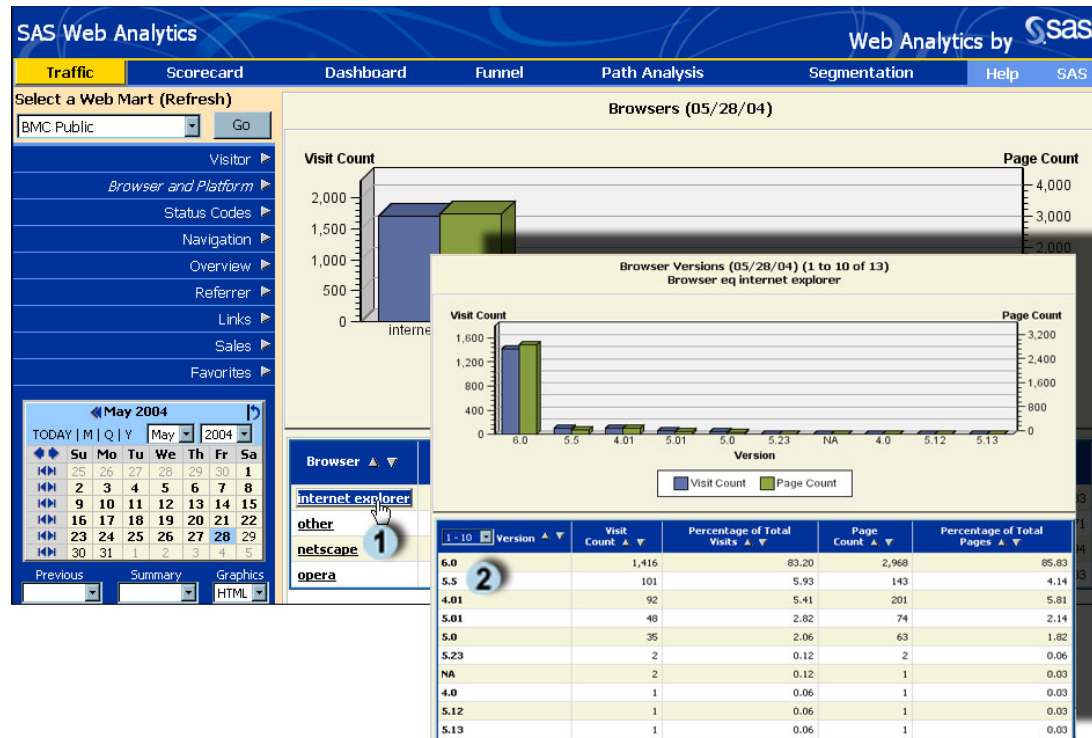
specifies how to display any missing values in the report as follows:

- ☐ If **Yes**, then the application will give placeholders to missing values.
- ☐ If **No**, then the application will display any missing values that are located within a grouping variable as blanks.

Drillable Reports

Overview: Drillable Reports

Display 10.3 Example Drillable Report in the SAS Web Analytics Report Viewer



A drillable report enables you to click on higher-level, summary information in order to view the detail data from which the summary data is derived. For example, in the Browsers main report (previous display), a user can select ❶ the Internet Explorer link to view the breakdown of the versions of Internet Explorer in ❷ the Browser Versions drilled report.

To create a drillable report, you create two reports: a main report and the drilled report. The main report can be an existing report or a new report. In the main report definition, you set the **Include in Report Menu?** field to **Yes** so users can select the main report from the list. In the drilled report definition, you can set the **Include in Report Menu?** field to **No** because users will access the drilled report through the main report.

To create a drillable report, create the drilled report first. Then, create the main report.

Examples of Drillable Reports

The default reports in SAS Web Analytics contain several examples of drillable reports. Studying these examples can help you set up your own drillable reports. The following example uses the Browsers report as the main report and the Browser

Versions report as the drilled report to illustrate how to set up the relationship between a main report and a drilled report.

Underlined values, such as the values in the first column of the Browsers report in the previous display, indicate that the Browser column is the drill mechanism: clicking values in the column opens a drilled report. The attribute that changes a column into a drill mechanism is the **Link** field. You can specify this attribute of the **Link** field ③ in the metadata table of the Browser column ④ of the Browsers report ⑤ (see the following display):

Display 10.4 The Link Field of a Column: The Drill Mechanism

The screenshot shows the SAS Web Analytics Administrator interface. On the left, a tree view under 'BMC Public' shows 'Browsers (Run)' selected, indicated by a circled 5. The main panel displays the configuration for the 'Column: Browser'. The 'Link' field is highlighted with a circled 3, showing the value '\$Browser and Platform.Browser Versions|browser'. Other fields include 'Entry' (Top), 'Variable Label' (Browser), 'Variable Name' (browser), 'Formula', and several flags like 'Is this field a classification variable?' (Yes), 'Hide this field?' (No), 'Is this field a percentage?' (No), 'Used in % calculation?' (No), 'Display ordinal?' (No), 'Sort Order' (None), 'Relative Column Position' (20), and 'Number of Rows' (0). A circled 4 points to the 'Browser (20)' entry in the left sidebar.

Specifying the Value for the Link Field for a Drillable Report

The value for the **Link** field (in the previous display) contains the information that is needed for the drill action to occur between the main report and the drilled report. The following table describes how this value for the Link field in the example (Display 10.4 on page 145) is formatted:

Table 10.1 How to Format the Value for the Link Field for a Drillable Report

\$Browser and Platform.Browser Versions browser		
Part	Description	Separator
\$Browser and Platform	The name of the report group that contains the drilled report. The report group is followed by the separator.	. (a period)
Browser Versions	The name of the drilled report. It is followed by the separator.	(a vertical slash or pipe)
browser	The name of the field that connects the main report and the drilled reports.	

To ensure that the link of a main report never points to a report that does not exist, first define the drilled report and then define the main report.

Adding a Filter to the Drilled Report

You must add a filter to the drilled report to make sure that it subsets only the rows whose values match the clicked cell in the main report. For example, in Display 10.3 on page 144, the browser versions ❷ in the drilled report should include only the versions of Internet Explorer ❶, where Internet Explorer is the value of the clicked cell in the main report.

The following display shows the metadata table ❷ of the Browser filter ❸ for the example drilled report:

The screenshot shows the SAS Web Analytics Administrator interface. The left sidebar contains a tree view of report components. The 'Traffic' tab is selected. Under 'Filters (New)', the 'Browser' filter is highlighted with a circled ❸. The main area displays the metadata table for the 'Browser (browser eq \$browser)' filter, which is circled with a ❷. The table has the following structure:

Filter: Browser (browser eq \$browser) Modify Delete Copy	
Entry	Top
Variable Name	browser
Operator	eq
Value	\$browser

When you create a drillable report, follow these general guidelines:

- First, create or modify the drilled report with the necessary filters.
- Next, create or modify the main report with the necessary links.

Variables You Can Set for Reports

You can use variables in different parts of the SAS Web Analytics Report Administrator. A variable is indicated by a leading \$. If a variable has the same name as an **Input Value** object, then the user will be able to choose from the list of the input values when he needs to resolve a variable.

Using Variables as Filter Values

You can use variables in filter values. The default report definitions enable you to set variables for most reports. The report definitions contain the filter

```
date between $start,$end
```

Because the report viewer has calendar widgets that are used to resolve these variables, the user is not directly prompted for these variables.

To enable the user to filter out records that do not meet a user-entered session count criterion, you can define the filter

```
session_count gte $session_count
```

When the user initially runs the report, the user is prompted to enter the session count. For subsequent runs, the user no longer has to provide the session count because the value for session count is prepopulated.

Using Variables as Hyperlinks

The **Link** field can be used to set a hyperlink from a row to another table. For example, you may want to let the user link from a main report (for example, Browser) to a linked report (for example, Browser Version) that contains the detailed breakout for one of the summarized metrics (for example, Internet Explorer).

To include some context for the hyperlink, use the following syntax:

```
{url}|variable1,variable2,...,variablen
```

To specify a URL to be a link to another SAS Web Analytics report, use the following syntax:

```
$(Report Group).(Report Name)
```

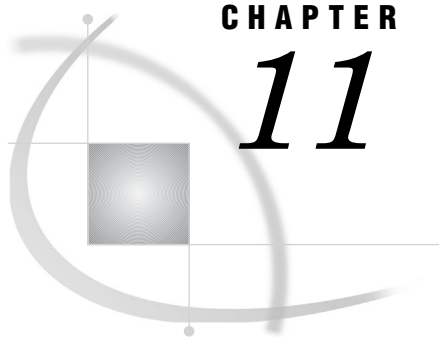
For example, a browser report, Browsers, links to another report, Browser Versions. Browser Versions uses the following filter:

```
browser eq $browser
```

To set the hyperlink from the browser name to the Browser Versions report and to pass in the value of the variable Browser, set the following link in the Browser field of the Browsers report as follows:

```
$Browser and Platform.Browser Versions|browser
```

For details, see “Specifying the Value for the Link Field for a Drillable Report” on page 145.



CHAPTER

11

Scorecard Administration

<i>Overview: Scorecard</i>	149
<i>Data Requirements for a Scorecard</i>	150
<i>Defining a New Scorecard: Preparation</i>	150
<i>Guidelines for Choosing the Input Metrics for a Scorecard</i>	150
<i>The Data Set for Storing the Target and Input Metrics for a Scorecard</i>	151
<i>Specifying the Input Information</i>	151
<i>The Types of a Metric in a Scorecard</i>	151
<i>The Business Direction of a Metric in a Scorecard</i>	151
<i>The Measurement Range of a Metric in a Scorecard</i>	151
<i>Running a Custom Scorecard</i>	152
<i>How to Create a New Scorecard</i>	152
<i>Working with Metrics</i>	154
<i>Changing an Input Metric to a Target Metric</i>	155
<i>Adding Variables That Are Not in the Web Log to a Scorecard</i>	156
<i>Testing the Scorecard Output: The %WADECIDE Macro</i>	157
<i>Explanation of the Example Code for the Scorecard</i>	157
<i>The Table Layout of the Scorecard Report</i>	158
<i>The Variables of the Site Metrics Scorecard Table</i>	159
<i>Name</i>	159
<i>Actual Values</i>	159
<i>Predicted Values</i>	159
<i>Performance</i>	159
<i>Relative Difference</i>	160
<i>Goal-Seeking Table (Site Metrics)</i>	160
<i>How Special Cases are Handled</i>	161
<i>Example of a Missing Day in the Input Data Set for a Scorecard</i>	161

Overview: Scorecard

The scorecard is a decision support report that enables the user to view the performance and the forecast values of the key performance indicators (KPIs) that are related to traffic on the Web site. The default KPI is the total session count for the current day that is being viewed. The scorecard performs the following tasks:

- determines the variables in the input data set that have a statistically significant impact on the target variable
- lists these variables in the order of how each variable affects the target variable.

The scorecard requires a single data set that contains a set of metric variables that are summarized by day. The required columns for this data set are Date, a target

variable, and one or more input variables. The default scorecard uses the Summary.Daily_Total_Day data set and uses Session Count as the target. Target metrics are the numeric metrics that measure the success of a site. Input metrics are the numeric metrics that are the potential drivers of the target metric.

The scorecard is generated by code within the %WAETL macro. After the %WAETL macro summarizes all of the detail data, the default data set called Daily_Total_Day (located in the Summary library) is used as input to the scorecard report. By default, the Daily_Total_Day data set contains values for all of the traffic-type metrics and the metrics about the health of the Web site.

Note: Examples of traffic-related metrics are Session Count, Page Count, Duration, and Bytes Sent. Examples of metrics about the health of a Web site (that are contained in the Summary.Daily_Total_Day data set) are the status code variables such as ERROR_302_CNT and ERROR_402_CNT. △

Data Requirements for a Scorecard

In order to produce the scorecard statistics for a given date, at least thirty continuous days of data must exist in the input data set (Summary.Daily_Total_Day, by default) in the time interval before that date. Therefore, no scorecard will be created for the first thirty days that SAS Web Analytics software initially processes Web log data. For example, if Web log processing starts on January 1, then the SAS Web Analytics Report Viewer shows the following message in the scorecard for any day earlier than January 31:

NOT ENOUGH PREVIOUS DAYS OF DATA

All the other numeric columns are set to zero for these days as well.

Defining a New Scorecard: Preparation

To define a new scorecard, you select the target metrics and the input metrics. Examples of target metrics include the following items:

- ☐ hits to particular content on the website
- ☐ number of sales or dollars from sales
- ☐ number of users who complete an online task
- ☐ users who unsubscribe
- ☐ single-hit sessions

Guidelines for Choosing the Input Metrics for a Scorecard

Use the following guidelines in choosing the input metrics for a scorecard:

- ☐ The input metrics should make business sense for affecting the target metric.
- ☐ The input metrics should be measures that can be controlled. For example, the number of files that are not found might be controlled using site maintenance to find and correct broken links. Referrals from a particular Web site might be increased with banner ads or sponsored links.
- ☐ Each input metric should measure a unique activity and represent distinct information about the target. For example, if the target is *total book sales*, then do

not include both the *percent book orders of total orders* and the *total book orders* as input metrics.

The Data Set for Storing the Target and Input Metrics for a Scorecard

If the input metrics and the target metrics are not stored in the Summary.Daily_Total_Day data set, then you can create either a new data set of daily summaries, or you can add the target metrics and the input metrics to the Summary.Daily_Total_Day data set (see Example 2 under Modifying the %SWA_ETL Macro). You can obtain data from Web logs or from external data sources such as customer relationship management databases.

Specifying the Input Information

After you verify that the data set for the input metrics exists, you define a set of input metrics that are used to create the scorecard report. To create a new scorecard, the information that you need to provide includes the following specifications:

- the types of scorecard variables (target and input)
- business direction
- measurement range

The Types of a Metric in a Scorecard

A metric that you specify as a scorecard variable must belong to one of the following types:

Target	the business metric (variable) that is the performance indicator of the Web site
Input	metrics that have the potential to affect the target

The Business Direction of a Metric in a Scorecard

The business direction is a specification that tells the scorecard which direction is desirable for a given metric. For example, you might want the values for Session Count to increase over time. Therefore, you might set the Positive Business Direction for Session Count to **up**. For another metric such as error count (whose values you want to decrease over time), you might set the Positive Business Direction for Error 404 Count to **down**. The Positive Business Direction value for each metric is stored in the report definition for the scorecard.

The Measurement Range of a Metric in a Scorecard

The measurement range enables you to specify a valid range of values for an input variable (within a goal-seeking report such as a scorecard). All values that lie outside this range are displayed as **missing**. You can specify the following measurement ranges:

- number
- positive number
- negative number
- percentage
- negative percentage
- positive percentage

- proportion
- negative proportion
- positive proportion

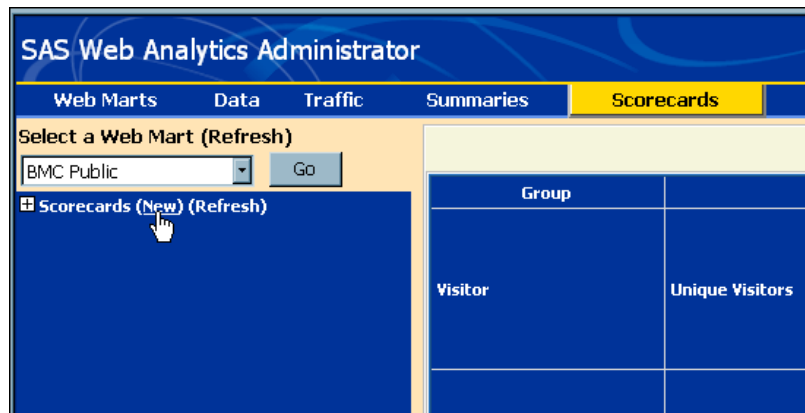
Running a Custom Scorecard

After a custom scorecard has been defined, it will run automatically within the %WAETL macro. The one exception to this automatic process is when the data set that is used by the scorecard was created outside of the %WAETL macro.

How to Create a New Scorecard

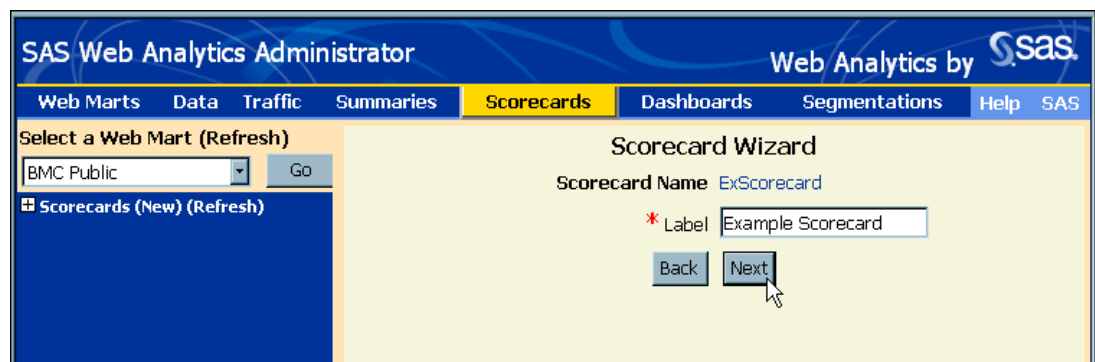
The following steps show you how to create a new scorecard by using an example:

- 1 Open the SAS Web Analytics Administrator. Select the **Scorecards** link from the banner.
- 2 Select a Web mart (if it is not already selected) and click **Go**.
- 3 Click **New** next to **Scorecards** in the navigation tree at the left.



The Scorecard Wizard appears.

- 4 Enter a name for the new scorecard in the **Name** field. The name can be a maximum of 25 characters long and can include spaces. The name is the value that is used to identify the scorecard to the SAS Web Analytics application. Click **Next**.
- 5 Enter a label for the new scorecard. The label is the text that appears below **Scorecards** in the navigation tree. Click **Next**.



- 6 From the drop-down menu, select a library that contains the data set that will be used for the scorecard. The default menu contains the Dated and Summary libraries by default. If your data set is not in one of these libraries, then click the **Show All** box to view an expanded list of libraries. When you have selected a library, click **Next**.
- 7 Select a data set from the drop-down menu. Click **Next**.

SAS Web Analytics Administrator

Web Analytics by SAS

Web Marts Data Traffic Summaries **Scorecards** Dashboards Segmentations Help SAS

Select a Web Mart (Refresh)

BMC Public Go

Scorecards (New) (Refresh)

Scorecard Wizard

Scorecard Name ExScorecard

Library summary

Label Example Scorecard

* Dataset

Back

decsupp_status_s0278269

decsupp_status_tmp

hourly_metrics

hourly_status

list_browser

list_platform

page_dates

page

page_frequencies

page_status

pathing

Click **Finish**.

- 8 Select a target metric from the drop-down menu. Click **Next**.
- 9 Select one or more input metrics from the drop-down menu. Click **Finish**.

SAS Web Analytics Administrator

Web Analytics by SAS

Web Marts Data Traffic Summaries **Scorecards** Dashboards Segmentations Help SAS

Select a Web Mart (Refresh)

BMC Public Go

Scorecards (New) (Refresh)

Site Metrics (Run)

Metrics (New)

Scorecard Wizard

Scorecard Name ExScorecard

Library summary

Dataset hourly_metrics_day

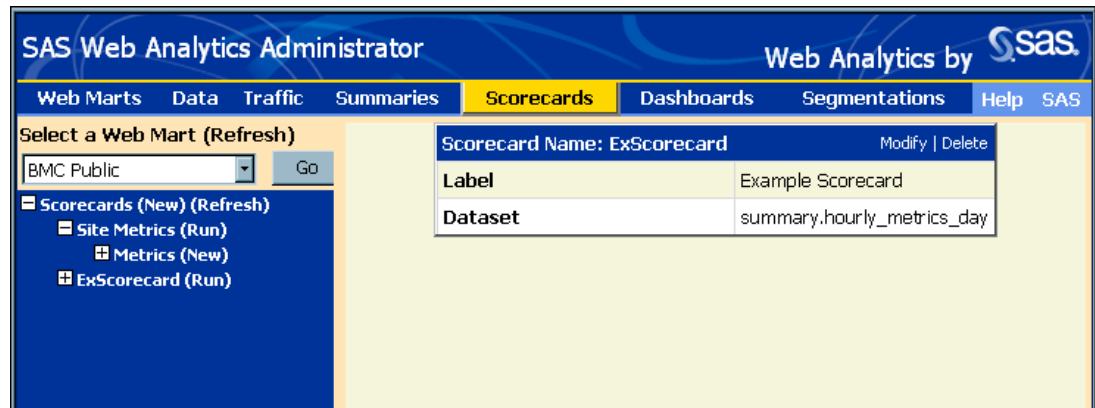
Label Example Scorecard

Target date

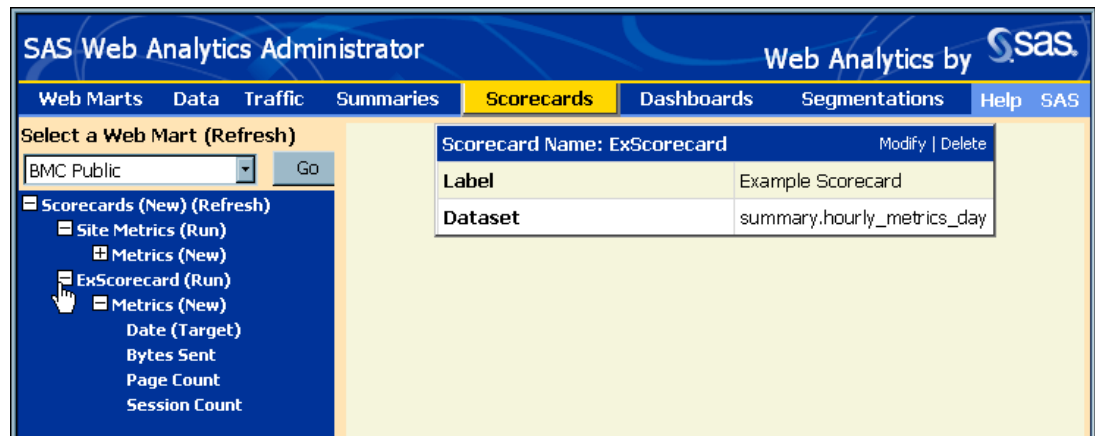
Metrics bytes_sent,page_count,session_count

Back Finish

- 10 Click **Continue** in order to create an entry within the Config.Wascrcrd data set for the new scorecard and to update the navigation tree. A metadata table appears on the right:



- 11 Click + next to the new scorecard name. The expanded node displays the input metrics and the target metric that you set up during the creation of the new scorecard in the navigation tree.



Working with Metrics

You can add a new input metric, modify an existing input metric, or delete an input metric for a particular scorecard. To modify or delete an input metric, follow these general steps:

- Select the **Scorecards** link from the banner of the SAS Web Analytics Administrator.
- In the navigation tree, expand the node of the scorecard to view its metrics.
- Select a metric. Its metadata table appears on the right.
- In the upper right corner of the metadata table, select the action that you want to perform to the metric.

To add an input metric or to specify the target metric for a scorecard, select **New** next to **Metrics** below the scorecard of interest. In the Scorecard New Metrics Wizard that appears, provide the value(s) for each requested field and click **Next**.

Changing an Input Metric to a Target Metric

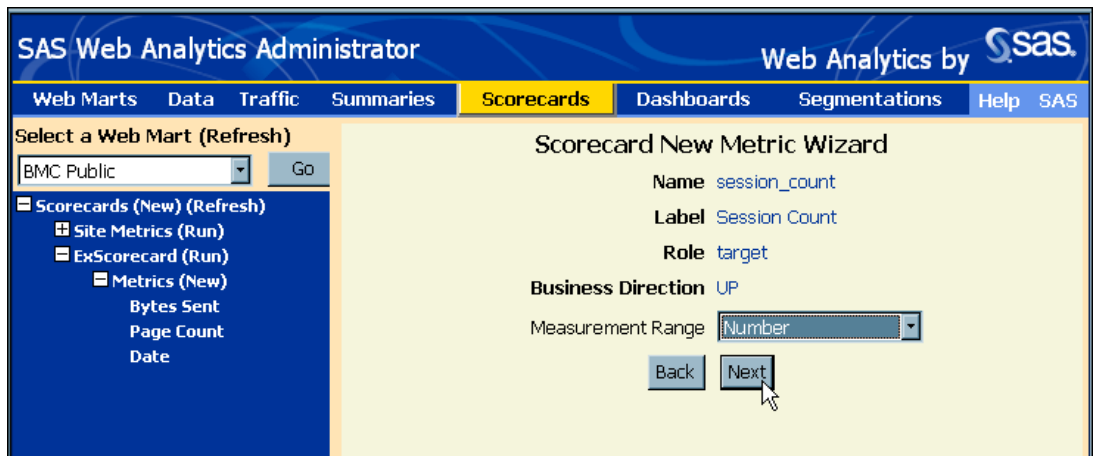
If you set up a scorecard metric as an input metric by mistake, and you want to change it to a target metric, then you must delete the input metric first and then re-create it as a target metric by following these steps:

- 1 Select the Scorecards link from the banner in the SAS Web Analytics Administrator (the **Scorecards** link is yellow when selected). In the navigation tree, click + to expand the scorecard name that contains the metric that you want to modify. Click the metric in the navigation tree to display the metadata table for the metric on the right. Select **Delete** in the upper right corner of the table to delete the metric.



Click **OK** in the dialog box. Click **Continue**. The metric is removed from the navigation tree.

- 2 In the navigation tree, select **New** next to **Metrics** below the scorecard (to create the metric again, but this time as a target metric). In the Scorecard New Metric Wizard that appears, provide the information for each of the prompts. In particular, select **target** for the role.



After you supply the value for Measurement Range, click **Finish**. Click **Continue**. The metadata table for the newly created metric appears on the right of the Scorecards page. To view the new metric in the navigation tree, expand the **Scorecards** node, and if necessary, expand the scorecard node of interest.

Input: Session Count	
Name	session_count
Label	Session Count
Business Direction	UP
Measurement Range	Number
Link	

Adding Variables That Are Not in the Web Log to a Scorecard

Occasionally you might want to add variables to the summary data sets after the %WAETL macro has processed the Web log data. To add non-Web log variables to the summary data sets, perform the following steps:

- 1 Before you create the scorecard and dashboard definitions, create a new summary data set (in the Summary library) that will combine the data (for example, daily sales totals) from a non-Web log data source with data from a Web log.

Here is the code that illustrates the preceding scenario:

```
data summary.scorecard_day;
  merge summary.daily_total_day
        summary.daily_sales end=eof;
  by date;
run;
```

- 2 Create the scorecard definitions by using the SAS Web Analytics Administrator.
- 3 In the batch program Daily.sas, add the following code for each scorecard after the %WAETL invocation:

```
/* Scorecard example */
data summary.scorecard_day;
  merge summary.daily_total_day
        summary.daily_sales end=eof;
  by date;
/**
 * Put custom code here to handle missing data
 */
/* Get the first and last date within the scorecard data set */
if _n_ then call symput('first_date', ""||put(date,date9.)||" 'd");
if eof then call symput('last_date', ""||put(date,date9.)||" 'd");
run;

%wadeide(program=scorecard,
```

```

date_1=&first_date,
date_2=&last_date,
swamart=&webmart\swamart,
indsn=summary.scorecard_day,
outlib=devlib,
use_first_two_weeks=no,
scorecard_to_run=scorecard_name
);

```

See also “General Information About SAS Web Analytics Summaries” on page 195.

Testing the Scorecard Output: The %WADECIDE Macro

To view the new scorecard, run the %WADECIDE macro within a SAS interactive session by using the example that follows as a guide. For more information about the %WADECIDE macro, see Appendix 1, “Macro Reference for the SAS Web Analytics 5.1 Solution,” on page 179.

The following block of code can be used to test a scorecard:

```

libname devlib 'c:\xxx';
%wadeide(program=scorecard,
        date_1='1jun2004'd,
        date_2='30jun2004'd,
        swamart=c:\xxx\swamart,
        temp_store= c:\xxx\swamart\temp,
        indsn=devlib.testdata,
        outlib=devlib,
        use_first_two_weeks=no,
        scorecard_to_run=Example Scorecard
);
proc print data=devlib.scorecard_data
        (where=('1jun2004'd le date le '30jun2004'd));
run;
proc print data=devlib.scorecard_metric_history
        (where=('1jun2004'd le date le '30jun2004'd));
run;

```

Explanation of the Example Code for the Scorecard

Here is the explanation of the preceding block of code:

- Use data that ranges from June 1, 2004 to June 30, 2004.
- Specify that the root directory of the SAS Web Analytics Web mart is at **c:\xxx\swamart**.
- Store the intermediate results in the **c:\xxx\swamart\temp** directory (the location of the Worklib library).
- Use the Devlib.Testdata data set as the data source.

Note: The Devlib library has been defined separately from the %WADECIDE macro. △

- Store the scorecard report in the Devlib library.

Note: When you provide an argument for the Outlib parameter, the output will not be stored in the default Summary library and you will not be able to see the

results using the SAS Web Analytics Report Viewer. To remedy this, two PROC PRINT steps have been added after the invocation of the %WADECIDE macro. △

- Exclude the first 14 days (first two weeks) of data
- Create the scorecard whose name is *Example Scorecard*.
- Display the scorecard report in the Output Window using PROC PRINT.
- Display the historical values for each metric plus seven days of forecasted values in the Output Window using PROC PRINT.

The Table Layout of the Scorecard Report

To view a scorecard, follow these steps:

- 1 Select the **Scorecard** link in the SAS Web Analytics Report Viewer banner.
- 2 If a Web mart is not selected, then select a Web mart from the **Select a Web Mart** drop-down menu (located at the upper left of the SAS Web Analytics Report Viewer interface).


Note: The title SAS Web Analytics (without the words Report Viewer) appears in the title bar of the SAS Web Analytics Report Viewer. In contrast, the SAS Web Analytics Administrator interface always includes the word Administrator in the title bar. △

- 3 Select a date from the calendar on the left.

The scorecard report has two tables. The upper table (Scorecard Table) contains the scorecard data for the selected day. The lower table (Goal-Seeking Table) contains the goal-seeking data.

Display 11.1 Example of a Scorecard in the SAS Web Analytics Report Viewer

SAS Web Analytics

Web Analytics by 

Traffic

Scorecard

Dashboard

Funnel

Segmentation

Refresh

Help

SAS

Select a Web Mart

SFI Demo2

Go

Scorecards

View the scorecard

For Date 2004-6-8

<< Jun 2004 >>

TODAY Jun 2004

Sund Mond Tues Wedn Thur Frid Satu

6 7 8 9 10 11 12

13 14 15 16 17 18 19

20 21 22 23 24 25 26

27 28 29 30

with HTML graphics

Run

Site Metrics Scorecard (6/8/04)

Name	Actual Values	Predicted Values	Performance	Relative Difference
Session Count (Target)	4,333	2,777.45	▲+	1.12
Page Count	6,582	4,598.18	▲+	0.93
Total Bytes Sent	36,109,211	29,362,525.73	◆	0.60

Site Metrics Scorecard Goal Seeking (6/8/04)

Name	0% Increase	5% Increase	10% Increase	15% Increase	20% Increase
Session Count (Long Term Forecast)	3,995.21	4,194.97	4,394.73	4,594.49	4,794.26
Page Count	6,582.00	7,034.34	7,486.69	7,939.03	8,391.37
Total Bytes Sent	36,109,211.00	41,176,481.76	46,243,752.53	51,311,023.29	56,378,294.06

The Variables of the Site Metrics Scorecard Table

The scorecard table lists the name of the scorecard and the date for which it was processed. This date is the date that you selected in the calendar. Directly below **Name** in the scorecard table is the target variable for the scorecard. The word **Target** is appended to the name of the target variable. The remaining columns in this table are Actual Values, Predicted Values, Performance, and Relative Difference.

Name

The variable Name is the name of the key performance indicator that is used to measure Web site activity. The first key performance indicator that is listed in the column is the target metric. The target key performance indicator is the key performance indicator that you have determined to be most important to gauging the activity on the site. The remaining key performance indicators in the column are sorted in descending order based on their relative significance in predicting performance of the target key performance indicator. Each metric links to a forecast plot. Click the metric to view the forecast plot.

Actual Values

The value in the Actual Values column is the numeric percentage, proportion, count, or amount for the associated key performance indicator. The Actual Values column contains the value for each metric (such as Session Count) on the day the report was run.

Predicted Values

The value in the Predicted Values column is the predicted value of the associated key performance indicator. The predicted value is derived from the history data, taking into account any historic trends. The Predicted Values column contains the value that was predicted for the day the report was run, given the values for that metric in the past. Predicted values are determined by performing a variety of forecasting methods on each metric. Some of the exponential smoothing methods include the following routines:

- ☐ single
- ☐ double
- ☐ linear
- ☐ damped trend
- ☐ seasonal
- ☐ Winters multiplicative
- ☐ Winters additive






The forecast models are created by using all of the historical data that has been collected (actual values), but the current daily actual values are not used. The forecast model for the current day is computed from the selected model, and the corresponding prediction intervals and standard errors for the forecast estimates are reported in the graph.

Performance

The symbol in the Performance column is determined by where the actual value of the metric falls within the 95% predicted confidence limits. In addition, the business

direction of the metric is used to determine the symbol. The following table lists the values for the Performance symbols:

Table 11.1 The Performance Symbols in a Scorecard

Symbol	Description
	Indicates that the metric is performing at a steady business state. The actual value for this metric falls within the 95% confidence level for the predicted value.
	Indicates that the metric value is increasing, and this trend matches the desired business direction. The actual value is above the 95% confidence level for the predicted value.
	Indicates that the metric value is increasing, and this trend does not match the desired business direction. The actual value is above the 95% confidence level for the predicted value.
	Indicates that the metric value is decreasing, and this trend matches the desired business direction. The actual value is below the 95% confidence level for the predicted value.
	Indicates that the metric value is decreasing, and this trend does not match the desired business direction. The actual value is below the 95% confidence level for the predicted value.

Note: The plus (+) and minus (-) signs next to the arrows are another way of indicating the direction of the metric. The signs have no additional meaning. Either a minus sign or a red color indicates a trend in the Negative Business Direction. Either a plus sign or a green color indicates a trend in the Positive Business Direction. △

Relative Difference

The Relative Difference column contains the value that is calculated using the following formula:

$$((\text{Actual Value} - \text{Predicted Value}) / \text{Standard Deviation}) / 2$$

This formula gives a scaled magnitude measure of the difference between the actual value and the predicted value for each metric in the scorecard. You use this value to get an idea of how common (or how abnormal) the actual value for the metric is today, as opposed to a typical value for the metric in general.

Goal-Seeking Table (Site Metrics)

Use the Goal-Seeking Table in the scorecard in order to determine how the increases or decreases in certain KPIs will affect the value of the long term forecast of the target KPI. The long term forecast of the target KPI is always listed first in the **Name** column. The Goal-Seeking Table shows the increases in the long term forecast of the target KPI from 0% to 50% in increments of 5%.

How Special Cases are Handled

If the scorecard processing determines that there are no metrics in the input data set that have a statistically significant impact on the target variable, then the scorecard table will display the following text:

```
NO SIGNIFICANT INPUTS FOUND FOR TARGET
```

All the other numeric columns will be set to zero for these days as well.

If a scorecard is processed on a day when a previous day is missing in the data, then the scorecard table will display the following text:


```
MISSING DAY --- NO FORECAST
```

Example of a Missing Day in the Input Data Set for a Scorecard

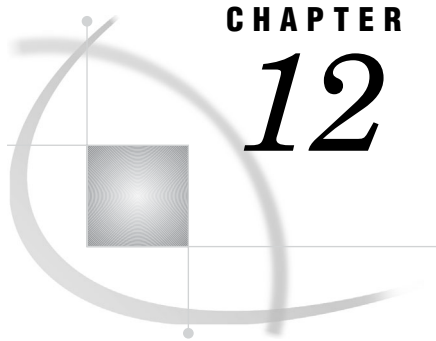
For example, the default scorecard is processed for March 4. For some reason, the data for March 1 is not in the default input data set (Daily_Total_Day). The text **MISSING DAY --- NO FORECAST** is placed in the scorecard data for the March 4 scorecard. This item will continue to cause the scorecard to fail until the data is reloaded or until the March 1 day ages out of the Daily_Total_Day data set. The default number of days that are kept in this data set is 90 days. The value for the number of days kept in history is located in the configuration data set, which can be viewed and modified using the SAS Web Analytics Administrator.

To access and to modify the number of days that are kept in this data set, complete the following steps:

- 1 Open the SAS Web Analytics Administrator. Select the Web mart for which you wish to view configuration data.
- 2 Select **Data**.
- 3 Select **Configuration Table** and then select **Web Analytics Configuration**. The Waconfig table appears.
- 4 To change the number of days that are kept in the daily summary table

(Summary.Daily_Total_Day), click  in the row for **wab_days_in_history**. **wab_days_in_history** has the description **The number of days available in daily summaries**.

- 5 You should coordinate any changes to this configuration table with the site administrator. Changes that you make to values for specific variables in the configuration table for the scorecard will also affect any other SAS Web Analytics reports that are controlled by the same variables.



CHAPTER

12

Dashboard Administration

<i>Overview: Dashboard</i>	163
<i>Data Requirements for a Dashboard</i>	163
<i>Specifying the Information for a Metric in a Dashboard</i>	164
<i>How to Create a New Dashboard</i>	165
165	
<i>Modifying the Metrics for a Dashboard</i>	166
<i>Adding a New Metric to a Dashboard</i>	168
<i>Testing the Dashboard Output: The %WADECIDE Macro</i>	170
<i>Explanation of the Example Code for the Dashboard</i>	170

Overview: Dashboard

A dashboard describes how a particular metric performs as specified by its recent trend and how that trend relates to its desired business direction. The dashboard displays the values for each site's metric, assigns a performance level to that metric as specified by its desired business direction, and displays a historical average, minimum, and maximum for that metric for a given date. The dashboard also provides a plot of the historical values (for the past 7 days) for a metric by using an overlaid trend line.

The dashboard requires a single data set that contains a set of metric variables that are summarized by day. The default dashboard uses the Summary.Daily_Total_Day data set, which contains only the daily metric data from the Web log. Additional metric variables can be added to the Summary.Daily_Total_Day data set or to a new daily data set that is created for the dashboard.

The dashboard is generated by code within the %WAETL macro. After the %WAETL macro summarizes all of the detail data, the default data set called Daily_Total_Day (located in the Summary library) is used as input to the dashboard. By default, the Summary.Daily_Total_Day data set contains values for all of the traffic-type metrics and health-type metrics for the Web site.

Note: Examples of traffic-type metrics are Session Count, Page Count, Duration, and Bytes Sent. Examples of metrics about the health of a Web site (that are contained in the Summary.Daily_total_day data set) are the error code variables such as ERROR_302_CNT and ERROR_402_CNT. △

Data Requirements for a Dashboard

In order to produce the dashboard statistics for a given date, at least thirty continuous days of data must exist in the input data set (Summary.Daily_Total_Day, by

default) before that date. Therefore, no dashboard will be created for the first thirty days that SAS Web Analytics software initially processes Web log data. For example, if Web log processing starts on January 1, then the SAS Web Analytics Report Viewer shows the following message in the dashboard table for any day earlier than January 31:

NOT ENOUGH PREVIOUS DAYS OF DATA

Specifying the Information for a Metric in a Dashboard

For each metric, you need to define the following fields:

Label	is the text that is displayed as the name of the report in the Traffic page of the SAS Web Analytics Report Viewer (and in the navigation tree of the Traffic page of the SAS Web Analytics Administrator).
Category	is a specification under which you can group together (within a dashboard) several metrics that have a common feature.
Positive Business Direction	is a specification that tells the dashboard which direction is desirable for a given metric. For example, you might want the values for Session Count to increase over time. Therefore, you might set the Positive Business Direction for Session Count to up . For another metric such as error count (whose values you want to decrease over time), you might set the Positive Business Direction for Error 404 Count to down . The Positive Business Direction value for each metric is stored in the report definition for the dashboard.
Link	can be used to set a hyperlink from a row to another table. To provide the destination information for the hyperlink, use the following syntax:

```
{url}|variable1,variable2,...,variablen
```

To specify a URL to be a link to another SAS Web Analytics report, use the following syntax:

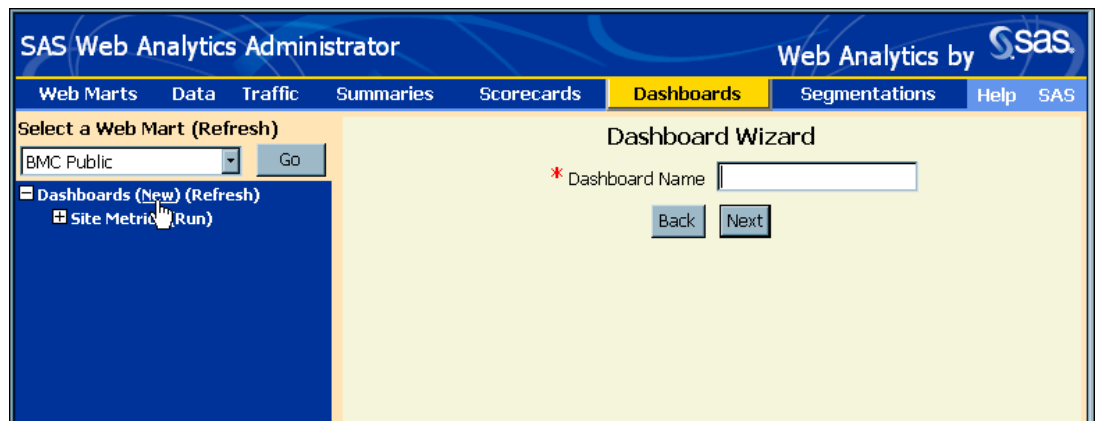
```
$(Report Group).(Report Name)
```

See also “Specifying the Value for the Link Field for a Drillable Report” on page 145.

How to Create a New Dashboard

The following example shows you how to create a new dashboard. In the following steps, the daily error count data from the Summary.Daily_Total_Day data is selected for the new dashboard in the example:

- 1 Open the SAS Web Analytics Administrator. Select the **Dashboards** link from the banner.
- 2 Select a Web mart (if it is not already selected) and click **Go**.
- 3 Click **New** next to **Dashboards** in the navigation tree at the left. The Dashboard Wizard appears.



- 4 Enter a name for the new dashboard in the **Dashboard Name** field. The name is the value that is used to identify the dashboard to the SAS Web Analytics application. The name can be a maximum of 25 characters long and can include spaces. Click **Next**.
- 5 Enter the label for the new dashboard in the **Dashboard Label** field. The label is the text that appears as the name of the dashboard that users will select in the Dashboards page of the SAS Web Analytics Report Viewer (and in the navigation tree of the Dashboards page of the SAS Web Analytics Administrator). Click **Next**.
- 6 From the drop-down menu, select a library that contains the data set that will be used for the dashboard. The default menu contains the Dated and Summary libraries by default. If your data set is not in one of these libraries, then click the **Show All** box to view an expanded list of libraries. When you have selected a library, click **Next**.
- 7 Select a data set and click **Next**.
- 8 Select one or more input metrics from the drop-down menu. Use the standard Windows keyboard conventions to select contiguous or non-contiguous items from a drop-down menu. Click **Finish**.

SAS Web Analytics Administrator

Web Analytics by SAS

Web Marts Data Traffic Summaries Scorecards **Dashboards** Segmentations Help SAS

Select a Web Mart (Refresh)

BMC Public Go

Dashboards (New) (Refresh)

Site Metrics (Run)

Dashboard Wizard

Dashboard Name: ExampleDash

Dashboard Label: Example Dashboard

Library: summary

Dataset: daily_total_day

Metrics: err_302_count, err_404_count, err_500_count

Back Finish

- 9 Click **Continue** to create an entry within the Config.Wadshbrd data set for the new dashboard and to update the navigation tree.

SAS Web Analytics Administrator

Web Analytics by SAS

Web Marts Data Traffic Summaries Scorecards **Dashboards** Segmentations Help SAS

Select a Web Mart (Refresh)

BMC Public Go

Dashboards (New) (Refresh)

Site Metrics (Run)

ExampleDash (Run)

Dashboard Name: ExampleDash Modify Delete	
Label	Example Dashboard
Dataset	summary.daily_total_day

Modifying the Metrics for a Dashboard

After you have created a new dashboard, you can modify the attributes for each of the metrics that you specified during the creation of the dashboard by following these steps:

- 1 Click + next to the dashboard of interest to expand the node in the navigation tree of the Dashboards page.

SAS Web Analytics Administrator

Web Analytics by SAS

Web Marts Data Traffic Summaries Scorecards **Dashboards** Segmentations Help SAS

Select a Web Mart (Refresh)

BMC Public Go

Dashboards (New) (Refresh)

Site Metrics (Run)

ExampleDash (Run)

Dashboard Name: ExampleDash Modify Delete	
Label	Example Dashboard
Dataset	summary.daily_total_day

The node **Metrics (New) (Category)** appears below the dashboard name. If you have entered the metrics during the creation of the dashboard, then you will also see those metrics listed below the **Metrics (New) (Category)** node.

SAS Web Analytics Administrator

Web Analytics by SAS

Web Marts Data Traffic Summaries Scorecards **Dashboards** Segmentations Help

Select a Web Mart (Refresh)

BMC Public Go

Dashboards (New) (Refresh)

- Site Metrics (Run)
- ExampleDash (Run)
- Metrics (New) (Category)**
 - New Category
 - Err 302 Count
 - Err 400 Count
 - Err 500 Count

Dashboard Name: ExampleDash Modify | Delete

Label	Example Dashboard
Dataset	summary.daily_total_day

Note: If no metrics have been specified for the dashboard, then see “Adding a New Metric to a Dashboard” on page 168 to add new metrics to a dashboard. △

See also “Specifying the Information for a Metric in a Dashboard” on page 164.

- 2 Select a metric from the list. The metadata table for the metric appears on the right.

SAS Web Analytics Administrator

Web Analytics by SAS

Web Marts Data Traffic Summaries Scorecards **Dashboards** Segmentations

Select a Web Mart (Refresh)

BMC Public Go

Dashboards (New) (Refresh)

- Site Metrics (Run)
- ExampleDash (Run)
- Metrics (New) (Category)**
 - New Category
 - Err 302 Count
 - Err 400 Count
 - Err 500 Count

Metric: Err 302 Count Modify | Delete

Variable	err_302_count
Label	Err 302 Count
Category	New Category
Business Direction	UP
Link	

- 3 Select **Modify** in the upper right corner of the metadata table. The Dashboard Form appears on the right.

Modify the values for each requested field as needed. The fields are described in “Specifying the Information for a Metric in a Dashboard” on page 164.

When you are finished, click **Update**. Click **Continue**. The metadata table for the metric reappears on the right.

- 4 To define another metric for the dashboard, repeat these steps, starting at step 2.

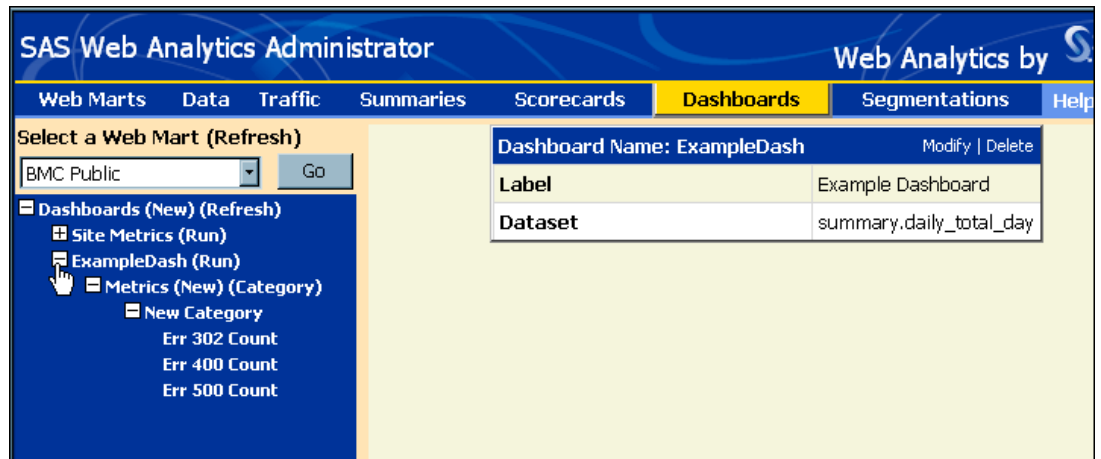
Adding a New Metric to a Dashboard

To add a new metric to a dashboard, follow these steps:

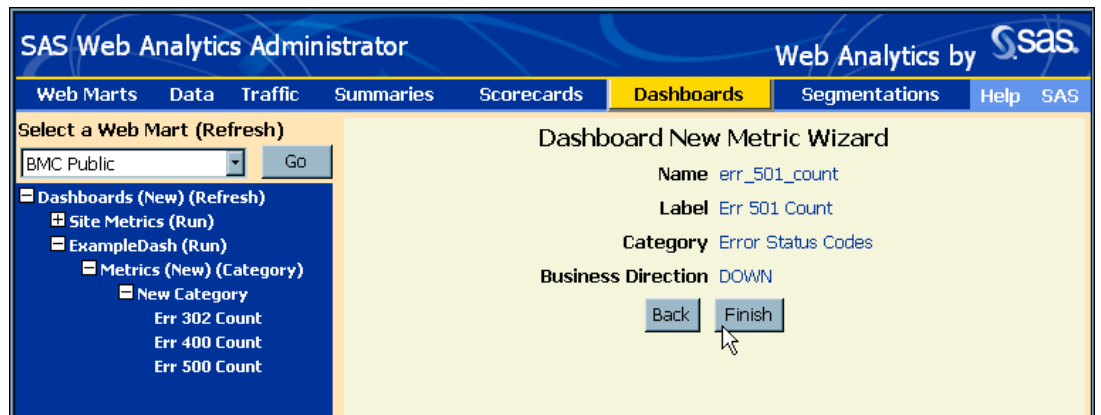
- 1 Click + next to the dashboard of interest to expand the node in the navigation tree of the Dashboards page.

Dashboard Name: ExampleDash Modify Delete	
Label	Example Dashboard
Dataset	summary.daily_total_day

The node **Metrics (New) (Category)** appears below the dashboard of interest.



- 2 Select **New** next to **Metrics**. The Dashboard New Metric Wizard opens. Provide the values for the fields that are required (see “Specifying the Information for a Metric in a Dashboard” on page 164).



- 3 Click **Finish**. Click **Continue**.

Testing the Dashboard Output: The %WADECIDE Macro

To view the new dashboard, run the %WADECIDE macro within a SAS interactive session as described below.

The following block of code can be used to test a dashboard:

```
libname devlib 'c:\xxx';
%wdecide(program=dashboard,
         date_1='1jun2004'd,
         date_2='30jun2004'd,
         swamart=c:\xxx\swamart,
         temp_store= c:\xxx\swamart\temp,
         indsn=devlib.testdata,
         outlib=devlib,
         dashboard_to_run=Example Dashboard
        );
proc print data=devlib.dashboard_data
          (where=('1jun2004'd le date le '30jun2004'd));
run;
proc print data=devlib.dashboard_metric_history
          (where=('1jun2004'd le date le '30jun2004'd));
run;
```

Explanation of the Example Code for the Dashboard

Here is the explanation of the preceding block of code:

- Use data that ranges from June 1, 2004 to June 30, 2004.
- Specify that the root directory of the SAS Web Analytics Web mart is at **c:\xxx\swamart**.
- Store intermediate results in the **c:\xxx\swamart\temp** directory (the location of the Worklib library).
- Use the Devlib.Testdata data set as the data source.

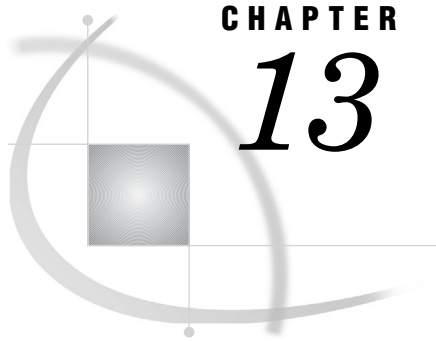
Note: The Devlib library has been defined separately from the %WADECIDE macro. △

- Store the dashboard report in the Devlib library.

Note: When you provide an argument for the Outlib parameter, the output will not be stored in the default Summary library and you will not be able to see the results using the SAS Web Analytics Report Viewer. To remedy this, two PROC PRINT steps have been added after the invocation of the %WADECIDE macro. △

- Create the dashboard whose name is *Example Dashboard*.
- Display the dashboard report in the Output Window using PROC PRINT.
- Display the historical values for each metric in the Output Window using PROC PRINT.

For more information about the arguments for the %WADECIDE macro, see “The %WADECIDE Macro” on page 180.



Setting Up a Segmentation Report

<i>Data Requirements for the Default Segmentation Report</i>	171
<i>Creating a Segmentation Report</i>	171
<i>Using the Segmentation Wizard</i>	171
<i>Testing a Segmentation Report</i>	174
<i>Explanation of a Code Block That Tests a Segmentation Report</i>	174
<i>Running a Segmentation Report</i>	174

Data Requirements for the Default Segmentation Report

The default segmentation report uses the data set called `Unique_Visitor_Segmentation`, which contains visitor-level information by day. Additional data can be added to this data set or to a new data set that contains the variables that are listed in the example scenario (with a single record for each date).

Note: For the default segmentation report analysis, the target variable must have no more than four levels. △

Creating a Segmentation Report

Before you begin to create a new segmentation report, you need to determine the following input information:

- type of input.
 - target — a variable that is used to determine site performance. For example, a target can consist of whether or not a visitor is a repeat visitor, or whether a visitor has made a purchase. Only one target is permitted.
 - metrics — a set of variables that have the potential to affect the target.
- measure level — the type of value contained within the input or target variable.
 - nominal — the value is an alphanumeric string such as either 0 or 1, or red, white and blue.
 - ordinal — the value is an integer or ordered character (A,B,C, and so on).
 - interval — the value is a range of numbers (which can include fractions).

Using the Segmentation Wizard

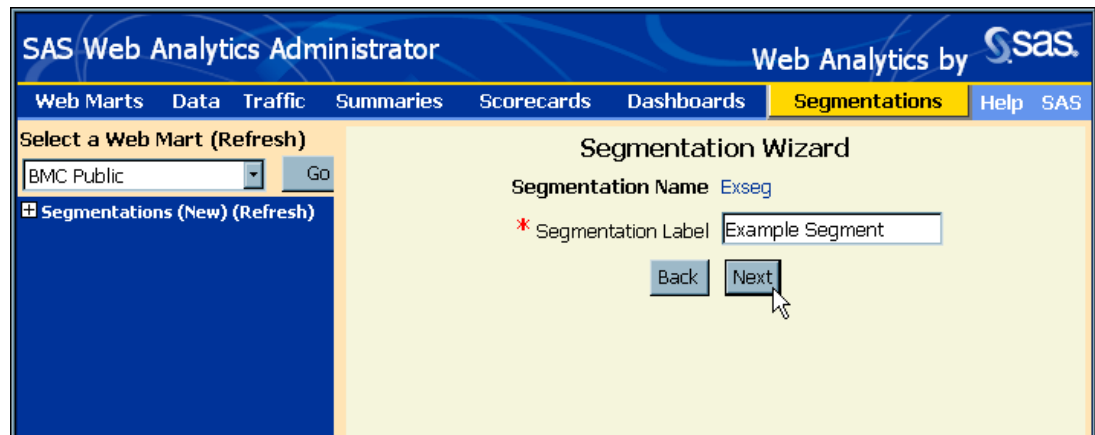
You use the Segmentation Wizard to define a set of inputs that are used by decision tree analysis. This analysis creates a set of *segments* that can be used to define the visitors to the Web site. Follow these instructions to create a segmentation report:

- 1 If a Web mart is not already selected, select a Web mart from the pull-down menu in the upper left corner of the SAS Web Analytics Administrator, and click **Go**.
- 2 Select the **Segmentations** link in the menu bar so that it is highlighted.
- 3 Click **New** next to **Segmentations** in the navigation tree on the left.



The Segmentation Wizard displays on the right.

- 4 Type the segmentation name. Click **Next**.
- 5 Type the segmentation label. Click **Next**.



The segmentation name and the segmentation label can each be up to 25 characters long. The segmentation name can be the same as the segmentation label, but they are used differently. The segmentation name is how the SAS Web Analytics software identifies a segmentation. The segmentation label is how the segmentation appears in the navigation tree of the SAS Web Analytics Administrator and the SAS Web Analytics Report Viewer.

- 6 Select a library from the drop-down menu.

SAS Web Analytics Administrator

Web Analytics by sas

Web Marts Data Traffic Summaries Scorecards Dashboards **Segmentations** Help SAS

Select a Web Mart (Refresh)

BMC Public Go

- Segmentations (New) (Refresh)
 - Repeat Visitor - Totals (Run)
 - Repeat Visitor - Averages (Run)
 - Example Segment (Run)
 - Metrics (New)
 - Repeat Visitor (Target)
 - Visitor Id (Visitor Id)
 - New Visitor (New Visitor)
 - Rptdate Visitor (Most Current Visitor)
 - Duration Anl

Segmentation Wizard

Segmentation Name Exseg

Segmentation Label Ex ample Segment

* Library ☐ Show All

Back

The default list contains the Dated and Summary libraries by default. If your data set is not in one of these libraries, then select the **Show All** option to view an expanded list of libraries. Click **Next**.

- 7 Select a data set from the drop-down menu on the left. Click **Next**.
- 8 Select a target variable from the drop-down menu. Click **Next**.
- 9 Select the visitor identification variable from the drop-down menu. Click **Next**.
- 10 Type an integer value for the number of segments to create. Click **Next**.
- 11 Enter the name of the program that contains the customized code. The program that is identified in this text box needs to exist in the SAS subdirectory of the Web mart before you can run the segmentation. Click **Next**.
- 12 Select the most current visitor from the drop-down menu. Click **Next**.

SAS Web Analytics Administrator

Web Analytics by sas

Web Marts Data Traffic Summaries Scorecards Dashboards **Segmentations** Help SAS

Select a Web Mart (Refresh)

BMC Public Go

- Segmentations (New) (Refresh)
 - Repeat Visitor - Totals (Run)
 - Repeat Visitor - Averages (Run)
 - Example Segment (Run)
 - Metrics (New)
 - Repeat Visitor (Target)
 - Visitor Id (Visitor Id)
 - New Visitor (New Visitor)
 - Rptdate Visitor (Most Current Visitor)
 - Duration Anl
 - Page Count Anl
 - Session Count Anl

Segmentation Wizard

Segmentation Name Exseg

Segmentation Label Example Segment

Library summary

Dataset unique_visitor_segmentation

Target repeat_visitor

Metrics duration_anl, page_count_anl, session_count_anl

Visitor visitor_id

Number of Segments to Create 4

Data Prep Include Code wauvisit.sas

New Visitor Indicator new_visitor

Most Current Visitor rptdate_visitor

Back Next

Click **Finish**. Click **Continue**.

- 13 Run the %WADECIDE macro to test the new segmentation report.

Testing a Segmentation Report

The %WADECIDE macro is the mechanism for testing a new segmentation report. The %WADECIDE macro enables you to create an analytic segmentation report without running the entire ETL process (%WAETL macro).

You can use the following block of code to test a segmentation report:

```
%wadehide(program=segmentation,
           date_1='30jun2004'd,
           swamart=c:\xxx\swamart,
           temp_store= c:\xxx\swamart\temp,
           definition_to_run=Exseg
           );
proc print data=summary.autoseg_segment_rules
           (where=(segmentation_name eq 'Exseg'));
run;
```

Explanation of a Code Block That Tests a Segmentation Report

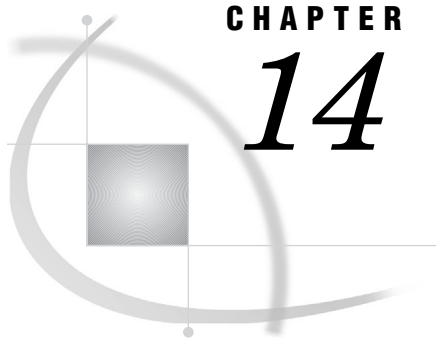
The previous block of code is an example of how you can use the %WADECIDE macro to test a segmentation report. An explanation of the lines of code follows:

- Create the Exseg segmentation report. (The label for the Exseg segmentation report is Example Segment.)
- Use the data for June 30, 2004.
- The root of the SAS Web Analytics Web mart is at **C:\xxx\swamart**.
- Store the intermediate results in the **C:\xxx\swamart\temp** directory (the location of the Worklib library).
- Display the Exseg segmentation report in the output window using PROC PRINT.

See “The %WADECIDE Macro” on page 180 for more information about the arguments and the uses of the %WADECIDE macro.

Running a Segmentation Report

Your custom segmentation report will automatically run the next time that the %WAETL macro runs.



Using Stored Processes in the SAS Web Analytics Application

<i>Working with Stored Processes for Report Definitions</i>	175
<i>The Stored Process for the Funnel Report</i>	175
<i>How to Add a Stored Process</i>	176
<i>Testing a Stored Process</i>	177

Working with Stored Processes for Report Definitions

The SAS Web Analytics application invokes stored processes that are associated with report definitions. Each stored process receives input parameters and sets output parameters by using macro variables. The final output of the stored process is a table that is read by the the SAS Web Analytics application. The SAS Web Analytics application uses this output table as the input for a report. The stored process communicates the name of the output table back to the SAS Web Analytics application by setting the output parameter `OUTPUT_DATASET` to the name of the output table.

The stored process must check to see whether the macro variable `&INFO` is defined and whether it is set to `true`. If the value of `&INFO` is `true`, then the stored process must create an empty table in the `Stp` library. This template table must have a definition that is identical to the actual output table. The stored process will also assign the output parameter `OUTPUT_DATASET` to the name of this template table.

The Web application will also pass the libraries that are defined in the data source libraries to the stored process. The input parameter `&LIBREFS` will contain a comma-separated list of the SAS libnames that the stored process will need to define. The input parameter `&LIBPATHS` is a comma-separated list of the respective paths for the library names. The Web application will also pass the defined parameters as macro variables to the stored process in order to display the values in the report that have been set by the user.

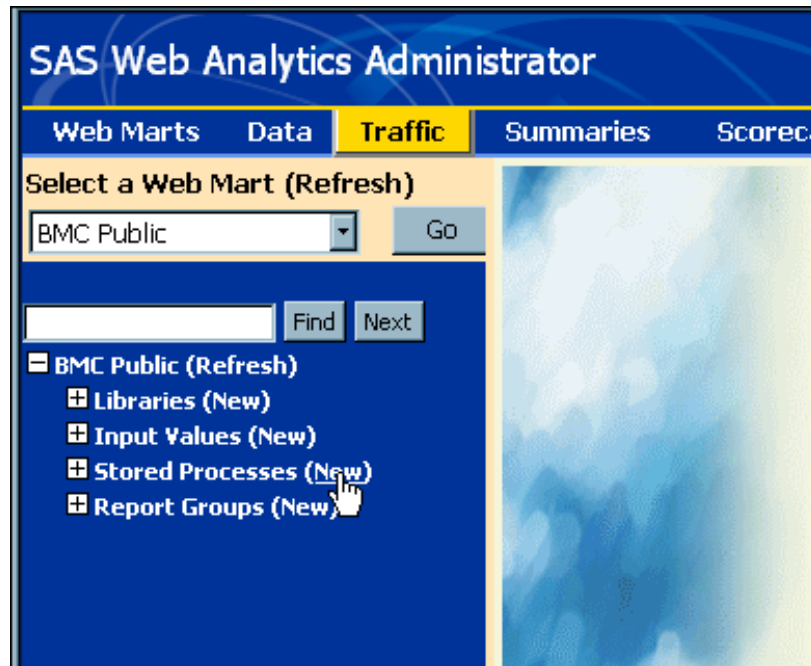
The Stored Process for the Funnel Report

SAS Web Analytics software by default includes the `Stp_wafunnel` stored process that invokes the `%WAFUNNEL` macro. The default funnel report in SAS Web Analytics is the output from the execution of the `Stp_wafunnel` stored process.

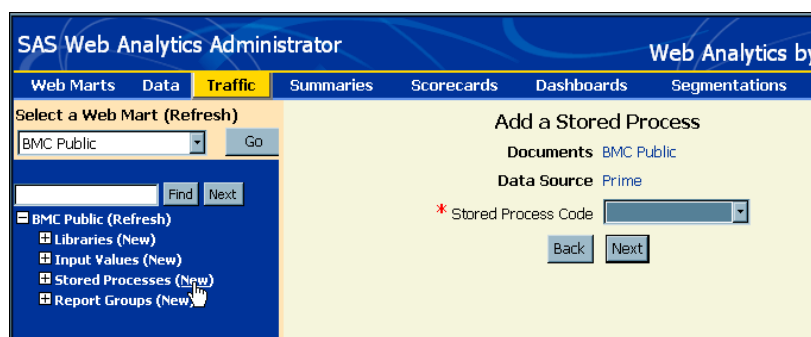
How to Add a Stored Process

You can add a custom SAS stored process (that you have developed) as a step in the process of associating a stored process with a SAS Web Analytics report. To add a stored process, follow these steps:

- 1 Open the SAS Web Analytics Administrator. Select the **Traffic** link in the banner.
- 2 Select a Web mart (if it is not already selected) and click **Go**.
- 3 In the navigation tree, click + to expand the Web mart.
- 4 Select **New** next to **Stored Processes**.



The Add a Stored Process form opens.



- 5 The **Documents** and **Data Source** fields are automatically supplied. From the drop-down menu, select the name of the SAS stored process code that you want to associate with a report definition. Click **Next**.

Note: The stored process code must reside in the directory that is specified by the **stored process repository** field in the definition for the selected Web Mart. You can find the location of your stored process repository in the Web marts page. △

- 6 Type a name for the stored process in the **Enter Label** field. The name that you type here will be added to the navigation tree. Click **Next**.
- 7 Type a description for the stored process in the **Enter Description** field. Click **Next**.

The screenshot shows the SAS Web Analytics Administrator interface. The top navigation bar includes 'Web Marts', 'Data', 'Traffic' (selected), 'Summaries', 'Scorecards', 'Dashboards', 'Segmentations', 'Help', and 'SAS'. On the left, under 'Select a Web Mart (Refresh)', 'BMC Public' is selected. Below this, a list of items is shown: 'BMC Public (Refresh)', 'Libraries (New)', 'Input Values (New)', 'Stored Processes (New)', and 'Report Groups (New)'. The main area is titled 'Add a Stored Process'. It contains the following fields: 'Documents' (BMC Public), 'Data Source' (Prime), 'Stored Process Code' (stp_wafunnel.sas), 'Label' (Custom Stored Process), and 'Description' (a custom code similar to the swa code). At the bottom of the form are 'Back' and 'Finish' buttons. A mouse cursor is pointing at the 'Finish' button.

Click **Finish**. Click **Done**. The new stored process report appears in the navigation tree.

- 8 Expand the node of the new stored process report. Click **New** next to **Parameters** in the navigation tree. On the right, in the **Enter name** field, type the name of the input macro variable. In the **Enter the value** field, enter either a value or the macro variable `$name`.

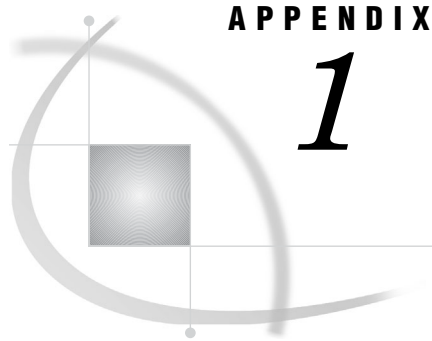
Note: If the name equals the name of the input macro variable, then the user will be prompted (at run time) to enter a value for the given input macro variable. If the name matches an input value that was defined under **Input Values** for your group, and this input value points to a data set, then the user will be presented with a drop-down menu from which to select a value which is populated from that data set. △

- 9 When you create a new report definition, you can select a stored process (from the stored processes that were previously defined) from the drop-down menu.

Testing a Stored Process

You can test a defined stored process by clicking **Run** next to the name of the stored process in the navigation tree on the left in the SAS Web Analytics Administrator. Clicking **Run** will execute the stored process and will display the results in the main frame of the SAS Web Analytics Administrator.

When a report that includes a stored process is run in the SAS Web Analytics Report Viewer, any associated stored processes will be executed before the report is displayed.



APPENDIX

1

Macro Reference for the SAS Web Analytics 5.1 Solution

<i>Overview: Macros for SAS Web Analytics 5.1</i>	179
<i>The %WADECIDE Macro</i>	180
<i>Purpose of the %WADECIDE Macro</i>	180
<i>The %WADECIDE Macro: Syntax</i>	180
<i>The %WADECIDE Macro: Details</i>	180
<i>The %WADECIDE Macro: Notes</i>	183
<i>The %WADECIDE Macro: Examples</i>	184
<i>The %WAETL Macro</i>	184
<i>Purpose of the %WAETL Macro</i>	184
<i>The %WAETL Macro: Syntax</i>	185
<i>The %WAETL Macro: Details</i>	185
<i>Using the %WAETL Macro to Initialize a New Web Mart</i>	186
<i>Using the %WAETL Macro to Load Data into an Existing Web Mart</i>	187
<i>Using the Daily.sas Program</i>	187
<i>How to Code Your Custom Invocation of the %WAETL Macro</i>	188
<i>Custom Invocation of the %WAETL Macro: Example 1</i>	188
<i>Custom Invocation of the %WAETL Macro: Example 2</i>	188
<i>Custom Invocation of the %WAETL Macro: Example 3</i>	188

Overview: Macros for SAS Web Analytics 5.1

This chapter is a reference for the major macros that are used in SAS Web Analytics 5.1. Arguments are not case-sensitive, unless specified as such.

The SAS Web Analytics solution provides the following macros:

- “The %WADECIDE Macro” on page 180
- “The %WAETL Macro” on page 184

The %WADECIDE Macro

Purpose of the %WADECIDE Macro

The %WADECIDE macro creates a decision support report (scorecard, dashboard, or segmentation) without requiring the execution of the entire SAS Web Analytics ETL process (%WAETL).

Why invoke a decision support report macro outside %WAETL?

It is not always convenient to wait for the next execution of the %WAETL macro in order to view a decision support report. For instance, if you are creating a new scorecard, it can be cumbersome to execute the %WAETL macro for every test you want to perform during the development process.

Also, by running the %WADECIDE macro, you can store provisional scorecards and dashboards in a test library instead of in the **summary** directory, where the %WAETL macro stores them. By isolating the test versions in a separate library, you can avoid cluttering up the **summary** directory and possibly mistaking a work-in-progress for actual decision support reports.

Finally, the %WADECIDE macro enables you to populate a newly defined scorecard or dashboard with the results of previous ETL processes. For example, if you define a scorecard but you want it to display the results as though they had existed for a while, then you would invoke the %WADECIDE macro.

The %WADECIDE Macro: Syntax

```
%WADECIDE(
  PROGRAM=program-name,
  DATE1=SASdate,
  <DATE2=SASdate>,
  SWAMART=path-to-mart,
  <TEMP_STORE=path-to-tempstore>,
  <INDSN=dataset-name>,
  <OUTLIB=libref-name>,
  <USE_FIRST_TWO_WEEKS=YES|NO>,
  <SCORECARD_TO_RUN=scorecard-name>,
  <DASHBOARD_TO_RUN=dashboard-name>,
);
```

The %WADECIDE Macro: Details

PROGRAM=program-name

specifies the name of the decision support report that you want to create. Valid values are:

- SCORECARD
- DASHBOARD
- SEGMENTATION

Note: You can only create one type of decision support report for each invocation of the %WADECIDE macro. If you want to create more than one type of report, you must invoke the %WADECIDE macro more than once. △

DATE1=SASdate

specifies the first date in your request, expressed as a SAS date literal, such as '01jan2004'd. For information about how to format SAS date literals, see SAS Help and Documentation.

If PROGRAM=DASHBOARD or PROGRAM=SCORECARD, then DATE_1 is the first date in the date range for the report; if PROGRAM=SEGMENTATION, then DATE_1 is the date for the report.

DATE2=SASdate

specifies the second date in your request, expressed as a SAS date literal, such as '01jan2004'd. For information about how to format SAS date literals, see SAS Help and Documentation.

If PROGRAM=DASHBOARD or PROGRAM=SCORECARD, then DATE_2 is the last date in the date range for the report. If you do not specify DATE_2, DATE_1 will be used as the default value for this argument.

If PROGRAM=SEGMENTATION, DATE_2 is ignored if it has been specified.

SWAMART=path-to-mart

specifies the path to the root directory for the designated SAS Web Analytics Web mart. The %WADECIDE macro uses this information to locate the data resources that are necessary to create the decision support report that you requested. For example, if your SAS Web Analytics Web mart is located at **c:\xxx\swamart** and its subdirectories, then the root directory is **c:\xxx\swamart**.

TEMP_STORE=path-to-tempstore

specifies the path to a directory that will be your Worklib library, where the %WADECIDE macro will store the intermediate data sets that it creates in the process of building a decision support report.

This path must exist before you run the %WADECIDE macro; it is not created for you.

Specify this argument if you want to be able to access these intermediate data sets after the %WADECIDE macro has completed. If you do not specify this argument, the %WADECIDE macro creates its own temporary Worklib library, which exists only as long as the SAS session in which you invoke the %WADECIDE macro.

INDSN=dataset-name

specifies the libname.memname for the data set that will be used to create the decision support report. If you have specified PROGRAM=DASHBOARD or PROGRAM=SCORECARD, the %WADECIDE macro uses this information. If you have specified PROGRAM=SEGMENTATION, then the %WADECIDE macro ignores this information.

If you do not specify this argument, the %WADECIDE macro uses the libname.memname that is stored in the INDSN field in the Config.Wadshbrd data set if PROGRAM=DASHBOARD or in the Config.Wascrcrd data set if PROGRAM=SCORECARD.

OUTLIB=libref-name

specifies the libref where you want the %WADECIDE macro to store its decision support report. If you have specified PROGRAM=DASHBOARD or PROGRAM=SCORECARD, the %WADECIDE macro uses this information. If you have specified PROGRAM=SEGMENTATION, the %WADECIDE macro ignores this information.

If you do not specify this argument, the %WADECIDE macro stores the decision support report in the SUMMARY library.

Do not specify this argument if you are running a scorecard or dashboard for a day or for a range of days that you want to access by the SAS Web Analytics Report Viewer.

If you want to store the scorecard or the dashboard elsewhere—for example, to store separate test versions of scorecards or dashboards as you are developing them—then specify an alternate library for this argument. Note that this library must be already defined before executing the %WADECIDE macro.

USE_FIRST_TWO_WEEKS=YES|NO

specifies whether to use the first 14 days of data to create the decision support report.

Code this value based on the type of data in your scorecard. Specify `USE_FIRST_TWO_WEEKS=NO` if the target variable for your scorecard has measures that need some accumulated history (for example, repeat visitors). You can create a model with a better fit by excluding the data of the first 14 days.

If not specified, the %WADECIDE macro uses the default `USE_FIRST_TWO_WEEKS=YES` which includes the data of the first 14 days.

Note: This parameter is used only if `PROGRAM=SCORECARD` is specified. If `PROGRAM=DASHBOARD` or `PROGRAM=SEGMENTATION` is specified, this parameter is ignored. △

SCORECARD_TO_RUN=scorecard-name

specifies the name of the scorecard that you want to create.

If specified, the %WADECIDE macro creates only the scorecard that you have identified. If not specified, the %WADECIDE macro creates all the scorecards that are defined in the `Config.Scrprd` data set.

Note: This parameter is used only if `PROGRAM=SCORECARD` is specified. If `PROGRAM=DASHBOARD` or `PROGRAM=SEGMENTATION` is specified, this parameter is ignored. △

DASHBOARD_TO_RUN=dashboard

specifies the name of the dashboard that is to be created. The dashboard must be defined in the dashboard configuration data set. Only the dashboard that is specified will be processed. If no dashboard is specified, then all the dashboards listed in the `Config.Wadshbrd` data set will be created.

Note: This parameter is used only if `PROGRAM=DASHBOARD` is specified. If `PROGRAM=SCORECARD` or `PROGRAM=SEGMENTATION` is specified, this parameter is ignored. △

The following table summarizes how the %WADECIDE macro uses arguments for each decision support report.

Note: “N/A” means “Not Applicable” and indicates that you do not have to specify the argument when invoking the %WADECIDE macro.

Italics indicate that the provided value is an example and you should replace it with a value that you choose. △

Table A1.1 Arguments for Decision Support Reports

Argument	Value for Dashboard	Value for Scorecard	Value for Segmentation
PROGRAM (Required)	DASHBOARD	SCORECARD	SEGMENTATION
DATE_1 (Required)	<i>'01jan2004'd</i>	<i>'01jan2004'd</i>	<i>'01jan2004'd</i>

Argument	Value for Dashboard	Value for Scorecard	Value for Segmentation
DATE_2 (Optional)	'31jan2004'd	'31jan2004'd	N/A
SWAMART (Required)	c:\xxx\swamart	c:\xxx\swamart	c:\xxx\swamart
TEMP_STORE (Optional)	c:\xxx\swamart\temp	c:\xxx\swamart\temp	c:\xxx\swamart\temp
INDSN (Optional)	summary.daily_total_day	summary.daily_total_day	N/A
OUTLIB (Optional)	devlib	devlib	N/A
USE_FIRST_TWO_WEEKS (Optional)	N/A	YES or NO	N/A
SCORECARD_TO_RUN (Optional)	N/A	Trial Scorecard	N/A
DASHBOARD_TO_RUN (Optional)	Trial Dashboard	N/A	N/A

The %WADECIDE Macro: Notes

The %WADECIDE macro performs error checking on the arguments that you specify and writes error messages in the log if it detects any problems. The format for these messages is as follows:

```
ERROR:(WADECIDE) hh:mm:ss [text of message]
```

where **hh:mm:ss** is the time when the %WADECIDE macro ran. The text of the message indicates the type of problem that the %WADECIDE macro detected.

If the text inside the parentheses is not “WADECIDE,” then it will be the name of the module that creates the decision support report that you have requested. This means that the error occurred in the decision support report module, not in the %WADECIDE macro.

The %WADECIDE Macro: Examples

Example 1:

This code creates the Segmentation Report for 5/28/2004 for the SAS Web Analytics Web mart whose root is at **c:\xxx\swamart**.

```
%wadehide(
  program=segmentation,
  date_1='28may2004'd,
  swamart=c:\xxx\swamart
);
```

Example 2

This code creates the Temp Dev Dashboard Report for 5/1/2004 through 5/31/2004 for the SAS Web Analytics Web mart whose root is at **c:\xxx\swamart**, storing intermediate data sets in the SAS Web Analytics Web mart whose root is at **c:\xxx\swamart\TempStore** directory.

```
%wadehide(
  program=dashboard,
  date_1='01may2004'd,
  date_2='31may2004'd,
  swamart=c:\xxx\swamart,
  temp_store=c:\xxx\swamart\TempStore,
  dashboard_to_run=Temp Dev
);
```

Example 3

This code creates all scorecard reports for 6/1/2004 for the SAS Web Analytics Web mart whose root is at **c:\xxx\swamart** using Devlib.Testdata as the data source. It writes the reports to the Devlib library. The report excludes the data from the first 14 days.

```
%wadehide(
  program=scorecard,
  date_1='01jun2004'd,
  swamart=c:\xxx\swamart,
  indsn=devlib.testdata,
  outlib=devlib,
  use_first_two_weeks=no
);
```

The %WAETL Macro

Purpose of the %WAETL Macro

The %WAETL macro is the central component for the SAS Web Analytics application. The %WAETL macro has the following primary functions:

- 1 to initialize a new Web mart
- 2 to load and manage the data for an existing Web mart.

Each of these functions has its own invocation of the %WAETL macro.

The %WAETL Macro: Syntax

The %WAETL macro was designed to be flexible so that you can invoke it in several ways to achieve the same results. You can use different sets of arguments to provide the macro with the information that it needs. The following is the syntax for the %WAETL macro:

```
% WAETL(
    <NAME=web-mart-name>,
    <DATA_STORE=path-to-datastore>,
    <TEMP_STORE=path-to-tempstore>,
    <DETAIL=path-to-libname.memname>,
    <SWAMART=path-to-mart>,
);
```

The % WAETL Macro: Details

NAME=*Web-mart name*

specifies the name of your SAS Web Analytics Web mart (this argument does not need to be part of your invocation if you provide the SWAMART= argument). If you provide this argument, it must match the spelling of the Name field in an observation of both Sasuser.Wainit and Sasuser.Wbinit, but is not case-sensitive.

Note: Do not use this argument if you are invoking the %WAETL macro with PROGRAM=INITIALIZE. △

DATA_STORE=*path-to-datastore*

specifies the path to the root directory of the SAS e-Data ETL application. If you are using a customized directory structure (i.e., your SAS e-Data ETL directories are outside your SAS Web Analytics Web mart directory structure), this argument is the mechanism for letting the %WAETL macro know where these other SAS e-Data ETL directories are located. If you are using the default directory structure for your SAS Web Analytics Web mart, you do not need to include this argument in your invocation. If you provide this argument, then it can be a quoted string or an unquoted string; both '**c:\xxx\data_store**' and **c:\xxx\data_store** are valid forms for this argument.

TEMP_STORE=*path-to-tempstore*

specifies the path to a directory that will act as your Worklib library, where the %WAETL macro will store the intermediate data sets that it creates. This path must exist prior to running the %WAETL macro because the path will not be created for you. If you want to have access to these data sets after the SAS session in which you invoke the %WAETL macro has finished, then you should provide information for this argument. If you do not specify this argument, then the %WAETL macro creates its own Worklib library, which will exist only as long as the SAS session in which you invoke the %WAETL macro. If you provide this argument, it must be an unquoted string (for example, **c:\xxx\temp_store** is a valid form, but '**c:\xxx\temp_store**' is not a valid form for this argument).

DETAIL=*path-to-libname.memname*

specifies the SAS libname.memname for the parsed SAS e-Data ETL data set. If you are using the defaults for your SAS Web Analytics Web mart, you do not need to include this argument in your invocation. If you do use this argument, before you invoke the %WAETL macro, you must have already identified the libname with a LIBNAME statement or some other technique. See SAS Help and Documentation for a comprehensive list of how you can do this.

SWAMART=*path-to-mart*

specifies the path to the root directory for the designated SAS Web Analytics Web mart (this argument does not need to be part of your invocation if you provide the NAME= argument). This information is needed so that the %WAETL macro can locate the appropriate data resources (for example, if your SAS Web Analytics Web mart is located at **c:\xxx\swamart** and its sub-directories, then the root directory is **c:\xxx\swamart**). If you provide this argument, it can be a quoted or an unquoted string (for example, both '**c:\xxx\swamart**' and **c:\xxx\swamart** are valid forms for this argument).

PROGRAM=*program-name*

specifies the function that you want the %WAETL macro to perform: either *initialize* or *warehouse* (the argument must match the spelling of one of these two words, but is not case-sensitive).

Using the %WAETL Macro to Initialize a New Web Mart

To initialize a new Web mart, use the following options to invoke the %WAETL macro:


```
%waetl(program=initialize,
        swamart= <webmartname>
        );
```

PROGRAM=INITIALIZE

specifies the %WAETL macro function for setting up a new SAS Web Analytics Web mart. The argument must match the spelling, but is not case-sensitive.

SWAMART=*<webmartname>*

specifies the path to the root directory of the SAS Web Analytics Web mart that you are setting up. Because the %WAETL macro uses this information to create the directory that you have specified, the directory must not already exist. After creating this directory, the %WAETL macro creates the default set of subdirectories that need to be in place before it can load data into the Web mart.

Note: If you want your Web mart to have a customized directory structure, then you need to create it by hand. The %WAETL macro cannot create a customized directory structure for you. 

The %WAETL macro also creates the metadata that will control the standard operations in the Web mart. Finally, it creates some program stubs that will help you use the Web mart in the future. One of these stubs, Daily.sas, which is created in the \sas subdirectory, is a program you can use for all subsequent invocations of the %WAETL macro for the Web mart you have just created.

Using the %WAETL Macro to Load Data into an Existing Web Mart

Once you have initialized a new Web mart, you can load data into it either by using the Daily.sas program or by coding your own invocation of the %WAETL macro.

Using the Daily.sas Program

The Daily.sas program (see the default code below) is created in the \sas subdirectory of your SAS Web Analytics Web mart when you run the %WAETL macro to initialize it. Before you can use the Daily.sas program, you will need to modify it according to the instructions in the program's header block. If you have a customized directory structure, you will also need to modify the arguments in the program's invocation of the %WAETL macro.

```
/**
 * Use this program to process weblog data into the SAS Web Analytics warehouse.
 *
 * IMPORTANT: You must supply the location of the weblogs to process.
 *
 *   Either
 *
 *   A. Specify the location in the WEBLOGS macro variable below by replacing
 *       with the location of your weblogs. WEBLOGS can
 *       be the full path either for a single weblog or for a directory that
 *       contains multiple weblogs.
 *
 *   Or
 *
 *   B. Launch the e-Data ETL administrative GUI from an interactive SAS session
 *       using the "edataetl" command and specify the location of your weblog(s)
 *       in the appropriate field.
 *
 *   If you use method B. then you must remove the "weblog_path=&weblogs" parameter
 *   from the %edataetl call below.
 */

%MACRO SWA_ETL;

*** TO DO: specify the location of your weblogs;
%let WEBLOGS=;

%let webmart=c:\xxx\swamart;

%let wbrc=0;

/* Extract raw weblog data into staging datasets. */
%edataetl( data_store   = &webmart\e-data-etl,
           weblog_path  = &weblogs,
           program      = extract
         );

/* Load web data into final detail dataset. */
%edataetl( data_store   = &webmart\e-data-etl,
           program      = load
         );

/* Load web detail data into warehouse. */
%if &wbrc = 0 %then %do;
```

```

        %waetl( swamart = &webmart,
                program = warehouse
                );

%end;

%MEND SWA_ETL;
%SWA_ETL;

```

How to Code Your Custom Invocation of the %WAETL Macro

If the `Daily.sas` program is inadequate for loading data into your SAS Web Analytics Web mart, then you will need to code your own invocation of the %WAETL macro. If you find yourself in this situation, it will be because you have performed extensive customizations for your Web mart. The exact nature of your customizations will determine how you will need to set up your invocation of the %WAETL macro. The following examples offer some suggestions for setting up your invocation of the %WAETL macro.

Custom Invocation of the %WAETL Macro: Example 1

Load the standard parsed Web log from the standard SAS e-Data ETL application directory for the **swamart** Web mart, storing the intermediate data sets in the existing **c:\xxx\swamart\tempstore** directory.

```

%waetl(name=swamart,
       temp_store=c:\xxx\swamart\tempstore,
       program=warehouse
       );

```

Custom Invocation of the %WAETL Macro: Example 2

Load the customized parsed Web log called **custom_log_detail** from the customized SAS e-Data ETL application directory (**c:\custom\e_data_etl**) for the Web mart whose root is at **c:\xxx\swamart**, storing intermediate data sets in the default temporary directory.

```

libname custom 'c:\custom\e-data-etl\detail';
%waetl(detail=custom.custom_log_detail,
       swamart=c:\xxx\swamart,
       program=warehouse
       );

```

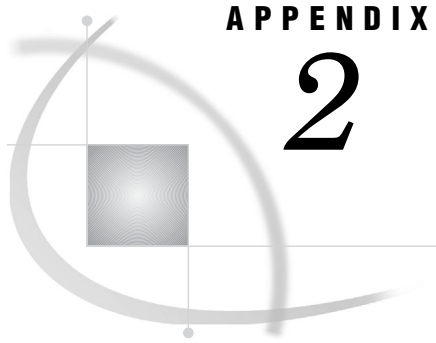
Custom Invocation of the %WAETL Macro: Example 3

Load the customized parsed Web log called **custom_log_detail** from the standard SAS e-Data ETL application library (Detail) for the **swamart** Web mart, storing intermediate data sets in the default temporary directory.

```

libname detail 'c:\xxx\swamart\e-data-etl\detail';
%waetl(name=swamart,
       detail=detail.custom_log_detail,
       program=warehouse
       );

```



APPENDIX

2

Alphabetical List of SAS Web Analytics Reports

List of Traffic Reports 189

List of Non-Traffic Reports 191

Alphabetical List of SAS Web Analytics Reports 191

List of Traffic Reports

The following table describes the traffic reports that are provided by default with SAS Web Analytics software. Any additional traffic reports that you might have created for your organization are not listed.

Table A2.1 Standard Traffic Reports

Type	Report	Description
Visitor	Unique Visitors	Identifies the visitors who have the most activity on your Web site. The report lists each unique visitor whose visit count is greater than one.
Browser and platform	Browsers	Displays a distribution of the different Web browsers that are used by visitors who navigate the Web site.
	Browser Versions	Displays a distribution of the different Web browser versions that are used by visitors who navigate the Web site.
	Platforms	Displays a distribution of the different platforms (operating systems) that are used by visitors who navigate the Web site.
Status codes	Status Codes	Provides information about the frequency that a code was returned by the server to the visitor's browser to report the outcome of a request.
	Hourly Status Codes	Provides information about the frequency of status codes categorized by hour of day.
	Error Status Codes	Provides information about the frequency of errors that occur when visitors attempt to access your Web site.
	Error Status Code Pages	Provides information about individual pages that generated errors.

Type	Report	Description
Navigation	Error Status Code Page Referrers	Provides information about the frequency of status codes for each page that was requested.
	Page Frequency	Displays a distribution of the pages that were requested. This report enables you to identify the pages and family of pages that are most often viewed.
	Entry Pages	Displays the most common requests that visitors make to enter the Web site.
	Referrer by Entry Page	Displays the pages that were requested immediately prior to the requested file.
	Entry Pages by Referrer	Displays the visit activity for the first requested file.
Overview	Exit Pages	Provides a list of all Web pages from which the current site was exited.
	Hourly Metrics	Displays daily basic traffic statistics for the site on an hourly basis.
	Site Metrics	Displays basic traffic statistics for the site.
	Day of Week Metrics	Displays daily basic traffic statistics for the site by day of week.
	Site Metrics by Day of Week	Displays statistics for the site by a particular day of the week.
Referrer	Visit Referrer Domains	Displays a distribution of referrer domains, which enables you to determine the origin of visitors.
	Entry Pages by Referrer	Displays the page and visit information that are associated with the domains.
	Referrer by Entry Page	Displays a list of entry pages for each referrer.
	Search Terms	Displays a distribution of search terms that are used by visitors to find your Web site.
	Referrer by Search Terms	Displays the domains from which visitors searched for specific items of information.
	Search Terms by Referrer	Displays a distribution of search terms that are used by visitors within referrer domains to find your Web site.
	Like Search Terms	Displays a list of search terms that are similar to a term that you specify.

List of Non-Traffic Reports

The following table describes the non-traffic reports that are provided by default with SAS Web Analytics software. Any additional non-traffic reports that you might have created for your organization are not listed.

Table A2.2 Standard Non-Traffic Reports

Category	Report	Description
Scorecard	Site Metrics	Enables you to view the performance and the forecast values of the key performance indicators (KPIs) that are related to traffic on the site that is being analyzed.
Dashboard	Site Metrics	Displays the values for each site's metrics, assigns a performance level to that metric based on its desired business direction, and displays historical information for that metric for a given date.
Funnel	Interactive Funnel	Provides a detailed description of any sequential process on a Web site, such as a sequence of Web pages that are visited.
Path analysis	Entry Paths	Displays a list of the most popular points of entry into the Web site.
	Referrer Entry Path	Displays a list of referrers that sent the most traffic to your Web site.
	Interactive Path Analysis	Displays navigational information about the order in which pages or URIs occur in visits whose visitors navigate the Web site. The report traces a path from any starting page or to any ending page or between any starting and ending pages.
Segmentation	Repeat Visitors - Totals	Displays a list of repeat visitors (totals) to the Web site, based on a set of rules that help predict which visitors will return.
	Repeat Visitors - Averages	Displays a list of repeat visitors (averages) to the Web site, based on a set of rules that help predict which visitors will return.

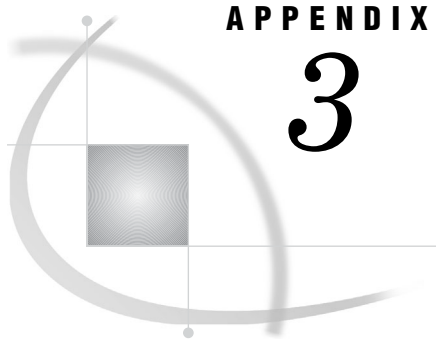
Alphabetical List of SAS Web Analytics Reports

The following table alphabetically lists all of the SAS Web Analytics reports that are provided with SAS Web Analytics software. The first column lists the name of the report. The second column contains the path by which you access the report. The access path always begins at the main menu of the SAS Web Analytics Report Viewer.

Note: The reports are grouped differently in SAS Web Analytics Administrator than in the SAS Web Analytics Report Viewer in order to accommodate the administration of tasks. △

Table A2.3 Alphabetical List of Standard SAS Web Analytics Reports

Report	Access Path
Browser Versions	Traffic ► Browser and Platform ► Browsers ► Browser Versions
Browsers	Traffic ► Browser and Platform ► Browsers
Day of Week Metrics	Traffic ► Overview ► Day of Week Metrics
Entry Pages	Traffic ► Navigation ► Entry Pages
Entry Pages by Referrer	Traffic ► Navigation ► Entry Pages ► Referrer by Entry Page ► Entry Pages by Referrer
Error Status Code Page Referrers	Traffic ► Status Codes ► Error Status Codes ► Error Status Code Pages ► Error Status Code Page Referrers
Error Status Code Pages	Traffic ► Status Codes ► Error Status Codes ► Error Status Code Pages
Error Status Codes	Traffic ► Status Codes ► Error Status Codes
Exit Pages	Traffic ► Navigation ► Exit Pages
Funnel	Funnel
Hourly Metrics	Traffic ► Overview ► Hourly Metrics
Hourly Status Codes	Traffic ► Status Codes ► Status Codes ► Hourly Status Codes
Interactive Path Analysis	Path Analysis ► Path Analysis Reports ► Interactive Path Analysis
Like Search Terms	Traffic ► Referrer ► Like Search Terms
Page Frequency	Traffic ► Navigation ► Page Frequency
Platforms	Traffic ► Browser and Platform ► Platforms
Referrer by Entry Page (Navigation)	Traffic ► Navigation ► Entry Pages ► Referrer by Entry Page
Referrer by Entry Page (Referrer)	Traffic ► Referrer ► Visit Referrer Domains ► Entry Pages by Referrer ► Referrer by Entry Page
Referrer by Search Terms	Traffic ► Referrer ► Search Terms ► Referrer by Search Terms
Repeat Visitor - Averages	Segmentation ► Segmentations ► Repeat Visitors - Averages
Repeat Visitor - Totals	Segmentation ► Segmentations ► Repeat Visitors - Totals
Search Terms	Traffic ► Referrer ► Search Terms
Site Metrics (Dashboard)	Dashboard ► Site Metrics
Site Metrics (Scorecard)	Scorecard ► Site Metrics
Site Metrics (Traffic)	Traffic ► Overview ► Site Metrics
Site Metrics by Day of Week	Traffic ► Overview ► Day of Week Metrics ► Site Metrics by Day of Week
Status Codes	Traffic ► Status Codes ► Status Codes
Unique Visitors	Traffic ► Visitors ► Unique Visitors
Visit Referrer Domains	Traffic ► Referrer ► Visit Referrer Domains



APPENDIX

3

The SUMMARY Engine

<i>The Waadmsum Data Set</i>	193
<i>Variables of the Waadmsum Data Set</i>	194
<i>General Information About SAS Web Analytics Summaries</i>	195
<i>How the SUMMARY Engine Performs an Initial Summarization</i>	196
<i>How the SUMMARY Engine Performs a Re-Summarization</i>	198
<i>Modifying the Waadmsum Metadata Data Set</i>	199
<i>Overview: The Waconfig Data Set</i>	200
<i>Example: Using a Parameter in the Waconfig Data Set to Modify the %WAETL Macro</i>	200
<i>The Parameters in the Waconfig Data Set</i>	200
<i>Changing Values in the Waconfig Data Set</i>	204
<i>How to Edit the Waconfig Data Set</i>	204
<i>Changing the Parameters in the Session Summary Control Category</i>	205
<i>The WAB_INIT_BY_SESS_VARS Parameter for Controlling Daily Detail Data Sets</i>	205
<i>The Parameters for Customizing Session Summary Data Sets</i>	207

The Waadmsum Data Set

The %WAETL macro generates and executes a series of PROC SUMMARY steps to transform the detail data sets (created by the SAS e-Data ETL processes) and the session data sets (derived by %WAETL macro from the detail data sets) into the final summarized form (a SAS summary data set) each time that it runs. The SUMMARY engine module within the %WAETL macro uses the information in the Waadmsum data set to build the PROC SUMMARY steps.

The supplied Waadmsum table has 15 rows; one row for each of the 15 summarization families. Each summarization family has one member for each of its summary levels: day, week, month, quarter, and year. For example, the Hourly Metrics family has the following SAS summary data sets as members:

- Hourly_Metrics_Day
- Hourly_Metrics_Week
- Hourly_Metrics_Month
- Hourly_Metrics_Qtr
- Hourly_Metrics_Year

The SUMMARY engine first expands the Waadmsum data set in order to create a separate definition for all of the individual members of each summarization family. Using this expanded data, the SUMMARY engine then creates a SAS summary data set for each member of every summarization family.

To add a new summary or to change the way that an existing summary is performed, you edit the parameters of the Waadmsum data set. To access the Waadmsum data set,

first select a Web mart. Then select **Data ► Configuration Tables ► Summary Engine Metadata** in the SAS Web Analytics Administrator.

Variables of the Waadmsum Data Set

The variables of the Waadmsum metadata data set contain all the information that the SUMMARY engine needs in order to generate the PROC SUMMARY steps that are executed from within the %WAETL macro.

Note: For more information about PROC SUMMARY, see SAS Help and Documentation. △

The following table lists the variables in the Waadmsum data set:

Table A3.1 The Variables in the Waadmsum Data Set

Variable	Description
Label	Is the name of a summarization family as it appears in the interface of the SAS Web Analytics Administrator.
Source_Table	Is the type of source data set (either session or detail) that provides the information for the summarization family.
Proc_Summary_Options	Are the options that are part of the PROC SUMMARY statement. The most commonly used option in the Waadmsum metadata is SUM MISSING (for more information about PROC SUMMARY options, see SAS Help and Documentation).
Input_Where	Is the WHERE statement that filters the Source_Table data set. The syntax for this field is: (where=<subsetting condition>) The entire WHERE statement must be contained within parentheses. If no filtering is needed, then leave this field blank.
Summary_Level_List	Is a comma-delimited list of the summary levels for the summarization family members. Five possible summary levels are possible: Day, Week, Month, Qtr (quarter), and Year. Each entry in this list becomes the suffix for a corresponding SAS summary data set.
Class_Var_List	Is a comma-delimited list of class variables. For more information about class variables, see the SAS Help and Documentation.
Id_Var_List	Is a comma-delimited list of ID variables. For more information about variables, see the SAS Help and Documentation.
Id_Var_Summary_Level_List	Is a comma-delimited list of the summary levels that apply for the Id_Var_List entries. The five possible summary levels are: Day, Week, Month, Qtr (quarter), and Year.
Analysis_Var_List	Is a comma-delimited list of analysis variables. For more information about analysis variables, see the SAS Help and Documentation. If no analysis variables are needed, then leave this field blank.

Variable	Description
Output_Dsn_Options_0	<p>Are the SAS data set options that you want to apply to the Results_Table data set for the re-summarizing PROC SUMMARY step of the summarization family. The most commonly used options are DROP, KEEP, and RENAME. For more information, see SAS Help and Documentation. If no data set options are needed, then leave this field blank.</p> <p>The WHERE data set option is coded separately in a different field and applies to both the initial and the re-summarizing steps.</p>
Summary_Outdsn_Where	<p>Is the WHERE data set option that filters the Results_Table data set. If you do not need to provide a filter, then leave this field blank.</p> <p>The SUMMARY engine creates a Where=(_type_ eq 'mask') statement for every Results_Table data set in order to make sure that only the summaries specified in Class_Var_List are written to Results_Table. If you need additional filtering, then define it in this field by using the following syntax: Where=(<i><subsetting condition></i>) The entire WHERE statement is not contained within parentheses.</p>
Output_Options	<p>Contains the options for the PROC SUMMARY output statement. The most commonly used option is SUM, which makes sure that the Analysis_Var_List entries that are summed have the same variable name in Results_Table as the source variables in Source_Table. For more information, see SAS Help and Documentation.</p>
Results_Table	<p>Is the prefix that all members in a summarization family share (for example, Hourly_Metrics). The SUMMARY engine combines Results_Table with a suffix derived from an entry in Summary_Level_List to create the name for each SAS summary data set.</p>

General Information About SAS Web Analytics Summaries

There is one metadata record in the Waadmsum data set for each summarization family.

Note: A summarization family, such as Hourly Metrics, consists of one member for each summary level: day, week, month, quarter, and year. △

It is easier to maintain one metadata record per family instead of one record per family member.

The record of a summarization family in the Waadmsum data set may not have information in all of the variables. Only the variables that are necessary to define the family summaries have values. Any of the variables that are not needed contain blanks.

Each Traffic report in the SAS Web Analytics Report Viewer displays the information from a SAS summary data set that corresponds to one family member. The following PROC SUMMARY steps are required to create each SAS summary data set:

- the initial step that summarizes the Source_Table information
- the re-summarization step that applies the results from the initial step to the information that already exists in the SAS summary data set

How the SUMMARY Engine Performs an Initial Summarization

In the following example, the SUMMARY engine uses the specific record from the Waadmsum data set to create the SAS summary data sets for the Hourly_Metrics summarization family.

Table A3.2 Fields in the Waadmsum Data Set for the Hourly_Metrics Summarization Family

Symbol	Variable in the Waadmsum Data Set	Value
	Label	Hourly Metrics
❶	Source_Table	detail
❷	Proc_Summary_Options	sum missing
	Input_Where	<i>blank</i>
	Summary_Level_List	day, week, month, qtr, year
❸❹	Class_Var_List	date, hour
	Id_Var_List	<i>blank</i>
	Id_Var_Summary_Level_List	<i>blank</i>
❺	Analysis_Var_List	entry_point, file_count, page_count, bytes_sent
❻	Output_Dsn_Options_0	<i>blank</i>
❼	Output_Dsn_Options_1	rename=(entry_point=session_count)
❽	Output_Dsn_Where_Stmt	<i>blank</i>
❾	Output_Options	sum=
❿	Results_Table	hourly_metrics

The SUMMARY engine builds the following PROC SUMMARY step for the initial summarization of the _Day member of the Hourly Metrics family from the metadata listed in the previous table:

```

❶ proc summary data=❷temp_lib..detail_20040115 sum missing chartype;
  ❸ class date
    hour
    ;
  ❹ types date*hour
    ;
  ❺ var entry_point
    file_count
    page_count
    bytes_sent

```

```

;
output out=&temp_lib..hourly_metrics_day_X(
  where=(_type_='11')
  rename=(entry_point=session_count)
  keep= date hour entry_point file_count
        page_count bytes_sent _type_)
sum=;
run;

```

The following table explains the features of the initial step of the PROC SUMMARY step:

Table A3.3 The Initial Step of the PROC SUMMARY Step

Symbol	Description of the Value
❶	The SUMMARY engine expands Source_Table so that it has both libname (&temp_lib) and memname (detail_20040115) components. The SUMMARY engine derives the yyymmdd suffix for the memname component from the dates in the SAS e-Data ETL source file (in this example, '20040115' indicates a date of January 15, 2004).
❷	The SUMMARY engine places the Proc_Summary_Options value (without any delimiting commas, and, if necessary, with chartype) into the PROC SUMMARY statement.
❸	The SUMMARY engine converts the comma-delimited entries in Class_Var_List to a variable list for the PROC SUMMARY CLASS statement.
❹	The SUMMARY engine uses the entries in Class_Var_List to create the PROC SUMMARY TYPES statement.
❺	The SUMMARY engine converts the comma-delimited entries in analysis_var_list to items in the variable list for the PROC SUMMARY VAR statement.
❻	The SUMMARY engine expands Results_Table so that it has both libname (&temp_lib) and memname (Hourly_Metrics_Day_X) components. The SUMMARY engine derives the _Day suffix for the memname component from an entry in Summary_Level_List. The '_X' suffix makes sure that the summarization will have a unique DSN even if several dates exist in the Web log that is parsed by the SAS e-Data ETL application.
❼	The SUMMARY engine places the Output_Dsn_Options_1 value verbatim into the OUTPUT statement. This PROC SUMMARY step is for an initial summarization of the SAS e-Data ETL data.
❼	The SUMMARY engine supplements the Output_Dsn_Options_1 value with additional output data set options that are necessary for the Output Out= data set to be created correctly.

Symbol	Description of the Value
⑧	The SUMMARY engine generates a WHERE statement to control what gets written to the Output Out= data set. If Summary_Outdsn_Where is not blank, then its filter is logically appended (with AND) to this generated WHERE statement.
⑨	The SUMMARY engine places the Output_Options value verbatim into the OUTPUT statement. The SUM= option indicates that the summed results in Results_Table are stored in variables that have the same name as the source variable. For example, the sum of all the File_Count variables is stored in a variable called File_Count.

How the SUMMARY Engine Performs a Re-Summarization

The following code is the PROC SUMMARY step that the SUMMARY engine builds for the re-summarization of the _Day members of the Hourly Metrics family. The code is built from the Waadmsum metadata listed above. The re-summarization step has components that are similar to the initial summarization step.

```

      ①
proc summary data=&temp_lib..hourly_metrics_day sum missing chartype;
  class date
    hour
  ;
  types date*hour
  ;
  ⑥ var session_count
    file_count
    page_count
    bytes_sent
  ;
      ⑥
  output out=summary.hourly_metrics_day
    where=( _type_='11' )
  ⑦ keep= date hour session_count file_count page_count bytes_sent _type_
    sum=;
run;

```

The following table discusses the differences between the re-summarization and the initial summarization steps:

Table A3.4 The Re-Summarization Step of the PROC SUMMARY Step

Symbol	Description of the Value
①	The SUMMARY engine uses the Output Out= data set from the initial summarization to identify the Input data set for the re-summarization.
⑤	The SUMMARY engine changes the VAR statement from the initial summarization to reflect the variable renaming that was performed during that re-summarization step. Because the Entry_Point field was renamed to Session_Count, the Entry_Point field needs to be referred to as Session_Count in the re-summarization step.
⑥	The SUMMARY engine expands Results_Table so that it has both libname (Summary) and memname (Hourly_Metrics_Day) components. The Summary libref makes sure that the results are written to the permanent library of the Web mart. The SUMMARY engine derives the _Day suffix for the memname component from an entry in Summary_Level_List and identifies the Output Out= data set as the SAS summary data set for the _Day member of the Hourly Metrics family.
⑦	The SUMMARY engine supplements the Output_Dsn_Options_0 value with additional output data set options that are necessary for the Output Out= data set to be created correctly. Because the Output_Dsn_Options_0 of this family is blank, the only text is the SUMMARY engine supplement. If the Output Out= data set for the re-summarization requires any options that the SUMMARY engine cannot derive exclusively, then the required options must be supplied by the text in Output_Dsn_Options_0.

Modifying the Waadmsum Metadata Data Set

To customize the default SAS Web Analytics summary data sets, you will need to change the Waadmsum metadata that control the summarizations.

CAUTION:

Make a backup copy of the Config.Waadmsum data set before you make any changes. △

The SUMMARY engine performs validity checks on the Waadmsum metadata before it generates any PROC SUMMARY steps. If the changes that you make corrupt the integrity of the Waadmsum metadata, then both the SUMMARY engine and the %WAETL macro will fail to run. If you back up your Waadmsum metadata, then you can restore the Waadmsum data set and continue to run the %WAETL macro while you research the problem with your changes to the Waadmsum metadata. If you do not have a backup, then you will not be able to run the %WAETL macro until you have resolved the problem.

To modify or delete a summarization family, see “Modifying or Deleting a Summarization” on page 127.

Overview: The Waconfig Data Set

The purpose of the Waconfig data set is to list parameters that you can manipulate in order to control the %WAETL macro. To control the statistical processing that is performed by the macro, it is easier to change the metadata entry in the Waconfig data set than it is to change the statements within the %WAETL macro itself.

The Waconfig data set contains the following columns:

Parameter	lists the names of parameters that correspond to macro variables.
Value	contains the values for the parameters.
Description	identifies the specifications in the %WAETL macro to which the values in the Value column correspond.

The %WAETL macro converts each record in the Waconfig data set to a macro variable.

Example: Using a Parameter in the Waconfig Data Set to Modify the %WAETL Macro

In the Waconfig data set, the WEB_DAYS_IN_HISTORY parameter is initially set up to retain 90 days of summaries. You can change the number of days that are retained by modifying the value for this parameter. For example, you can set the value for WEB_DAYS_IN_HISTORY to 60 in the Waconfig data set. Because the %WAETL macro reads the Waconfig data set as the source for its variables, the %WAETL macro then uses 60 as the number of days of summaries to keep.

The Parameters in the Waconfig Data Set

The following table lists the parameters of the Waconfig data set by category:

Table A3.5 Parameters in the Waconfig Data Set

Category	Description of Category	Parameter
Pathing	Specifies the parameters that are used to create the Summary.Pathing data set.	WAB_R1_SUPPORT
		WAB_R7_SUPPORT
		WAB_R30_SUPPORT
		WAB_ITEMSET_SIZE
Data Aging Thresholds	Specifies the parameters that control how often the historical data is aged and therefore how often it is routinely removed from the Web mart.	WAB_DAYS_IN_HISTORY
		WAB_WEEKS_IN_HISTORY
		WAB_MONTHS_IN_HISTORY
		WAB_QUARTERS_IN_HISTORY
		WAB_YEARS_IN_HISTORY
		WAB_NUM_DETAIL_DATA_SETS
		WAB_NUM_SESSION_DATA_SETS
		WAB_VISITOR_DAYS_IN_HISTORY
		WAB_PATHING_DAYS_IN_HISTORY

Category	Description of Category	Parameter
Decision Support Reports	Specifies the parameters that control the Decision Support report data sets.	WAB_AUTOSEG_TRAIN_SAMP_PCT WAB_AUTOSEG_VALID_SAMP_PCT WAB_AUTOSEG_NUM_SEGMENTS WAB_AUTOSEG_ALPHA WAB_AUTOSEG_MINWORTH WAB_AUTOSEG_MAXDEPTH WAB_AUTOSEG_LEAFSIZE WAB_DAYS_IN_SCORECARD WAB_DAYS_IN_DASHBOARD
Session Summary Control	Specifies how session-related summaries are assigned.	WAB_INIT_BY_SESS_VARS WAB_FIRST_SESS_VARS WAB_LAST_SESS_VARS WAB_SUM_SESS_VARS
Summary Engine	Specifies which data set is the metadata source for the %WASUMENG macro.	WAB_METADSN

Listed below in alphabetical order are all the default values for the parameters in the Waconfig data set as it is shipped with SAS Web Analytics. For the parameters with blanks in the Default Value column, the default value is null.

Table A3.6 Parameters in the Waconfig Data Set

Parameter	Default Value	Description
WAB_AUTOSEG_ALPHA	0.20	For automatic segmentation, a threshold p-value for the significance level of a candidate-splitting rule. This value applies only for targets that have an interval measurement level.
WAB_AUTOSEG_LEAFSIZE		For automatic segmentation, the smallest number of training observations that a new branch can have.
WAB_AUTOSEG_MAXDEPTH	10	For automatic segmentation, the number of splitting rules that are needed to define the node. The root node has a depth of 0. The children of the root node have a depth of 1, and so on.
WAB_AUTOSEG_MINWORTH	0.0	For automatic segmentation, a threshold value for the logworth of a candidate-splitting rule. This value applies only for targets that have a nominal or an ordinal measurement level.

Parameter	Default Value	Description
WAB_AUTOSEG_NUM_SEGMENTS	4	The number of segments that are generated by the autosegmentation model.
WAB_AUTOSEG_TRAIN_SAMP_PCT	0.6	The percentage (expressed as a decimal number) that determines whether an observation gets assigned to the Training data set or to the Validation data set.
WAB_AUTOSEG_VALID_SAMP_PCT	0.4	The proportion of total visitor IDs that are used to identify or to validate the automatic segmentation model.
WAB_DAYS_IN_DASHBOARD	365	The number of days that are available for dashboard reports.
WAB_DAYS_IN_HISTORY	90	The number of days that are available for daily summary reports.
WAB_DAYS_IN_SCORECARD	365	The number of days that are available for scorecard reports.
WAB_FIRST_SESS_VARS		A comma-delimited list of session variables that have values for the First.Session_Id variable. There is only one variable, Status_Code, listed by default.
WAB_INIT_BY_SESS_VARS	Status_Code	A comma-delimited list of session variables that have values for the Initialize_By_Session variable.
WAB_ITEMSET_SIZE	7	The maximum number of URLs per page that are allowed in a path.
WAB_LAST_SESS_VARS		A comma-delimited list of session variables.
WAB_METADSN	config.wasumeng	The summary metadata set that causes creation of the PROC SUMMARY code. This code is used to build the descriptive summary in the data warehouse.
WAB_MONTHS_IN_HISTORY	36	The number of months that are available in the monthly summary tables.
WAB_NUM_DETAIL_DATA_SETS	7	The number of separate date-partitioned detail data sets that are kept in the Detail library.
WAB_NUM_SESSION_DATA_SETS	14	The number of separate date-partitioned detail data sets that are kept in the Dated library.
WAB_PATHING_DAYS_IN_HISTORY	30	The number of days of history that are kept in the Summary.Pathing data set.

Parameter	Default Value	Description
WAB_QUARTERS_IN_HISTORY	12	The number of quarter years that are available in the quarterly summary tables.
WAB_R1_SUPPORT	10	The minimum number of total sessions that correspond to a unique path over an interval of one day.
WAB_R30_SUPPORT	100	The minimum number of total sessions that correspond to a unique path over an interval of 30 days.
WAB_R7_SUPPORT	40	The minimum number of total sessions that correspond to a unique path over an interval of 7 days.
WAB_SUM_SESS_VARS		A comma-delimited list of session variables with the sum of all Session_Id values.
WAB_VISITOR_DAYS_IN_HISTORY	60	The number of days of history that are kept in the Summary.Visitor_Day data set.
WAB_WEEKS_IN_HISTORY	52	The number of weeks that are available in the weekly summary tables.
WAB_YEARS_IN_HISTORY	3	The number of years that are available in the yearly summary tables.

Changing Values in the Waconfig Data Set

If the default values for the parameters in the Waconfig data set do not meet your needs, then you can change the values for these parameters by using the SAS Web Analytics Administrator. The following procedure shows you how to change any values in the Waconfig data set.

CAUTION:

Make a backup copy of the Config.Waconfig data set before you make any changes to it.

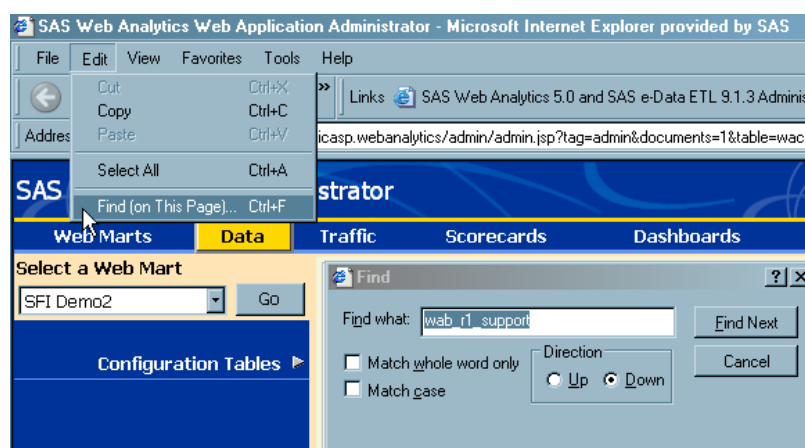
If the changes that you made do not work properly, then you will be able to restore your original settings by using the backup copy. △


How to Edit the Waconfig Data Set

To edit the Waconfig data set, open the SAS Web Analytics Administrator interface, and follow these steps:

- 1 If a Web mart is not already selected, select a Web mart from the drop-down menu in the upper left corner of the SAS Web Analytics Administrator, and click **Go**.
- 2 Select the **Data** link from the menu bar so that **Data** is highlighted in yellow.
- 3 Move the mouse over **Configuration Tables** (located in the navigation area on the left of the Web page), and then select **Web Analytics Configuration** in the expanded navigation tree. The Waconfig data set is displayed.
- 4 Find the row whose description matches the summarization that you want to change. To use the browser's Find function, select **Edit**, and then **Find (on This Page)**. In the **Find** dialog box, type the name of the parameter in order to quickly locate the row you want to change in the Waconfig data set.

Display A3.1 Using the Browser's Find Function to Locate a Row in Waconfig



- 5 Click  in the **ID for Maintenance** column of the row you want to change. The Metadata Administration Form (shown in the following display) enables you to edit any values for that row.

Display A3.2 The Metadata Administration Form

The screenshot shows the 'Analytics Administrator' interface with a top navigation bar containing 'Data', 'Traffic', 'Scorecards', 'Dashboards', 'Segmentations', 'Refresh', 'Help', and 'SAS'. On the left, there is a sidebar with 'Mart' and 'Configuration Tables'. The main area is titled 'Metadata Administration Form' and contains three input fields: 'Value' with '10', 'Description' with 'The minimum count of total sessions for path for 1 day.', and 'Parameter Name' with 'wab_r1_support'. At the bottom are 'Submit' and 'Cancel' buttons.

- 6 Type the new value in the **Value** text box. Click **Submit**, and then click **Continue**.
- 7 To find and change any additional summarization parameters in the Waconfig data set, repeat steps 4, 5 and 6 above.

Changing the Parameters in the Session Summary Control Category

The parameters in the Session Summary Control category enable you to customize the way that the %WAETL macro creates the analysis variables. The parameter WAB_INIT_BY_SESS_VARS affects the way in which the %WAETL macro creates the daily detail data sets. The following parameters affect the way in which the %WAETL macro creates the daily session summary data sets:

- ☐ WAB_FIRST_SESS_VARS
- ☐ WAB_LAST_SESS_VARS
- ☐ WAB_SUM_SESS_VARS

The WAB_INIT_BY_SESS_VARS Parameter for Controlling Daily Detail Data Sets

The initial value for the WAB_INIT_BY_SESS_VARS parameter is **status_code**. Setting **status_code** as the initial value for the WAB_INIT_BY_SESS_VARS parameter means that when the %WAETL macro creates the daily detail data sets, the first time that a status code value for a session is encountered, an indicator variable (Init_By_Sess_A) is set to **1**. For all of the subsequent times that this particular status code is encountered for that session, the variable Init_By_Sess_A is set to **0**. In this manner, the indicator variable Init_By_Sess_A enables you to count how many sessions have experienced each status code value.

The following table illustrates how the value for the variable `Init_By_Sess_A` changes when a sequence of values for each specific status code is encountered:

Table A3.7 How the Indicator Variable `Init_By_Sess_A` Is Set When a Status Code Is Encountered

Sequence in a Session	Status_Code	Init_By_Sess_A
1	200	1
2	200	0
3	401	1
4	302	1
5	200	0
6	500	1
7	401	0
8	302	0

If you want to supplement the SAS Web Analytics reports with a similar tally by session, you need to add the appropriate session variable name to the Value field of the `WAB_INIT_BY_SESS_VARS` parameter in the Waconfig data set.

Note: The Value field requires a comma-delimited variable list; therefore, you must insert a comma before each variable name that you add to the list. \triangle

The `%WAETL` macro automatically creates an `Init_By_Sess_X` indicator variable to represent each variable in the list of values for the `WAB_INIT_BY_SESS_VARS` parameter, where *X* is a letter suffix that identifies the position of the source variable in the list. For example, `Init_By_Sess_A` is the indicator variable for `status_code`, the first variable in the list; `Init_By_Sess_B` is for the second variable in the list; `Init_By_Sess_C` is for the third variable; and so on.

If you add a variable to the Value field for the `WAB_INIT_BY_SESS_VARS` parameter, then you must make changes to the metadata in the Wasumeng summary table so that the SUMMARY engine will include the new variable in the summarizations that are performed by the `%WAETL` macro. For more information about `%WAETL`, see Appendix 1, “Macro Reference for the SAS Web Analytics 5.1 Solution,” on page 179. Because the `Init_By_Sess_A` variable has already been set up as the indicator variable for the parameter value `status_code`, you might want to model the updates that you make to the Wasumeng summary table by using an existing summary that includes the variable `Init_By_Sess_A` in its `ANALYSIS_VARS` parameter.

The Parameters for Customizing Session Summary Data Sets

Each session data set has one observation per Session_ID field and is created when the %WAETL macro summarizes the Weblog_Detail detail data set for a selected date by the Session_ID field. You can customize the variables in the session data sets by using any of the following methods (singly or in combination) :

- Calculate the sum of the values of the WAB_SUM_SESS_VARS parameter for the Session_ID field.
- Use the first (entry) value of the WAB_FIRST_SESS_VARS parameter for the Session_ID field.
- Use the last (exit) value of the WAB_LAST_SESS_VARS parameter for the Session_ID field.

The value of each of these parameters is a comma-delimited list of variable names. To customize a variable by using one of these three methods, add a variable name to the list in the Value column of the corresponding parameter.

Although the SAS Web Analytics software does not use this feature by default, the following example shows how this customization would work by using these variables in a hypothetical Weblog_Detail detail data set:

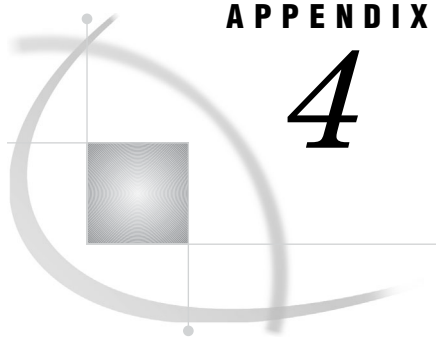
```
Wab_sum_sess_vars: var_1, var_2, var_3
Wab_first_sess_vars: var_a, var_b
Wab_last_sess_vars: var_z
```

Detail Source Data Set

Session_ID	Sequence	Var_a	Var_b	Var_z	Var_1	Var_2	Var_3
0001	1	A1	B1	Z1	11	21	31
0001	2	A2	B2	Z2	12	22	32
0001	3	A3	B3	Z3	13	23	33
0001	4	A4	B4	Z4	14	24	34

Session Data Set

Session_ID	Var_a	Var_b	Var_z	Var_1	Var_2	Var_3
0001	A1	B1	Z4	50	90	130



APPENDIX

4

The Summary Data Sets for SAS Web Analytics 5.1

<i>Autoseg_Dectree Data Set</i>	210
<i>Autoseg_Nodestat Data Set</i>	211
<i>Autoseg_Path Data Set</i>	212
<i>Autoseg_Segment_Rules Data Set</i>	213
<i>Browser_<Summary Level> Data Set</i>	214
<i>Browser_Version_<Summary Level> Data Set</i>	215
<i>Daily_Total_<Summary Level> Data Set</i>	216
<i>Dashboard_Data Data Set</i>	217
<i>Scorecard_Data Data Set</i>	218
<i>Scorecard_Metric_History Data Set</i>	220
<i>Dashboard_Metric_History Data Set</i>	221
<i>Search_Term_<Summary Level> Data Set</i>	222
<i>Hourly_Metrics_<Summary Level> Data Set</i>	223
<i>Hourly_Status_<Summary Level> Data Set</i>	224
<i>Page_Dates Data Set</i>	225
<i>Page_<Summary Level> Data Set</i>	225
<i>Page_Status_<Summary Level> Data Set</i>	227
<i>Pathing Data Set</i>	228
<i>Pathing_Dmdb Data Set</i>	229
<i>Pathing_<Summary Level> Data Set</i>	230
<i>Platform_<Summary Level> Data Set</i>	231
<i>Referrer_Domain_<Summary Level> Data Set</i>	232
<i>Referrer_Entry_Point_<Summary Level> Data Set</i>	233
<i>Referrer_Page_Status_<Summary Level> Data Set</i>	234
<i>Referrer_Search_Term_<Summary Level> Data Set</i>	235
<i>Status_<Summary Level> Data Set</i>	236
<i>Top_Entry_Paths_<Summary Level> Data Set</i>	237
<i>Top_Referrer_Paths_<Summary Level> Data Set</i>	238
<i>Visitor_<Summary Level> Data Set</i>	239

Autoseg_Dectree Data Set

Library: Summary

Data set name: Autoseg_Dectree

Summary levels: Not applicable for this data set

Description: This data set is created using the Nodestat output data set that is created in PROC ARBORETUM. It contains node information for all defined segmentation reports.

Default reports using this data (Group-Name; Report-Name): Segmentation; Repeat Visitors

Processes that use this data set as input: None

Created by: The segmentation function of either the %WADECIDE macro or the %WAAUTOSG macro

Display 18.1 Variable Descriptions

NAME	TYPE	LENGTH	DESCRIPTION	ROLE
DATE	NUM	8	analysis date	CLASS
SEGMENTATION_NAME	CHAR	40	segment name is assigned using the SAS Web Analytics Administrator	CLASS
VAR1	NUM	8	node	ANALYSIS
VAR2	NUM	8	parent	ANALYSIS
VAR3	NUM	8	link width	ANALYSIS
VAR4	CHAR	1000	training and validation target totals	CLASS
VAR5	CHAR	32	text above the node	CLASS
VAR6	CHAR	32	text below the node	CLASS
VAR7	NUM	8	Indicates whether the decision tree node is missing	ANALYSIS


Autoseg_Nodestat Data Set

Library: Summary

Data set name: Autoseg_Nodestat

Summary levels: Not applicable for this data set

Description: This data set is created from various data sets created by PROC ARBORETUM and contains the probability information for each node and leaf combination. The number of variables within the data set increases each time a new segment is defined. The variables Segmentation_Name, Leaf, and Node will be populated for all segmentation reports.

Note: The variable Leaf might be missing in some instances. 

Default reports using this data (Group-Name; Report-Name): Segmentation; Repeat Visitors

Processes that use this data set as input: None

Created by: The segmentation function of either the %WADECIDE macro or the %WAAUTOSG macro

Display 18.2 Variable Descriptions

NAME	TYPE	LENGTH	DESCRIPTION	ROLE
LEAF	NUM	8	Leaf	ANALYSIS
NODE	NUM	8	Node	ANALYSIS
P_REPEAT_VISITORNO	NUM	8	Predicted: repeat_visitor=No, specific for the repeat visitor segmentation report	ANALYSIS
P_REPEAT_VISITORYES	NUM	8	Predicted: repeat_visitor=Yes, specific for the repeat visitor segmentation report	ANALYSIS
SEGMENTATION_NAME	CHAR	40	segment name assign through the WAA Administrator	CLASS

Autoseg_Path Data Set

Library: Summary

Data set name: Autoseg_Path

Summary levels: Not applicable for this data set

Description: This data set is created using the PROC ARBORETUM output data set. It is used to build the segmentation rules for each segment within a segmentation report.

Default reports using this data (Group-Name; Report-Name): Segmentation; Repeat Visitors

Processes that use this data set as input: None

Created by: The segmentation function of either the %WADECIDE macro or the %WAAUTOSG macro

Display 18.3 Variable Descriptions

NAME	TYPE	LENGTH	DESCRIPTION	ROLE
CHARACTER_VALUE	CHAR	19	CHARACTER VALUE	CLASS
LEAF	NUM	8	LEAF	ANALYSIS
NODE	NUM	8	NODE	ANALYSIS
NUMERIC_VALUE	NUM	8	NUMERIC VALUE	ANALYSIS
RELATION	CHAR	11	RELATION	CLASS
SEGMENTATION_NAME	CHAR	40	segment name assign through the WA Administrator	CLASS
VAR_NAME	CHAR	32	VAR NAME	CLASS
VARIABLE	CHAR	256	VARIABLE	CLASS

Autoseg_Segment_Rules Data Set

Library: Summary

Data set name: Autoseg_Segment_Rules

Summary levels: Not applicable for this data set

Description: The purpose of the Autoseg_Segment_Rules data set is to create the actual segmentation report within the report viewer. This data set is created using the PROC ARBORETUM output data sets. For each segmentation report, Segmentation_Name, Segmentation_Rule_Name, and Segmentation_Rule_Label are always populated. The variables that begin with P_ are specific to segmentation reports and only will be populated for the corresponding segmentation report.

Default reports using this data (Group-Name; Report-Name): Segmentation; Repeat Visitors

Processes that use this data set as input: None

Created by: The segmentation function of either the %WADECIDE macro or the %WAAUTOSG macro

Display 18.4 Variable Descriptions

NAME	TYPE	LENGTH	DESCRIPTION	ROLE
P_REPEAT_VISITORNO	NUM	8	Predicted: repeat_visitor=No	ANALYSIS
P_REPEAT_VISITORYES	NUM	8	Predicted: repeat_visitor=Yes	ANALYSIS
DATE	NUM	8	analysis date	CLASS
SEGMENT_RULE	CHAR	1024	segment rule using variable labels - used in report	CLASS
SEGMENT_RULE_LABEL	CHAR	1024	segment rule using variable names	CLASS
SEGMENTATION_NAME	CHAR	40	segment name is assigned through the VWA Administrator	CLASS

Browser_<Summary Level> Data Set

Library: Summary

Data set name: Browser_<Summary Level>

Summary levels: Each analysis variable is summed for the summary level of the corresponding data set:

_Day
_Week
_Month
_Qtr
_Year

Description: This data set contains distribution data on the different Web browsers used by visitors navigating the Web site.

Default reports using this data (Group-Name; Report-Name): Platform; Browsers

Processes that use this data set as input: None

Display 18.5 Variable Descriptions

NAME	TYPE	LENGTH	DESCRIPTION	ROLE
BROWSER	CHAR	40	The browser that is used to access the Web site. A browser is a program that accesses and displays files and other data that are available on the Internet and other networks	CLASS
DATE	NUM	8	Date of the first day in the summary level	CLASS
PAGE_COUNT	NUM	8	The number of page views per browser type. A page view occurs when a page is of a valid application type and is loaded from a server to a browser when the status code for the page is between 200 and 299, or is 304	ANALYSIS
SESSION_COUNT	NUM	8	The number of sessions that are logged per browser type	ANALYSIS

Browser_Version_<Summary Level> Data Set

Library: Summary

Data set name: Browser_Version_<Summary Level>

Summary levels: Each analysis variable is summed for the summary level of the corresponding data set:

_Day
_Week
_Month
_Qtr
_Year

Description: This data set contains distribution data on the different Web browser versions that are used by visitors navigating the Web site.

Default reports using this data (Group-Name; Report-Name): Platform; Browser Version

Processes that use this data set as input: None

Display 18.6 Variable Descriptions

NAME	TYPE	LENGTH	DESCRIPTION	ROLE
BROWSER	CHAR	40	The browser that is used to access the Web site. A browser is a program that accesses and displays files and other data that are available on the Internet and other networks.	CLASS
BROWSER_VERSION	CHAR	8	The version identifier for the browser.	CLASS
DATE	NUM	8	Date of the first day in the summary level	CLASS
PAGE_COUNT	NUM	8	The number of page views per browser version. A page view occurs when a page is of a valid application type and is loaded from a server to a browser when the status code for the page is between 200 and 299, or is 304.	ANALYSIS
SESSION_COUNT	NUM	8	The number of sessions that are logged per browser version.	ANALYSIS

Daily_Total_<Summary Level> Data Set

Library: Summary

Data set name: Daily_Total_<Summary Level>

Summary levels: Each analysis variable is summed for the summary level of the corresponding data set:

_Day
_Week
_Month
_Qtr
_Year

Description: This data set contains Web site traffic metric values that are summarized for the corresponding summary level.

Default reports using this data (Group-Name; Report-Name):

Status Codes; Status Codes
Overview; Site Metrics
Overview; Weekday Metrics

Processes that use this data set as input: Scorecard, dashboard, and segmentation

Display 18.7 Variable Descriptions

NAME	TYPE	LENGTH	DESCRIPTION	ROLE
DATE	NUM	8	Date of the first day in the summary level	CLASS
DOW_ID	CHAR	2	Day of Week identifier	CLASS
DURATION	NUM	8	Total time (in seconds) for all sessions	ANALYSIS
ERR_302_COUNT	NUM	8	302 Status: Found	ANALYSIS
ERR_304_COUNT	NUM	8	304 Status: Not Modified	ANALYSIS
ERR_400_COUNT	NUM	8	400 Status: Bad Request	ANALYSIS
ERR_401_COUNT	NUM	8	401 Status: Unauthorized	ANALYSIS
ERR_403_COUNT	NUM	8	403 Status: Forbidden	ANALYSIS
ERR_404_COUNT	NUM	8	404 Status: Not Found	ANALYSIS
ERR_405_COUNT	NUM	8	405 Status: Method Not Allowed	ANALYSIS
ERR_408_COUNT	NUM	8	408 Status: Request Time-out	ANALYSIS
ERR_500_COUNT	NUM	8	500 Status: Internal Server Error	ANALYSIS
ERR_501_COUNT	NUM	8	501 Status: Not Implemented	ANALYSIS
FILE_COUNT	NUM	8	The number of total file downloads	ANALYSIS
ONE_HIT_SESSION_COUNT	NUM	8	The number of sessions with one page hit	ANALYSIS
PAGE_COUNT	NUM	8	Page Count	ANALYSIS
PAGE_VIEW_2TO4_IND	NUM	8	The number of sessions with two to four page hits	ANALYSIS
PAGE_VIEW_GE5_IND	NUM	8	The number of sessions with greater than 5 page hits	ANALYSIS
PAGE_VIEW_LE1_IND	NUM	8	The number of sessions with zero or one page hits	ANALYSIS
SESSION_COUNT	NUM	8	The total number of sessions	ANALYSIS
TOTAL_BYTES_SENT	NUM	8	The total number of bytes sent	ANALYSIS

Each analysis variable is summed for the corresponding summary level of the data set.

Dashboard_Data Data Set

Library: Summary

Data set name: Dashboard_Data

Summary levels: This data set is a repository for the dashboard report data. All defined dashboards are generated each day using the Daily_Total_Day data set. Therefore, the values that are represented in the Dashboard data set are values with the _Day summary level.

Description: This data set contains the values resulting from the dashboard process. The values for each day's dashboard are appended to this data set. The combination of a Report_Date value and the Dashboard name is used to look up a particular dashboard's data for a particular day. Values for this data set are not generated until after 30 days of Web log processing. See also: Dashboard_Metric_History

Default reports using this data (Group-Name; Report-Name): Dashboard; Dashboard

Processes that use this data set as input: None

Display 18.8 Variable Descriptions

NAME	TYPE	LENGTH	DESCRIPTION	ROLE
ACTUAL	NUM	8	The observed metric value for the day	ANALYSIS
BUSINESS_DIR	CHAR	5	Positive business direction (up or down)	ANALYSIS
CATEGORY	CHAR	45	Label used to group metrics	ANALYSIS
DASHBOARD	CHAR	25	Identifying name for the dashboard	ANALYSIS
INTERCEPT	NUM	8	Intercept	ANALYSIS
LABEL	CHAR	64	The name of the metric	ANALYSIS
MAX	NUM	8	30 Day Maximum	ANALYSIS
MEAN	NUM	8	30 Day Average	ANALYSIS
MIN	NUM	8	30 Day Minimum	ANALYSIS
PERFORMANCE	CHAR	15	indicates how the metric has performed with respect to its Positive Business Direction*	ANALYSIS
PVALUE	NUM	8	Performance significance	ANALYSIS
REPORT_DATE	NUM	8	Date for which the dashboard was generated	ANALYSIS
SLOPE	NUM	8	The value of the slope of the 7 day trend line	ANALYSIS
SLOPE_STDERR	NUM	8	The standard error for the slope of the 7 day trend line	ANALYSIS
STD	NUM	8	Standard deviation for the metric's value	ANALYSIS
STD_TREND	NUM	8	Standardized trend line slope	ANALYSIS
VAR_NAME	CHAR	64	The name of the metric	ANALYSIS

*The performance variable will have the value 1 through 5

1 means that the metric's trend is up, and the trend matches the positive business direction.

2 means that the metric's trend is up, and the trend does not match the positive business direction.

3 means that the metric's trend is level (neutral).

4 means that the metric's trend is down, and the trend matches the positive business direction.

5 means that the metric's trend is down, and the trend does not match the positive business direction.

Scorecard_Data Data Set

Library: Summary

Data set name: Scorecard_Data

Summary levels: This data set is a repository for the scorecard report data. All defined scorecards are generated each day by using the Daily_Total_Day data set. So the values represented in the scorecard data set are values with the _Day summary level.

Description: This data set contains the values that result from the scorecard process. The values for each day's scorecard are appended to this data set. The combination of a Report_Date value and the Scorecard name is used to look up a scorecard's data for a day. Values for this data set are not generated until after 30 days of Web log processing. See also: Scorecard_Metric_History

Default reports using this data (Group-Name; Report-Name): Scorecard; Scorecard

Processes that use this data set as input: None

Display 18.9 Variable Descriptions

NAME	TYPE	LENGTH	DESCRIPTION	ROLE
ACTUAL	NUM	8	Actual Value for the metric	ANALYSIS
DATE	NUM	8	The date of the metric data	ANALYSIS
ERROR	NUM	8	Prediction Errors	ANALYSIS
LOWER	NUM	8	Lower 95% confidence limit	ANALYSIS
PREDICT	NUM	8	The value that was predicted for the metric	ANALYSIS
STD	NUM	8	Prediction standard errors	ANALYSIS
UPPER	NUM	8	Upper 95% confidence limit	ANALYSIS
_0	NUM	8	0% Increase*	ANALYSIS
_5	NUM	8	5% Increase*	ANALYSIS
_10	NUM	8	10% Increase*	ANALYSIS
_15	NUM	8	15% Increase*	ANALYSIS
_20	NUM	8	20% Increase*	ANALYSIS
_25	NUM	8	25% Increase*	ANALYSIS
_30	NUM	8	30% Increase*	ANALYSIS
_35	NUM	8	35% Increase*	ANALYSIS
_40	NUM	8	40% Increase*	ANALYSIS
_45	NUM	8	45% Increase*	ANALYSIS
_50	NUM	8	50% Increase*	ANALYSIS
LABEL	CHAR	40	Label of the metric	ANALYSIS
WEIGHT	NUM	8	Represents 1-p-value for the relation of the metric with the target.	ANALYSIS
NAME	CHAR	45	Name of the metric	ANALYSIS
BUSINESS_DIR	CHAR	8	Desired or positive trend direction for the metric. For example the desired business direction for the trend in error counts would be DOWN while the desired direction for number of sessions would be UP	ANALYSIS
BUSINESS_GOAL	NUM	8		ANALYSIS
DIFFERENCE	NUM	8	Relative Difference calculated as difference= error/std	ANALYSIS
IMPORTANCE	NUM	8	Rank for the metric which represents where it ranks in order with the other input metrics as far as its ability to affect the target metric	ANALYSIS
MEASUREMENT_RANGE	CHAR	40	Describes the types of numbers the metric will be expressed in. Possible values are: NEGATIVE NUMBER, POSITIVE NUMBER, PERCENTAGE, NEGATIVE PERCENTAGE, POSITIVE PERCENTAGE, PROPORTION, NEGATIVE PROPORTION, OR POSITIVE PROPORTION	ANALYSIS
PERFORMANCE	NUM	8	Number from 1 to 5 which maps to a icon used in the report viewer to represent how the metric is performing relative to its desired business direction	ANALYSIS
REPORT_DATE	NUM	8	Date on which the report was run	ANALYSIS
SCORECARD	CHAR	15	Name identifier for the used to map rows in the data set to a specific scorecard definition	ANALYSIS

*This value is used in the goal-seeking table. This number represents the value that the input metric must reach in order to cause the corresponding percentage change in the target metric.

Scorecard_Metric_History Data Set

Library: Summary

Data set name: Scorecard_Metric_History

Summary levels: This data set is a repository for the scorecard report data. All defined scorecards are generated each day using the Daily_Total_Day data set. So the values represented in the scorecard data set are values with the _Day summary level.

Description: This data set contains the historical values for each metric in each defined scorecard. The values for each day's scorecard are appended to this data set. The combination of a Report_Date value and the Scorecard name is used to look up a particular scorecard's historical data. Values for this data set are not generated until after 30 days of Web log processing. See also: Scorecard_Data

Default reports using this data (Group-Name; Report-Name): Scorecard; Scorecard

Processes that use this data set as input: None

Display 18.10 Variable Descriptions

NAME	TYPE	LENGTH	DESCRIPTION	ROLE
ACTUAL	NUM	8	Actual Values	ANALYSIS
DATE	NUM	8	Historical date for the metric value	ANALYSIS
ERROR	NUM	8	Prediction Errors	ANALYSIS
LOWER	NUM	8	Lower Confidence Limits	ANALYSIS
PREDICT	NUM	8	Predicted Values	ANALYSIS
STD	NUM	8	Prediction Standard Errors	ANALYSIS
UPPER	NUM	8	Upper Confidence Limits	ANALYSIS
LABEL	CHAR	40	Label of the Metric variable	ANALYSIS
PARM	CHAR	32	Parameter Name	ANALYSIS
PVALUE	NUM	8	P-Values of Parameter Estimate	ANALYSIS
TVALUE	NUM	8	T-Values of Parameter Estimate	ANALYSIS
NAME	CHAR	45	Name of the metric variable	ANALYSIS
REPORT_DATE	NUM	8	Date the report was run	CLASS
SCORECARD	CHAR	25	Identifier for the scorecard	ANALYSIS

Dashboard_Metric_History Data Set

Library: Summary

Data set name: Dashboard_Metric_History

Summary levels: This data set is a repository for the dashboard report data. All defined dashboards are generated each day using the Daily_Total_Day data set. So the values represented in the dashboard data set are values with the _Day summary level.

Description: This data set contains the historical values for each metric in each defined dashboard. The values for each day's dashboard are appended to this data set. The combination of a Report_Date value and the Dashboard name is used to look up a particular dashboard's historical data. Values for this data are not generated until after 30 days of Web log processing. See also: Dashboard_Data

Default reports using this data (Group-Name; Report-Name): Dashboard; Dashboard

Processes that use this data set as input: None

Display 18.11 Variable Descriptions

NAME	TYPE	LENGTH	DESCRIPTION	ROLE
ACTUAL	NUM	8	Actual	ANALYSIS
CATEGORY	CHAR	45	Used to group metrics together	ANALYSIS
DASHBOARD	CHAR	25	Dashboard identifier	ANALYSIS
DATE	NUM	8	Historical date for the value	CLASS
LABEL	CHAR	64	Metric variable label	ANALYSIS
REPORT_DATE	NUM	8	Date on which report was run	ANALYSIS
TREND_LINE	NUM	8	The value on the trend line for this day	ANALYSIS
VAR_NAME	CHAR	64	Metric variable name	ANALYSIS

Search_Term_<Summary Level> Data Set

Library: Summary

Data set name: Search_Term_<Summary Level>

Summary levels: Each analysis variable is summed for the summary level of the corresponding data set:

_Day
_Week
_Month
_Qtr
_Year

Description: This data set contains the number of sessions that resulted in the use of a search term for the corresponding summary level.

Default reports using this data (Group-Name; Report-Name): Referrer; Search Terms

Processes that use this data set as input: None

Display 18.12 Variable Descriptions

NAME	TYPE	LENGTH	DESCRIPTION	ROLE
DATE	NUM	8	Date of the first day in the summary level	CLASS
SEARCH_TERM	CHAR	1024	A search or query that is entered by a visitor at a referring site, which resulted to a link that sent visits to the Web site.	ANALYSIS
SESSION_COUNT	NUM	8	Total number of sessions that used the search term to reach the site	ANALYSIS

Hourly_Metrics_<Summary Level> Data Set

Library: Summary

Data set name: Hourly_Metrics_<Summary Level>

Summary levels: Each analysis variable is summed for the summary level of the corresponding data set:

_Day
_Week
_Month
_Qtr
_Year

Description: This data set contains Web traffic metrics per hour for a Web site.

Default reports using this data (Group-Name; Report-Name): Overview; Hourly Metrics

Processes that use this data set as input: None

Display 18.13 Variable Descriptions

NAME	TYPE	LENGTH	DESCRIPTION	ROLE
BYTES_SENT	NUM	8	The number of bytes downloaded per hour	ANALYSIS
DATE	NUM	8	Date of the first day in the summary level	CLASS
FILE_COUNT	NUM	8	The number of files that were viewed during a given hour.	ANALYSIS
HOUR	NUM	8	The hour in which the data was collected, based on a 24-hour clock (hour 0 is midnight-12:59 a.m., etc.)	CLASS
PAGE_COUNT	NUM	8	The number of pages that were viewed during a given hour.	ANALYSIS
SESSION_COUNT	NUM	8	The number of sessions that were logged per hour.	ANALYSIS

Hourly_Status_<Summary Level> Data Set

Library: Summary

Data set name: Hourly_Status_<Summary Level>

Summary levels: Each analysis variable is summed for the summary level of the corresponding data set:

_Day
_Week
_Month
_Qtr
_Year

Description: This data set contains the information about all of the status codes (which contain the results of the browser's requests to the Web server) that were returned by the Web site server for each hour in the specified time interval.

Default reports using this data (Group-Name; Report-Name): Status Codes; Hourly Status Codes

Processes that use this data set as input: None

Display 18.14 Variable Descriptions

NAME	TYPE	LENGTH	DESCRIPTION	ROLE
DATE	NUM	8	Date of the first day in the summary level	CLASS
FILE_COUNT	NUM	8	The number of requests with the designated status code	ANALYSIS
HOUR	NUM	8	The hour in which the data was collected, based on a 24-hour clock (hour 0 is midnight-12:59 a.m., etc.)	CLASS
PAGE_COUNT	NUM	8	The number of requests that are also valid page views with the designated status code. A page view is loaded from the server when the status code for the page ranges from 200 to 299 or is 304.	ANALYSIS
SESSION_COUNT	NUM	8	The total number of sessions in which the status code occurred	ANALYSIS
STATUS_CODE	NUM	8	The code that was returned by the server to the visitor's browser to report the outcome of a request	CLASS

Page_Dates Data Set

Library: Summary

Data set name: Page_Dates

Summary levels: Not Applicable

Description: This data set is a utility data set that is used for the Interactive Funnel report. It contains the variable Dates, which is used to populate a drop-down menu in the SAS Web Analytics Report Viewer.

Default reports using this data (Group-Name; Report-Name): None

Processes that use this data set as input: None

Note: A stored process creates the data set for the Funnel report. △

Page_<Summary Level> Data Set

Library: Summary

Data set name: Page_<Summary Level>

Summary levels: Each analysis variable is summed for the summary level of the corresponding data set:

_Day

_Week

_Month

_Qtr

_Year

Description: This data set contains a distribution of the pages that were requested. This data set enables you to identify the pages and families of pages that are most often viewed. You can also view session and error counts and percentages for the pages. This information can be used to define business segments and to modify or optimize paths.

Default reports using this data (Group-Name; Report-Name): Navigation; Page Frequency

Processes that use this data set as input: None

Display 18.15 Variable Descriptions

NAME	TYPE	LENGTH	DESCRIPTION	ROLE
CLIENT_ERROR_COUNT	NUM	8	The total number of client errors for a given page. Client errors are identified as requests whose status codes range from 400 to 499.	ANALYSIS
DATE	NUM	8	Date of the first day in the summary level	CLASS
ENTRY_PAGE_COUNT	NUM	8	Entry Page Count	ANALYSIS
ERR_302_COUNT	NUM	8	302 Status: Found	ANALYSIS
ERR_304_COUNT	NUM	8	304 Status: Not Modified	ANALYSIS
ERR_400_COUNT	NUM	8	400 Status: Bad Request	ANALYSIS
ERR_401_COUNT	NUM	8	401 Status: Unauthorized	ANALYSIS
ERR_403_COUNT	NUM	8	403 Status: Forbidden	ANALYSIS
ERR_404_COUNT	NUM	8	404 Status: Not Found	ANALYSIS
ERR_405_COUNT	NUM	8	405 Status: Method Not Allowed	ANALYSIS
ERR_408_COUNT	NUM	8	408 Status: Request Time-out	ANALYSIS
ERR_500_COUNT	NUM	8	500 Status: Internal Server Error	ANALYSIS
ERR_501_COUNT	NUM	8	501 Status: Not Implemented	ANALYSIS
EXIT_PAGE_COUNT	NUM	8	Exit Page Count	ANALYSIS
FILE_COUNT	NUM	8	The number of file views associated with a given page or URI.	ANALYSIS
PAGE_COUNT	NUM	8	The number of page views	ANALYSIS
REQUESTED_FILE	CHAR	1024	The page or URI (Uniform Resource Identifier) that was visited on the site. URIs are formatted strings that identify a resource through a name, location, or any other characteristic.	CLASS
SERVER_ERROR_COUNT	NUM	8	The total number of server errors for a given page. Server errors are identified as requests whose status codes range from 500 to 599.	ANALYSIS
SESSION_COUNT	NUM	8	The total number of sessions in which visitors viewed the page or URI	ANALYSIS

Page_Status_<Summary Level> Data Set

Library: Summary

Data set name: Page_Status_<Summary Level>

Summary levels: Each analysis variable is summed for the summary level of the corresponding data set:

_Day
_Week
_Month
_Qtr
_Year

Description: This data set contains information about the frequency of status codes by the page that was requested on the site. Status codes that range from 400 to 499 indicate the errors that were committed by the visitor's browser. Status codes that range from 500 to 599 indicate the errors that occurred on the Web site server.

Default reports using this data (Group-Name; Report-Name): Status Codes; Page Error Counts

Processes that use this data set as input: None

Display 18.16 Variable Descriptions

NAME	TYPE	LENGTH	DESCRIPTION	ROLE
DATE	NUM	8	Date	CLASS
FILE_COUNT	NUM	8	The number of requests with the designated status code.	ANALYSIS
PAGE_COUNT	NUM	8	The number of requests that are also valid page views with the designated status code. A page view is loaded from the server when the status code for the page ranges from 200 to 299 or is 304.	ANALYSIS
REQUESTED_FILE	CHAR	1024	Requested File	CLASS
SESSION_COUNT	NUM	8	The number of sessions that had a minimum of one request with the designated status code.	ANALYSIS
STATUS_CODE	NUM	8	The code that was returned by the server to the visitor's browser to report the outcome of a request	CLASS

Pathing Data Set

Library: Summary

Data set name: Pathing

Summary levels: None

Description: The Pathing data set is used as input to the path analysis module in the SAS Web Analytics software. The data contains every request that was made by every session for a time interval of up to 30 days. The Requested_File field is used for the Entry Paths reports. The Sequenced_Requested_File field is used for the Entry Paths and the Referrer Entry Paths reports.

Default reports using this data (Group-Name; Report-Name):

Path Analysis; Entry Paths

Path Analysis; Referrer Entry Paths

Processes that use this data set as input: Pathing, funnel processes

Display 18.17 Variable Descriptions

NAME	TYPE	LENGTH	DESCRIPTION	ROLE
DATE	NUM	8	Date	CLASS
DATETIME	NUM	8	Date and time	ANALYSIS
ENTRY_POINT	NUM	8	Session Starts	ANALYSIS
REFERRER	CHAR	1024	Referrer	CLASS
REQUESTED_FILE	CHAR	1024	Requested File	CLASS
SEQUENCE	NUM	8	Indicates which click in the session	ANALYSIS
SEQUENCED_REQUESTED_FILE	CHAR	1037	The requested file tagged with its sequence number	CLASS
REFERENCE	CHAR	12	Numeric id for the requested file	CLASS
SEQUENCED_REFERENCE	CHAR	12	Numeric id for the sequence requested file	CLASS
SESSION_ID	CHAR	245	Session ID	CLASS

Pathing_Dmdb Data Set

Library: Summary

Data set name: Pathing_Dmdb

Summary levels: None

Description: The Pathing_Dmdb data set is created from the Pathing data set. It is a formatted data-mining data set that is required by PROC PATH as input. A numeric code (reference and sequenced reference) for the URI is used instead of the original URI variable.

Default reports using this data (Group-Name; Report-Name): None

Processes that use this data set as input: Pathing

Display 18.18 Variable Descriptions

NAME	TYPE	LENGTH	DESCRIPTION	ROLE
DATE	NUM	8	Date	CLASS
SEQUENCE	NUM	8	Indicates which click in the session	ANALYSIS
REFERENCE	NUM	5	Numeric id for the requested file	ANALYSIS
SEQUENCED_REFERENCE	NUM	5	Numeric id for the sequence requested file	ANALYSIS
SESSION_ID	CHAR	245	Session ID	CLASS

Pathing_<Summary Level> Data Set

Library: Summary

Data set name: Pathing_<Summary Level>

Summary levels:

R1 – the most current date

R7 – the most current 7 days

R30 – the most current 30 days

Description: These data sets represent the rolling summaries that generate standard pathing reports.

Default reports using this data (Group-Name; Report-Name): Navigation; Paths

Processes that use this data set as input: None

Display 18.19 Variable Descriptions

NAME	TYPE	LENGTH	DESCRIPTION	ROLE
CONF	NUM	8	The confidence measure of the path. The percentage of sessions that go to the last item given they have gone to the previous item	ANALYSIS
COUNT	NUM	8	The number of sessions that followed the entire path	ANALYSIS
SIZE	NUM	8	The number of items in the path	ANALYSIS
SUPPORT	NUM	8	The percentage of sessions that followed the entire path	ANALYSIS
DATE	NUM	8	The first date of the summary level. For example in the R30 data set date is the day of the first day in the 30 days	CLASS
ITEM1	CHAR	1024	First requested file in path	CLASS
ITEM2	CHAR	1024	Second requested file in path	CLASS
ITEM3	CHAR	1024	Third requested file in path	CLASS
ITEM4	CHAR	1024	Fourth requested file in path	CLASS
ITEM5	CHAR	1024	Fifth requested file in path	CLASS
ITEM6	CHAR	1024	Sixth requested file in path	CLASS
ITEM7	CHAR	1024	Seventh requested file in path	CLASS

Platform_<Summary Level> Data Set

Library: Summary

Data set name: Platform_<Summary Level>

Summary levels: Each analysis variable is summed for the summary level of the corresponding data set:

_Day
_Week
_Month
_Qtr
_Year

Description: The Platform_<Summary Level> data set contains a distribution of the different platforms (operating systems) that are used by visitors who navigate the Web site. Your Web site will display differently over different platforms.

Default reports using this data (Group-Name; Report-Name): Platform; Platforms

Processes that use this data set as input: None

Display 18.20 Variable Descriptions

NAME	TYPE	LENGTH	DESCRIPTION	ROLE
DATE	NUM	8	Date of the first day in the summary level	CLASS
PAGE_COUNT	NUM	8	The number of page views per platform type. A page view occurs when a page is of a valid application type and is loaded from a server to a browser when the status code for the page is between 200 and 299, or is 304.	ANALYSIS
PLATFORM	CHAR	40	The platform used while accessing the Web site. A platform is specific computer software designed to control the hardware of a specific data-processing system in order to allow users and application programs to make use of it.	CLASS
SESSION_COUNT	NUM	8	The number of sessions logged per platform type.	ANALYSIS

Referrer_Domain_<Summary Level> Data Set

Library: Summary

Data set name: Referrer_Domain_<Summary Level>

Summary levels: Each analysis variable is summed for the summary level of the corresponding data set:

_Day
_Week
_Month
_Qtr
_Year

Description: This data set contains a distribution of referrer domains, which enables you to determine the origin of visitors. This report provides valuable information that can help you evaluate affiliate campaign strategies and assess incoming referral traffic patterns.

Default reports using this data (Group-Name; Report-Name): Referrer; Session Referrer Domains

Processes that use this data set as input: None

Display 18.21 Variable Descriptions

NAME	TYPE	LENGTH	DESCRIPTION	ROLE
DATE	NUM	8	Date of the first day in the summary level	CLASS
PAGE_COUNT	NUM	8	The total number of viewed pages in sessions that were referred through a specific domain.	ANALYSIS
REFERRER_DOMAIN	CHAR	128	The domain from which the content group visit originated. The domain is a series of alphanumeric strings that are separated by periods (e.g., www.sas.com). The domain is the address of a computer network connection and identifies the owner of the address.	CLASS
SESSION_COUNT	NUM	8	The total number of sessions that were referred through a specific domain.	ANALYSIS

Referrer_Entry_Point_<Summary Level> Data Set

Library: Summary

Data set name: Referrer_Entry_Point_<Summary Level>

Summary levels: Each analysis variable is summed for the summary level of the corresponding data set:

_Day
_Week
_Month
_Qtr
_Year

Description: This data set contains the most common requests that visitors make in order to enter the Web site. This data set enables you to determine which pages are most frequently visited first in a session.

Default reports using this data (Group-Name; Report-Name): Referrer; Referrer Entry Pages

Processes that use this data set as input: None

Display 18.22 Variable Descriptions

NAME	TYPE	LENGTH	DESCRIPTION	ROLE
DATE	NUM	8	Date of the first day in the summary level	CLASS
FIRST_REQUESTED_FILE	CHAR	1024	The first page or URI (Uniform Resource Identifier) that was visited on the site. URIs are formatted strings that identify a resource through a name, location, or any other characteristic.	CLASS
REFERRER_DOMAIN	CHAR	128	The domain from which the content group visit originated. The domain is a series of alphanumeric strings that are separated by periods (e.g., www.sas.com). The domain is the address of a computer network connection and identifies the owner of the address.	CLASS
SESSION_COUNT	NUM	8	The total number of sessions in which visitors entered the Web site via the designated page or URI.	ANALYSIS

Referrer_Page_Status_<Summary Level> Data Set

Library: Summary

Data set name: Referrer_Page_Status_<Summary Level>

Summary levels: Each analysis variable is summed for the summary level of the corresponding data set:

_Day
_Week
_Month
_Qtr
_Year

Description: The Referrer_Page_Status_<Summary Level> data set contains the distribution of the status codes that were returned when visitors requested a page on the Web site. The status codes are tallied by requested page within referring page in order to provide a comprehensive picture of how successfully that the referred visitors navigated to your Web site.

Default reports using this data (Group-Name; Report-Name): Status Codes; Referrer Page Error Codes

Processes that use this data set as input: None

Display 18.23 Variable Descriptions

NAME	TYPE	LENGTH	DESCRIPTION	ROLE
DATE	NUM	8	Date of the first day in the summary level	CLASS
FILE_COUNT	NUM	8	The number of requests with the designated status code.	ANALYSIS
PAGE_COUNT	NUM	8	The number of requests that are also valid page views with the designated status code. A page view is loaded from the server when the status code for the page ranges from 200 to 299 or is 304.	ANALYSIS
REFERRER	CHAR	1024	The page or URI (Uniform Resource Identifier) that was requested immediately prior to the requested file on the Web site. URIs are formatted strings that identify a resource through name, location, or any other characteristic.	CLASS
REQUESTED_FILE	CHAR	1024	The page or URI that was visited on the Web site. URIs are formatted strings that identify a resource through name, location, or any other characteristic.	CLASS
SESSION_COUNT	NUM	8	The number of sessions that had at least one request with the designated status code.	ANALYSIS
STATUS_CODE	NUM	8	Status Code	CLASS

Referrer_Search_Term_<Summary Level> Data Set

Library: Summary

Data set name: Referrer_Search_Term_<Summary Level>

Summary levels: Each analysis variable is summed for the summary level of the corresponding data set:

_Day
_Week
_Month
_Qtr
_Year

Description: This data set contains a distribution of search terms that are used by visitors to find your Web site. This report shows the search terms that are used to reach the Web site through a search engine from a referring party. This information can be useful when determining which keywords to specify for search engines in order to direct new traffic to your Web site.

Default reports using this data (Group-Name; Report-Name): Referrer; Search Terms

Processes that use this data set as input: None

Display 18.24 Variable Descriptions

NAME	TYPE	LENGTH	DESCRIPTION	ROLE
DATE	NUM	8	Date of the first day in the summary level	CLASS
REFERRER_DOMAIN	CHAR	128	The domain from which the content group visit originated. The domain is a series of alphanumeric strings that are separated by periods (e.g., www.sas.com). The domain is the address of a computer network connection and identifies the owner of the address.	ANALYSIS
SEARCH_TERM	CHAR	1024	A search or query that is entered by a visitor at a referring site, which resulted to a link that sent visits to your Web site.	CLASS
SESSION_COUNT	NUM	8	The number of sessions resulting from the use of the search term	ANALYSIS

Status_<Summary Level> Data Set

Library: Summary

Data set name: Status_<Summary Level>

Summary levels: Each analysis variable is summed for the summary level of the corresponding data set:

_Day
_Week
_Month
_Qtr
_Year

Description: This data set contains information about the frequency of status codes. Status codes that range from 400 to 499 indicate the errors that were caused by the visitor's browser. Status codes that range from 500 to 599 indicate the errors that occurred on the Web site's server.

Default reports using this data (Group-Name; Report-Name): Status Codes; Status Codes

Processes that use this data set as input: None

Display 18.25 Variable Descriptions

NAME	TYPE	LENGTH	DESCRIPTION	ROLE
DATE	NUM	8	Date of the first day in the summary level	CLASS
FILE_COUNT	NUM	8	The number of requests with the designated status code.	ANALYSIS
PAGE_COUNT	NUM	8	The number of requests that are also valid page views with the designated status code. A page view is loaded from the server when the status code for the page is between 200 and 299 or is 304.	ANALYSIS
SESSION_COUNT	NUM	8	The number of sessions that had a minimum of one request with the designated status code.	ANALYSIS
STATUS_CODE	NUM	8	The code that was returned by the server to the visitor's browser to report on the outcome of a request.	CLASS

Top_Entry_Paths_<Summary Level> Data Set

Library: Summary

Data set name: Top_Entry_Paths_<Summary Level>

Summary levels:

R1 – the most current date

R7 – the most current 7 days

R30 – the most current 30 days

Description: This data set contains a list of the most popular points of entry into the Web site. This report shows the detailed navigation patterns that visitors follow after they reach the Web site. Understanding these navigation patterns can help you interpret the user experience when visiting your Web site by providing the following navigation patterns:

- the subject areas that attract interest
- the high site exit paths (paths that lead to site abandonment)
- the paths to the significant return on investment (ROI) events
- the most popular links from any entry point

Default reports using this data (Group-Name; Report-Name): Navigation; Entry Paths

Processes that use this data set as input: None

Display 18.26 Variable Descriptions

NAME	TYPE	LENGTH	DESCRIPTION	ROLE
CONF	NUM	8	(confidence) The percentage of all sessions whose visitors visit the last page or URI in a set, after the visitors visit all previous pages or URIs in the set. The value is calculated as follows: Confidence = (the number of sessions whose visitors visit every page or URI in the set / the number of sessions whose visitors visit every page or URI in the set before visiting the last page or URI) * 100.	ANALYSIS
COUNT	NUM	8	The number of sessions that traversed the path.	ANALYSIS
SIZE	NUM	8	Used by the display to size the lines between notes (not used in SWA version 5.0)	ANALYSIS
SUPPORT	NUM	8	The percentage of sessions that traversed the path (Item 1 --> Item 2 --> ... --> Item N).	ANALYSIS
DATE	NUM	8		CLASS
ITEM1	CHAR	1037	Represents the page or URI (Uniform Resource Identifier) that specifies the first request in a session.	CLASS
ITEM2	CHAR	1037	Represents the page or URI that specifies the second request in a session.	CLASS
ITEM3	CHAR	1037	Represents the page or URI that specifies the 3 rd request in a session.	CLASS
ITEM4	CHAR	1037	Represents the page or URI that specifies the 4th request in a session.	CLASS
ITEM5	CHAR	1037	Represents the page or URI that specifies the 5th request in a session.	CLASS
ITEM6	CHAR	1037	Represents the page or URI that specifies the 6th request in a session.	CLASS

The default number of items that can exist in a path for the SAS Web Analytics application is 7. The minimum number of sessions that are allowed for 1 day is 10 sessions; for 7 days is 40 sessions, and for 30 days is 100 sessions.

Top_Referrer_Paths_<Summary Level> Data Set

Library: Summary

Data set name: Top_Referrer_Paths_<Summary Level>

Summary levels:

R1 – the most current date

R7 – the most current 7 days

R30 – the most current 30 days

Description: This data set contains a list of the referrers that sent the most traffic to your Web site. This data set is designed to show the detailed navigation patterns that visitors follow after they reach the site from different referring agents. Understanding these navigation patterns can help you to interpret the user experience from each referrer by providing the following information:

- the areas of your Web site that attract the most interest
- the high site exit paths (paths that lead to site abandonment)
- the paths that result in significant ROI (return on investment) events
- the areas of your Web site to which affiliate programs are sending visitors
- which affiliate programs are performing well or performing poorly
- which banner advertisements are performing well or performing poorly

Default reports using this data (Group-Name; Report-Name): Navigation; Referrer Paths

Processes that use this data set as input: None

Display 18.27 Variable Descriptions

NAME	TYPE	LENGTH	DESCRIPTION	ROLE
CONF	NUM	8	(Confidence) The percentage of all sessions whose visitors visit the last page or URI in a set, after the visitors visit all previous pages or URIs in the set. The value is calculated as follows: Confidence = (the number of sessions whose visitors visit every page or URI in the set / the number of sessions whose visitors visit every page or URI in the set before visiting the last page or URI) * 100.	ANALYSIS
COUNT	NUM	8	The number of sessions that traversed the path.	ANALYSIS
SIZE	NUM	8	Used by the display to size the lines between notes (not used in SWA version 5.0)	ANALYSIS
SUPPORT	NUM	8	The percentage of sessions that traversed the path (Item 1 → Item 2 → ... → Item N).	ANALYSIS
DATE	NUM	8	Date of the first day in the summary level	CLASS
ITEM1	CHAR	1037	Represents the page or URI (Uniform Resource Identifier) that specifies the first request in a session.	CLASS
ITEM2	CHAR	1037	Represents the page or URI that specifies the second request in a session.	CLASS
ITEM3	CHAR	1037	Represents the page or URI that specifies the 3 rd request in a session.	CLASS
ITEM4	CHAR	1037	Represents the page or URI that specifies the 4th request in a session.	CLASS
ITEM5	CHAR	1037	Represents the page or URI that specifies the 5th request in a session.	CLASS
ITEM1	CHAR	1037	Represents the page or URI (Uniform Resource Identifier) that specifies the first request in a session.	CLASS
ITEM2	CHAR	1037	Represents the page or URI that specifies the second request in a session.	CLASS

The default number of items that can exist in a path for the SAS Web Analytics application is 7. The minimum number of sessions that are allowed for 1 day is 10 sessions, for 7 days is 40 sessions, and for 30 days is 100 sessions.

Visitor_<Summary Level> Data Set

Library: Summary

Data set name: Visitor_<Summary Level>

Summary levels: Each analysis variable is summed for the summary level of the corresponding data set:

_Day
_Week
_Month
_Qtr
_Year

Description: This data set contains data which identifies the visitors that have the highest activity on your site.

Default reports using this data (Group-Name; Report-Name): Visitor; Unique Visitor

Processes that use this data set as input: None

Display 18.28 Variable Descriptions

NAME	TYPE	LENGTH	DESCRIPTION	ROLE
DATE	NUM	8	Date of the first day in the summary level	CLASS
DURATION	NUM	8	Total time (in seconds) for all sessions	ANALYSIS
PAGE_COUNT	NUM	8	Total number of pages viewed	ANALYSIS
PAGE_VIEW_2TO4_IND	NUM	8	The number of sessions with two to four page hits	ANALYSIS
PAGE_VIEW_GE5_IND	NUM	8	The number of sessions with more than five page hits	ANALYSIS
PAGE_VIEW_LE1_IND	NUM	8	The number of sessions with zero to one page hit	ANALYSIS
SESSION_COUNT	NUM	8	Total number of sessions	ANALYSIS
BEG_DATETIME	NUM	8	Beginning Session Datetime	ANALYSIS
END_DATETIME	NUM	8	Ending Session Datetime	ANALYSIS
VISITOR_ID	CHAR	225	Visitor Identifier	CLASS

Glossary

active session

a session that is still in progress for either of the following reasons: 1) the visitor is currently requesting resources such as pages, pictures, or files at the Web site, or 2) the visitor has not made any requests for a while, but the session timeout (usually 30 minutes) has not been reached.

bytes received

the number of bytes that a Web server has received from a particular client browser. Most Web server log files do not record bytes received. See also bytes sent.

bytes sent

the total number of bytes that a server has delivered in response to a request. Because of retransmissions and network problems, bytes sent can sometimes be larger than the size in bytes of the resource or file that was received. Bytes sent is sometimes referred to as bytes transferred.

bytes transferred

See bytes sent.

CAM (custom access module)

a SOURCE entry containing SAS code that users can change in order to change how data in a Web server log file is processed by either the Extract program or the Load program.

clickstream analysis

the analysis and interpretation of the actions of Web site visitors. These actions are recorded in the Web log as a chain of time-ordered related events, such as a trail of mouse clicks that a visitor leaves. The purpose of clickstream analysis is to understand and predict the actions of visitors as well as the paths that visitors take through a site. This analysis typically involves data-mining techniques such as identifying sequences and associations.

clickstream reporting

the process of summarizing the actions that are recorded in a Web server log file into various classes, dimensions, or buckets. The summarization is based on the visitors' URLs, the amount of time spent on each page, and elements of the domain names. This reporting describes demographic information about the visitor population, the site activity rates, and the relative demand for various areas of the site, such as ad banners or links on a page.

clickthrough rate

the percentage of times that visitors to a Web page click on a banner advertisement.

entry page

the first page that a visitor views when entering a Web site.

entry point

the first page that an Internet visitor views when visiting a Web site. In WebHound, the entry point page marks the start of a session. See also exit point.

exit page

the last page that a visitor views before leaving a Web site.

exit point

the last page that a visitor views before leaving a Web site. In WebHound, the exit point marks the end of a session.

file count

the total number of files that a particular Web site visitor downloads during a session. See also hit, page request.

file hit

See hit.

funnel report

a report that enables a user to analyze the behavior of visitors to a Web site by providing a graphical display of the visitor drop-off rates through a sequence of pages that resulted in a desired transaction.

hit

the result of a successful request (sent to a Web server) for a resource such as an HTML page, a GIF file, or an executable file. Each hit generates an entry in a Web server log file. By contrast, a page request (a particular type of hit) does not include the objects on the page. Requests for an HTML file and a GIF file are both considered to be hits, but only the request for the HTML file is typically considered to be a page request. See also page request.

page count

the total number of pages that the SAS e-Data ETL application has identified in a Web server log file. The page count does not include objects on a Web page, such as GIF files or audio files. Page count and page views are synonyms. See also file count, hit.

page request

an attempt to access a Web page. Each page request generates an entry in a log file. Unlike a hit, a page request does not include the objects on the page, such as GIF files and audio files. A hit includes all objects on the page as well as the page itself. See also visit, hit.

referrer ID

the URL of the Web page that a visitor clicked on in order to visit the current page.

report definition

a specification that is used for generating a report. A report definition includes information such as the table and level, the names of the variables, the report style, and other attributes.

request

an attempt to access a Web page or a resource on a Web server. A request can be either a page request or a hit. See also page request, hit.

segment

a group of Web site visitors with one or more common attributes that have been identified by a rule. Segments are created by using a type of predictive model called a decision tree. The decision tree uses a set of independent variables to determine whether a visitor will return to the Web site at some time in the future.

session

a period of activity that starts when a visitor first accesses a particular Web site and that ends when the visitor has not performed any actions at that Web site within a specified time interval (usually 30 minutes). A session ID is associated with each session, and the activity that occurs during the session is recorded in a Web server log file.

session ID

a unique number that is assigned to a Web site visitor and which is used to track the visitor's path and the time of entry and exit.

session start

the time of a visitor's request for an entry point page of your Web site. The session start time is recorded in the Web server log file. See also session, entry point.

status code

in a Web server log file, a three-digit code that the server issues to describe the success or failure of a visitor's request for a file from a Web site. A status code between 200 and 299 indicates that the request was successful. A status code of 400 or greater indicates a bad request, an unauthorized request, a page not found, or some other type of failure.

stored process

a SAS program that is stored on a server and which can be executed as requested by client applications. There are two types of stored processes: IOM Direct Interface Stored Processes and SAS Stored Processes.

ubiquitous identifier

a variable whose value remains constant for all of a visitor's clicks in a Web site during a visit. One or more ubiquitous identifiers are specified in the Wbconfig data set of a Web mart and are used to track the clickstream activity of a unique Web site visitor.

unique visitors

the number of individuals who visit your Web site within a specified reporting period (hour, day, week, or month, depending on the report that you select). A unique visitor can have more than one session during the reporting period. See also session.

visit

an instance of a person using a Web browser to access one or more files on a Web site.

Web mart

a generic data mart that contains Web log information. Web mart is a shortened form of the term Web data mart. See also webhouse.

webhouse

a data warehouse that contains data from the log files of Web servers or proxy servers. A webhouse can contain multiple Web marts.

Index

- A**
- Actual Values column 159
 - administration tasks 16
 - Advanced Customizations frame 28, 52
 - modifying CAMs 53
 - Advanced Customizations window 62
 - Aggregate_analysis_stats module 92
 - Aggregate_job_datasets module 92
 - Analysis_jobn.sas modules 95
 - architecture 4
- B**
- bad status codes 49
 - browsers
 - identifying 43
 - matching Web log contents with Control.Wbbrowsr table 44
 - buckets
 - restructure buckets 52
 - business direction of scorecard metrics 151
 - business objectives
 - capabilities for achieving 3
 - financial and service organizations 3
 - retail industries 3
 - telecommunications organizations 3
- C**
- CAMs (custom access modules) 52
 - copying 53
 - deleting from Work library 59
 - modifying 53, 55
 - CGI delimiter 42
 - CGI parameters
 - setting visitor ID values from 75
 - CGI Params_n detail table 107
 - CGI program locations 43
 - CGIParms detail table 32
 - Checkpoint_load_init module 100
 - Checkpoint_load_term module 105
 - Check_sas_logs_for_metrics module 94
 - Clean_up_working_areas module 100
 - CLF (Common Log Format) 74
 - client tier 4
 - Common Log Format (CLF) 74
 - compressed files
 - processing compressed Web server log files 50
 - properties 50
 - Compressed Files frame 28, 50
 - configuration
 - Control.Wbconfig table 61
 - Control library 61
 - customizing metadata tables 61
 - table descriptions 61
 - Control.Wbbrowsr table
 - list of Web browsers 43
 - matching Web log contents with 44
 - Control.Wbconfig table 61
 - Control.Wbfields table 61
 - Control.Wbinfile table 61
 - setting INFILE option in 37
 - Control.Wbpagview table 61
 - MIME type as page view 49
 - Control.Wbpgms table
 - CGI program locations 43
 - interpretation assignments for URLs 43
 - interpretation values of executable URLs 43
 - Control.Wbplfrm table
 - list of platforms 44
 - matching Web log contents with 45
 - Control.Wbspecl table 61
 - adding IP addresses to 47
 - Control.Wbspider table 61
 - editing contents of 48
 - cookie delimiter 42
 - cookies
 - setting visitor ID values from 75
 - Cookies detail table 32
 - Cookies_n detail table 107
 - Create_formats module 80
 - Create_generations module 100
 - Create_unique_urls module 100
 - custom access modules
 - See CAMs (custom access modules)
 - custom data sources 9
 - custom directory structure
 - loading data into 70
 - custom Web logs 64
 - adding date to 67
 - adding support for 64
 - adjusting time zone 67
 - data reader errors 66
 - macro variables 64
 - skipped records or fields 67
 - customizing data 17
 - Custom_log_format module 81
 - Custom_log_input module 81
 - Custom_log_rxfree module 81
 - Custom_log_rxmatch module 81
 - Custom_log_rxpars module 81
 - Custom_log_set_server_type module 81
 - Custom_log_test_harness module 81
- D**
- daily detail data sets 205
 - Daily.sas program 10, 69, 187
 - macros run by 10
 - Dashboard 15
 - dashboard metrics 163
 - adding to dashboard 168
 - modifying 166
 - specifying information for 164
 - Dashboard reports 14
 - dashboards 163
 - creating 165
 - data requirements 163
 - testing output 170
 - Dashboards page 128
 - DASHBOARD_TO_RUN= argument
 - %WADECIDE macro 182
 - data directory 6
 - data flow 10
 - Data page 119
 - customizing graphical components of reports 121
 - customizing summarization parameters 120
 - customizing user interface settings 121
 - data reader errors 66
 - data set options
 - for detail tables 32
 - data sets
 - daily detail data sets 205
 - for storing scorecard metrics 151
 - session data sets 207
 - Waadmsum 193
 - Waconfig 200
 - data sources 8
 - DATA_STORE= argument
 - %WAETL macro 185
 - date ranges
 - in Web Mart Status table 113

DATE1= argument
 %WADECIDE macro 181

DATE2= argument
 %WADECIDE macro 181

dates
 adding to Web logs 67
 in Web logs 76

Datetime_logic module 85

decision support 180

DETAIL= argument
 %WAETL macro 186

Detail library
 contents after running Load process 105
 contents of 105
 detail tables 105

detail tables 105
 data set options for 32
 descriptions of 32
 generations of 34
 history tables to keep 34
 properties 31

Detail Tables frame 28, 31

Determine_available_weblogs module 81

Determine_browser_and_platform module 87

Determine_executable_program module 86

Determine_organization module 87

Determine_server_and_sitename module 85

Determine_unique_url module 88

Determine_weblog_type module 82

directory structure 4

DNS lookup 40
 reverse domain name resolution 40
 script 41

domain names
 reverse domain name resolution 40

drillable reports 144
 adding filters to 146
 examples 144
 Link field value 145

E

e-data-etl directory 6

%EDATAETL macro
 extracting and loading Web log data 71
 running Extract process 77
 running Load process 98

ELF (Extended Log Format) 74

ETL processes 10
 administrative tasks 17
 Daily.sas program for 69

executable programs 43

executable URLs 43

Execute_the_analysis_jobs module 92

Execute_the_data_read_jobs module 91

Execution Tuning frame 28, 51

execution tuning properties 51
 maximum number of open files 52
 parallel jobs properties 51
 restructure buckets 52

Extended Log Format (ELF) 74

Extract process 71
 creating SAS reader programs 74
 customizing 12
 data reader errors 66
 dates in Web logs 76

determining visitors and sessions 74

determining Web server type 74

functions of 73

identifying available Web logs 73

modifying modules in 76

modules in 80

overview 73

parallel processing 51

reading Web logs 11

running 77

running individually 71

sorting data 74

temporary data sets and 11

temporary working area 95

Extract_build_filenames module 82

Extract_initialization module 91

F

fields
 skipped 67
 Wbfields table 61

files
 maximum number of open files 52
 temporary files 34

Filtering frame 28, 45

filtering properties 45
 actions 46
 bad status codes 49
 Force Non-Page View action 46
 Keep action 46
 non-pages 48
 Skip action 46
 special client list 47
 spiders 47
 Tally action 46

filters
 adding to drillable reports 146
 variables as filter values 146

financial organizations 3

Force Non-Page View action 46

Funnel reports 14, 16
 stored process for 175

G

generations
 for detail tables 34

goal-seeking table 160

graphical components of reports
 customizing 121

H

history tables 34

hits 48

host names
 Control.Wbspec table 61

hyperlinks
 variables as 147

I

\$ID2URL. format 109

IIS (original) 74

INDSN= argument
 %WADECIDE macro 181

INFILE options
 Control.Wbinfile table 61
 for Web log types 37
 setting in Control.Wbinfile table 37

input
 custom data sources 9
 data sources 8
 Web server log files 8

input metrics
 changing to target metrics 155
 data set for storing 151
 for scorecards 151
 guidelines for choosing 150

IP addresses
 adding to Control.Wbspec table 47
 Control.Wbspec table 61
 resolving to host names 64
 reverse domain name resolution 40
 visitor tracking and 74

iPlanet 74

J

Job_build_data_attributes module 83

Job_build_inputs module 83

Job_build_main module 82

Job_build_output_logic module 91

K

Keep action 46

L

libraries
 allocating for Web marts 72

Load process 12, 71
 contents of Detail library after running 105
 Detail library 105
 modules of 97, 99
 overview 97
 running 98
 running individually 71

loading data
 into custom directory locations 70

M

macro variables
 custom Web logs 64

macros 179
 run by Daily.sas program 10
 %WADECIDE 180
 %WAETL 184

Main_variable_assignments module 85

- metadata tables
 - customizing 61
- metrics
 - See dashboard metrics
 - See scorecard metrics
- Microsoft Proxy 74
- middle tier 4
- MIME type
 - as page view 49
 - Control.Wbpagview table 61
- misc directory 5
- missing input information 161

N

- NAME= argument
 - %WAETL macro 185
- Name variable 159
- navigation 112
- Netscape/iPlanet 74
- non-pages 48
 - default action for 49
 - MIME type as page view 49

O

- open files
 - maximum number of 52
- operating system
 - number of open files 52
- OUTLIB= argument
 - %WADECIDE macro 181
- Output_cookie module 90
- Output_pathinfo_parms module 90
- Output_query_string_parms module 90
- Output_referrer_pathinfo_parms module 90
- Output_referrer_query_parms module 90

P

- page views 48
 - MIME type as 49
- Page_logic module 87
- parallel jobs
 - properties 51
 - SAS session options for 37
- parallel processing
 - number of jobs created 51
- Parse_main_url module 86
- Parse_referrer module 87
- Parsing frame 28, 41
- parsing properties 41
 - CGI delimiter 42
 - CGI program locations 43
 - cookie delimiter 42
 - identifying platforms 44
 - identifying Web browsers 43
 - number of URL levels to parse 42
 - starting level of URL parsing 42
 - URL_Prefix field 42
- Path Analysis reports 14, 15
- Pathing detail table 32
- Pathing_n detail table 108
- performance
 - compressed files and 50
 - non-pages and 48
- Performance column 159
- platforms
 - identifying 44
 - matching Web log contents with Control.Wbpltfm table 45
- Predicted Values column 159
- processing
 - compressed Web server log files 50
 - customizing 61
 - maximum number of sessions 51
- Processing frame 28, 35
- processing properties 35
 - adding/deleting Web server log fields 38
 - DNS lookup 40
 - INFILE options for Web log types 37
 - SAS session options 37
- PROGRAM= argument
 - %WADECIDE macro 180
 - %WAETL macro 186
- Properties window 30
 - Advanced Customizations frame 52
 - Compressed Files frame 50
 - customizing a single property 31
 - customizing multiple properties 31
 - Detail Tables frame 31
 - Execution Tuning frame 51
 - Filtering frame 45
 - Parsing frame 41
 - Processing frame 35
 - property frames 28
 - selecting Web marts 30
 - Temporary Location frame 34
 - Web Log Location frame 35

R

- re-summarizations 198
- reader programs 74
- Read_jobn.sas modules 95
- records
 - skipped 67
- ReferrerParms detail table 33
- ReferrerParms_n detail table 107
- registering Web marts 24
- Relative Difference column 160
- report categories 14
- report definitions
 - creating 137
 - stored processes for 175
- report groups 131
 - creating 133
 - deleting 136
 - modifying 135
- Report Viewer 13
- reporting process
 - customizing 12
- reports
 - alphabetical list of 191
 - creating 137
 - customizing 18
 - customizing graphical components of 121
 - customizing user interface settings 121
 - drillable reports 144
 - list of 189, 191
 - non-traffic reports 191
 - refining 139
 - Segmentation reports 171
 - setting up 15, 18
 - supplied reports 14
 - Traffic reports 189
 - variables as filter values 146
 - variables as hyperlinks 147
 - variables for 146
- restructure buckets 52
- retail industries 3
- reverse domain name resolution 40
- robots 47
 - Control.Wbspider table 61

S

- SAS Business Intelligence architecture 4
- SAS e-Data ETL 2
 - customizing processing 61
- SAS e-Data ETL Administrator
 - creating Web marts 20
- SAS reader programs 74
- SAS Web Analytics 2
 - administration tasks 16
 - macros 179
 - purpose of 2
- SAS Web Analytics Administrator
 - copying Web marts 117
 - Dashboards page 128
 - Data page 119
 - deleting Web marts 117
 - editing Web marts 116
 - interface 113
 - main menu 112
 - navigating 112
 - Scorecards page 127
 - Segmentations page 129
 - Summaries page 125
 - Traffic page 123
 - updating Web marts 118
 - Web Marts page 115
- SAS Web Analytics Report Viewer 13
- sashelp directory
 - customizing metadata tables in 61
 - modifying catalogs in 62
- sashelp library 5
- sasmacros library 5
- sasmisc library 5
- Save_macro_variable_settings module 82
- Scorecard 15
- scorecard metrics 149, 154
 - business direction of 151
 - changing input metric to target metric 155
 - data set for storing 151
 - input metrics 150
 - measurement range of 151
 - types of 151
- Scorecard reports 14
 - table layout of 158
- scorecard table
 - Actual Values column 159
 - Name variable 159
 - Performance column 159
 - Predicted Values column 159

- Relative Difference column 160
- variables of 159
- scorecards 149
 - adding non-Web log variables to 156
 - creating 152
 - data requirements 150
 - defining 150
 - goal-seeking table 160
 - input information 151
 - missing input information 161
 - running custom scorecards 152
 - table layout of report 158
 - testing output 157
 - viewing 158
- Scorecards page 127
- SCORECARD_TO_RUN= argument
 - %WADECIDE macro 182
- search engines 47
- Segmentation reports 14, 15
 - creating 171
 - data requirements 171
 - running 174
 - testing 174
- Segmentation Wizard 171
- Segmentations page 129
- server tier 4
- servers
 - session continuity across 37
- service organizations 3
- session data sets 207
- Session Summary Control category 205
- sessions
 - continuity across servers 37
 - defining 75
 - defining duration 75
 - definition 36
 - determining 74
 - for processing tasks 51
 - maximum number of 51
 - SAS options for parallel jobs 37
 - splitting 36
 - timeout threshold 75
- Skip action 46
- skipped records or fields 67
- Sort_buckets module 91
- sorting data
 - Extract process 74
- special client list 47
 - default action 47
 - editing table contents 47
 - wildcard character in 47
- Special_client_logic module 85
- Spider_client_logic module 85
- spiders 47
 - Control.Wbspider table 61
 - default action for 48
 - editing Control.Wbspider table 48
 - identifying new spiders 48
- status codes 49
- Status_code_logic module 87
- stored processes
 - adding 176
 - for Funnel reports 175
 - for report definitions 175
 - testing 177
- summaries 125, 195

- Summaries page 125
 - creating summarizations 126
 - deleting summarizations 127
 - modifying summarizations 127
- summarizations
 - creating 126
 - customizing parameters 120
 - deleting 127
 - modifying 127
 - re-summarizations 198
- summarizing process
 - customizing 12
- SUMMARY engine 193
 - initial summarizations 196
 - re-summarizations 198
- supplied reports 14
- SWAMART= argument
 - %WADECIDE macro 181
 - %WAETL macro 186
- system options
 - for parallel jobs 37

T

- Tally action 46
- target metrics
 - changing input metrics to 155
 - data set for storing 151
 - for scorecards 151
- telecommunications organizations 3
- temporary data sets
 - Extract process and 11
- temporary files
 - changing path for 35
 - deleting 35
 - directory path for 34
- Temporary Location frame 28, 34
- temporary location properties 34
 - changing path for temporary files 35
- deleting temporary files 35
- temporary working area
 - Extract process and 95
- temporary working directory 7
- TEMP_STORE= argument
 - %WADECIDE macro 181
 - %WAETL macro 185
- three-tier architecture 4
- time zone
 - adjusting 67
- timeout threshold 75
- Traffic page 123
 - copying items 124
 - creating items in 123
 - deleting items 124
 - modifying items 124
- Traffic reports 14
 - list of 189
- tuning
 - See* execution tuning properties

U

- Unique_URL detail table 109
- Unique_URLs detail table 33

- URL parsing
 - number of levels to parse 42
 - starting level of 42
- URL_Prefix field 42
- URLs
 - executable URLs 43
 - interpretation assignments for 43
 - interpretation values of executable URLs 43
 - Unique_URLs detail table 33
- USE_FIRST_TWO_WEEKS= argument
 - %WADECIDE macro 182
- user interface settings
 - customizing 121
- User_assignments_after_input CAM 76
- User_assignments_after_input module 83
- User_assignments_before_output CAM 76
- User_assignments_before_output module 88
- User_assignments_by_session_ID CAM 76
- User_assignments_by_session_ID module 92
- User_assignments_for_data_mining CAM 97
- User_assignments_for_data_mining module 82, 100
- Usermods library 61
 - modifying 62

V

- variables
 - as filter values 146
 - as hyperlinks 147
 - in scorecard table 159
 - non-Web log, adding to scorecards 156
 - setting for reports 146
- visitor ID 74
 - setting values from CGI parameters 75
 - setting values from cookies 75
- Visitor_ID_logic module 86
- visitors
 - determining 74
 - tracking 74

W

- Waadmsum data set 193
 - modifying 199
 - variables in 194
- Waconfig data set 200
 - changing values in 204
 - customizing session summary data sets 207
 - editing 204
 - example 200
 - parameters in 200
 - parameters in Session Summary Control category 205
 - web_init_by_sess_vars parameter 205
- %WADECIDE macro 180
 - details 180
 - examples 184
 - notes 183
 - purpose of 180
 - syntax 180
 - testing dashboard output 170
 - testing scorecard output 157

- %WAETL macro 10, 184
 - creating Web marts 19
 - custom invocation of 70, 188
 - details 185
 - functions of 20
 - initializing Web marts 186
 - loading data into Web marts 187
 - purpose of 184
 - syntax 185
 - Waconfig data set and 200
- Web browsers
 - identifying 43
 - matching Web log contents with Control.Wbbrowsr table 44
- Web log formats 9
- Web Log Location frame 28, 35
- Web log location properties 35
 - setting Web log location 35
- Web logs
 - adding/deleting fields 38
 - closing off 63
 - compressing 64
 - custom Web logs 64
 - dates in 76
 - determining Web server type 74
 - extracting and loading data 71
 - identifying available logs 73
 - INFILE options for various types 37
 - location of 35
 - management guidelines 63
 - matching contents with Control.Wbbrowsr table 44
 - matching contents with Control.Wbpltfm table 45
 - processing a single file 35
 - processing all files in a directory 35
 - reading 74
 - renaming 63
 - setting location of 35
 - transferring to SAS Web Analytics host 64
- Web logs directory 7
- Web mart directories
 - creating 7
- Web mart properties 28
 - See also* Properties window
 - advanced customizations properties 52
 - canceling changes 29
 - compressed files properties 50
 - customizing 28
 - customizing a single property 31
 - customizing multiple properties 31
 - detail tables properties 31
 - editing 28
 - execution tuning properties 51
 - filtering properties 45
 - parsing properties 41
 - processing properties 35
 - saving changes 29
 - table of 28
 - temporary location properties 34
 - Web log location properties 35
- Web Mart Status table
 - date ranges in 113
- Web marts
 - allocating libraries for 72
 - copying 117
 - creating 19
 - creating with SAS e-Data ETL Administrator 20
 - creating with %WAETL macro 19
 - deleting 117
 - directory structure 4
 - editing 116
 - initializing 186
 - loading data into 187
 - registering 24
 - setting up 16, 30
 - updating 118
- Web Marts page 115
 - copying Web marts 117
 - deleting Web marts 117
 - editing Web marts 116
 - updating Web marts 118
- Web server log files 8
 - adding/deleting fields 38
 - compressing 50
 - editing fields 38
 - Extract process and 11
 - field variable parameters 39
 - parsing 41
 - processing compressed files 50
- Web servers
 - determining type of 74
- webdatasrv directory 5
- web_init_by_sess_vars parameter 205
- Weblog_Detail detail table 33
- Weblog_Detail_n detail table 106
- Webmart Location frame 28
- wildcard character
 - in special client list 47
- Work library
 - deleting CAMs from 59

Your Turn

If you have comments or suggestions about *SAS Web Analytics 5.1: Administrator's Guide*, 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

