SAS/Warehouse Administrator™ Usage and Enhancements

Terry Lewis, SAS Institute Inc., Cary, NC

ABSTRACT

SAS/Warehouse Administrator software makes it easier to build, maintain, and access data warehouses by bringing the strengths of the SAS® System relative to data warehousing together into one consistent interface plus adding additional capabilities not currently available in other SAS system products.

This paper explains how to use SAS/Warehouse Administrator and describes many of the enhancements that have been added to the first release of SAS/Warehouse Administrator since it was first introduced in an experimental form at SUGI 21 in 1996.

INTRODUCTION

The SAS system has long been used to provide data warehouse functionality even before the term was coined. In 1995, we at SAS Institute recognized that it would be helpful to supply a framework that would gather together many of the tools in the SAS System that lend themselves to data warehousing into a single product. Additionally, some new functionality would also be introduced. This product, known as SAS/Warehouse Administrator, enables a site Data Warehouse Administrator (or DWA) or consultant to:

- Define data warehouses and warehouse business subjects through an easy-to-use user interface. Warehouse elements consist of a flexible hierarchy of data and non-data types, including detail tables, summary tables, data marts and information marts.
- Define the operational data sources that feed a warehouse.
- Define the processes by which operational data is transformed, loaded and summarized into the various warehouse elements. Supply tools to assist in the transformation of data.
- Store a variety of metadata for all warehouse elements and processes. Provide various tools to enable the browsing, updating, searching and exporting of warehouse metadata.
- Generate SAS software code to build or load the warehouse elements. Allow substitution or insertion of any user-written code in the process.
- Build distributed warehouses.
- Store warehouse elements in non-SAS formats, for example, DBMS formats.
- Provide job scheduling capabilities.

As the previous list indicates, the central focus of the product was to provide a flexible, customizable architecture to aid the DWA in the implementation phase of a warehouse project.

The production release of SAS/Warehouse Administrator is scheduled for the first quarter of 1997.

REQUIREMENTS

The software runs on any SAS 6.12 platform, typically a PC running Windows, and requires Base SAS software and SAS/FSP® software. The generated code from SAS/Warehouse Administrator can run on any SAS 6.08 platform or above. Additional SAS products may be required depending on what warehouse facilities are used. For example, SAS/Connect® software would be required for remote processing and SAS/Access® software would be required for accessing or loading data stored in DBMS tables.

OVERVIEW

The DWA will begin using SAS/Warehouse Administrator once the following prerequisite steps have been completed:

- The warehouse project has been justified to management.
- The warehouse project team has been assembled.
- End users have been interviewed and business requirements defined. IT requirements have been identified.
- The logical and physical data models for the warehouse subject(s) have been designed.

Once the logical and physical data models have been designed, SAS/Warehouse Administrator software may be used to graphically prototype those designs to see if that is the desired warehouse model. This can be done quickly, without filling in other information that is required to actually build and load the warehouse elements.

It's important to note that typically the DWA would not try to create the entire warehouse at one time. The process of designing, implementing and review is an iterative one, i.e., smaller, incremental projects are much more manageable and produce immediate results. A good rule of thumb is to implement, at most, one subject area at a time.

Assuming you are the DWA, once you have the models and are ready to build the warehouse, the following basic steps would be followed:

1. Invoke SAS/Warehouse Administrator.
2. Define the metadata repositories, i.e., the warehouse environment and any data warehouses. This is done at entry to the product and in the Explorer window.
3. Define the input, i.e., the operational data sources, to the warehouse. This is done in the Explorer window via properties windows.
4. Define the various elements that compose the warehouse. This could include detail tables, summary tables, data marts, and information marts. This is also done in the Explorer window via properties windows.
5. Define the processes that populate those warehouse elements. This is done in the Process Editor.
6. Generate the code to load one or more warehouse elements. This can be done in either the Process Editor or Explorer window. SAS/Warehouse Administrator generates the appropriate code based on the information, i.e., metadata, you entered in the Properties windows within the Explorer or Process Editor.
7. Either:
   a) run the generated code immediately, or
   b) save the code and run it later, or
   c) pass the generated code to the Job Scheduler and run it at some future date and time as a separate executable job.

Obviously, once the code has been executed, logs, tables, etc., should be checked to ensure the warehouse elements have been loaded successfully.

The following sections explain each step in more detail.

INVOKING SAS/WAREHOUSE ADMINISTRATOR

To start SAS/Warehouse Administrator, invoke the SAS System in display manager mode. Then, on the command line, type either

DW

or

DW FOLDER=*library.catalog.entry.* FOLDER

The **DW** command invokes a new facility in SAS 6.12 software -- the SAS Desktop. The SAS Desktop allows the end user to store folders, libraries, catalogs and applications in an easy-to-use, object-oriented interface. However, the primary use of the SAS Desktop with SAS/Warehouse Administrator software is to facilitate the creation and organization of Data Warehouse Environments and related applications in a desktop folder.

If the **FOLDER**= parameter is not specified, SAS/Warehouse Administrator uses the default folder SASUSER.FOLDER.SAS_WA.FOLDER. This folder will be created automatically if it does not already exist. An example of the SAS Desktop is shown in Figure 1:

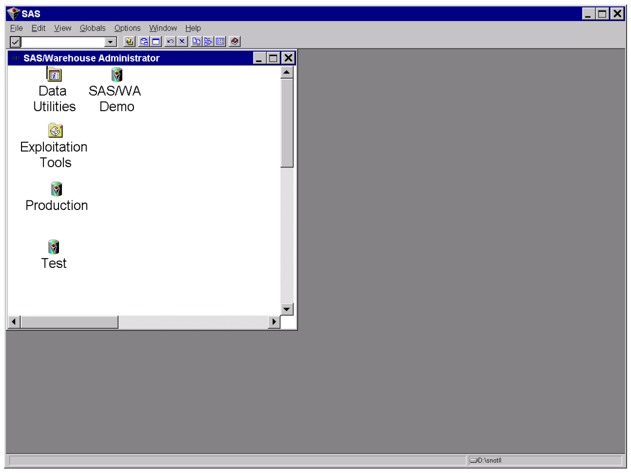
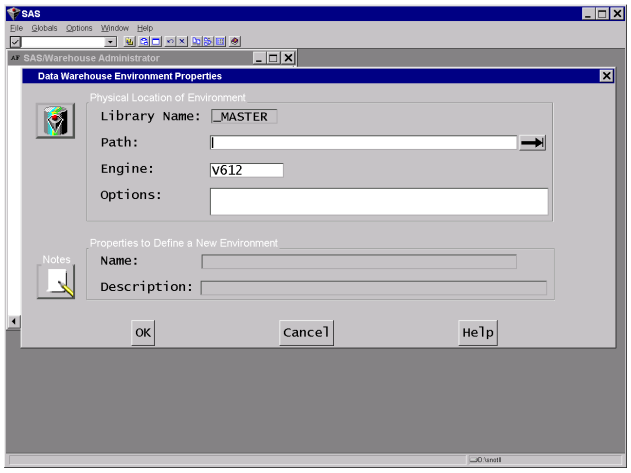

SAS
File Edit View Globals Options Window Help
SAS/Warehouse Administrator
Data Utilities
SAS/WA Demo
Exploitation Tools
Production
Test

Figure 1 - The SAS Desktop

Figure 1 displays SAS Desktop that already contains three Data Warehouse Environments and two application folders.

DEFINING THE WAREHOUSE ENVIRONMENT

When you enter SAS/Warehouse Administrator software the first time, no Data Warehouse Environments will exist. To create one, either open the `File` pull-down menu or open the pop-up menu with the right mouse button, select `Add Item >`, then select `Data Warehouse Environment`. SAS/Warehouse Administrator opens the Properties window for Data Warehouse Environment as shown in Figure 2:

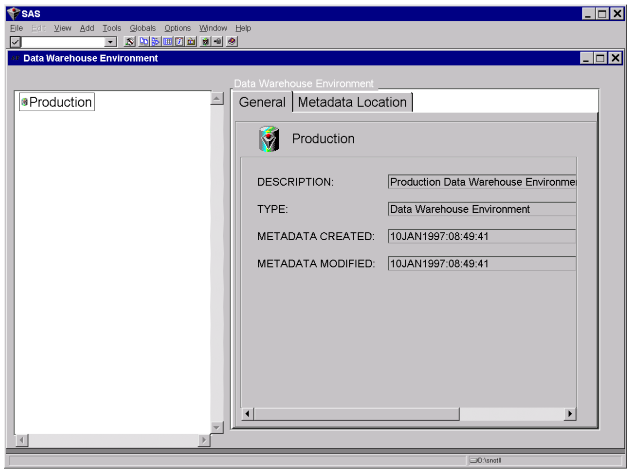

SAS
File Globals Options Window Help
SAS/Warehouse Administrator
Data Warehouse Environment Properties
Physical Location of Environment
Properties to Define a New Environment
Notes
OK
Cancel
Help

Library Name: MASTER
Path:
Engine: V612
Options:
Name:
Description:

Figure 2 - Warehouse Environment Properties Window

The information you enter on the Data Warehouse Environment Properties window is used to generate a LIBNAME statement for the environment. That LIBNAME path will serve as a repository for numerous SAS data sets that will store global metadata information, i.e., metadata that is not specific to a single warehouse. By including server parameters in the `Options:` field, the metadata may also be shared via SAS/Share® software.

Once you have defined your Data Warehouse Environment, you will want to position the cursor over the Data Warehouse Environment icon on the SAS Desktop, open the pop-up menu and select `Edit.` This will open the SAS/Warehouse Administrator Explorer window as shown in figure 3:


SAS
File Edit View Add Tools Globals Options Window Help
Data Warehouse Environment
Production
Data Warehouse Environment
General
Metadata Location
Production

DESCRIPTION: Production Data Warehouse Environme
TYPE: Data Warehouse Environment
METADATA CREATED: 10JAN1997:08:49:41
METADATA MODIFIED: 10JAN1997:08:49:41

Figure 3 - Explorer Window

The SAS/Warehouse Administrator Explorer window contains two panes -- the left side is an organizational chart that depicts the hierarchy of the Data Warehouse Environment. The right side displays information on the currently selected warehouse element. The information displayed on the right side depends on the view that is currently selected. The `View` pull-down menu allows you to select from five different views, `Large Icons`, `Small Icons`, `Details`, `Metadata-General` and `Metadata Details`. These different views enable you to easily see as much or as little metadata as desired. SAS/Warehouse Administrator remembers the view that you used last and will return to that view automatically in your next session.

The first task you will want to do in the Explorer window is to define a Data Warehouse. The Data Warehouse is similar to the Data Warehouse Environment in that it is simply a representation of a location to store metadata. However, where the Data Warehouse Environment is used to store global metadata, the Data Warehouse is used to store metadata specific to a single warehouse. To define a Data Warehouse within an environment, open the `Add` pull-down menu and select `Data Warehouse.` SAS/Warehouse Administrator will immediately open the Data Warehouse Properties window. This window contains two tabs: General and Metadata Location, that allow you to enter general metadata like name, description, owner and administrator as well as the location information where the metadata will be stored. Figure 4 shows the General tab on the Data Warehouse Properties window:

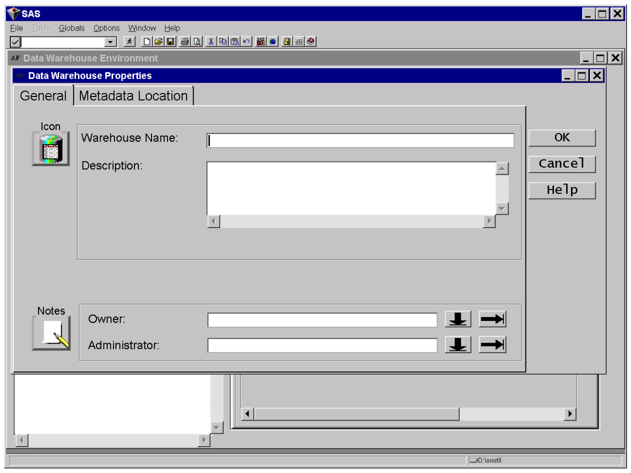


Figure 4 - The General Tab

In addition to entering information in the text entry fields, you can associate free-form Notes with the Data Warehouse by clicking on the `Notes` button and filling in the subsequent Notes window. Additionally, you can change the icon associated with the Data Warehouse by clicking the `Icon` button.

The General Tab, as well as many other tabs, is used in the Properties windows of several warehouse elements. This facilitates a consistent look and feel as you continue to define more warehouse elements.

Note that you can define multiple warehouses per Data Warehouse Environment. This gives you a great deal of flexibility when designing the hierarchy of information in the warehouse. You may decide that storing all information in a single warehouse may be unwieldy for larger projects. Performance may also be enhanced with separate warehouses, since the metadata data sets will be smaller.

DEFINING INPUT DATA SOURCES

After you have defined the metadata stores, i.e., the Data Warehouse Environment and the Data Warehouse, you will want to define the operational data sources that supply input data to one or more of your warehouse elements.

Operational data sources are represented by defining an Operational Data Definition or ODD. An ODD is typically either a SAS table or something that looks like a SAS table, i.e., a view. To access operational data stored in flat files, you define a SAS Data step view that functions as the SAS/Warehouse Administrator ODD. To access operational data stored in DBMS tables, you define either a SAS/ACCESS® view or PROC SQL view that functions as the ODD.

ODDs are logically grouped into ODD groups. You may have as many ODDs or ODD groups as you like. ODD groups are defined similarly to Data Warehouses, i.e., in the Explorer window, open the `Add` pull-down menu and select `Operational Data Group`. Note that unlike the Data Warehouse object, the new ODD group is added immediately to the hierarchy with a default name. To change the name, select `Properties…` off the pull-down or right mouse button pop-up menu. Optionally, you can use the `Rename…` function off the right mouse button pop-up menu.

To define an ODD within an ODD group, open the `Add` pull-down menu (or via the right mouse button pop-up menu) and select `Operational Data Definition`. Once the icon shows up in the hierarchy, select `Properties…` to invoke the ODD Properties window. The ODD Properties window contains three tabs: General, Data Location, and Columns.

The General tab has been previously discussed. The Data Location tab prompts you for the location of the ODD via the host, SAS library, and SAS table/view fields. Note that this tab requests information on the location of the ODD itself, which, in many cases, is different from the location of the actual operational table. For example, a DBMS table you are accessing as an operational data source may reside on MVS, however, a PROC SQL view that functions as the ODD of that DBMS table may reside on your local PC if you used the remote capabilities of the Query Window to define the view.

Note that if you do not have an ODD defined yet, you may invoke a number of tools to define the ODD from the `Tools…` pull-down menu as shown in Figure 5:

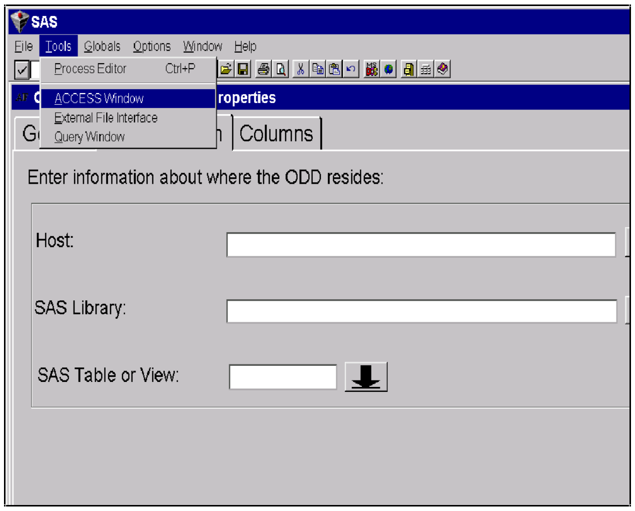


Figure 5 - The Tools Menu

The Access Window, External File Interface, and Query Window may all be used to define views on to Operational data sources that may then serve as ODDs within SAS/Warehouse Administrator.

Location context is an important thing to remember when completing the HOST field on the Data Location tab. Normally, the HOST for an ODD is local or remote based on the context of the job that runs that accesses the ODD. Note that the platform the job executes on may not be the same location as the platform that is executing SAS/Warehouse Administrator.

The Columns tab is used to define the columns in the ODD to SAS/Warehouse Administrator. The column information, such as name, type, length, format, informat and description, may be typed in and added singly for each column, or you may use one of several methods available from the `Import` button or pull-down menu selection. You can import column metadata from the data location you supplied on the Data Location tab or from a variety of other sources:

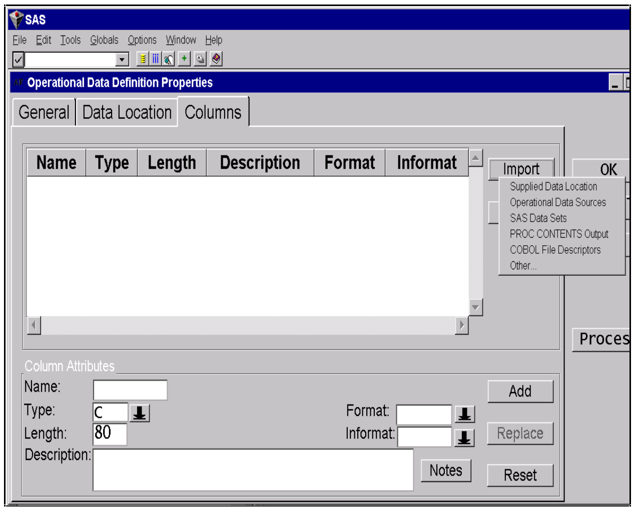


Figure 6 - The Columns Tab

This facility allows you to easily define column metadata without spending a lot of type typing in column information. Column metadata may also be imported from other ODDs, from any allocated SAS data set, from the output of a PROC CONTENTS OUT= operation, or from COBOL File Definitions via the COB2SAS program.

As with the General tab, the Data Location tab and the Columns tab are used in the Properties windows of other warehouse elements.

DEFINING WAREHOUSE ELEMENTS

ODD groups and ODDs are components of a Data Warehouse Environment, but not components of a Data Warehouse. This was by design so that ODDs could be shared between multiple warehouses without having to define them more than once. However, there are many different types of elements within a Data Warehouse. The two primary warehouse elements are Subjects and Data Marts.

Subjects are the business subjects of your business enterprise. Examples of business subjects are “Customer”, “Sales”, and “Purchases”. These subjects may then be composed of a number of different data collections that may reside as SAS data sets, database tables, Multi-Dimensional Data Bases, charts, reports, or graphs. Figure 7 shows the logical structure within a subject:

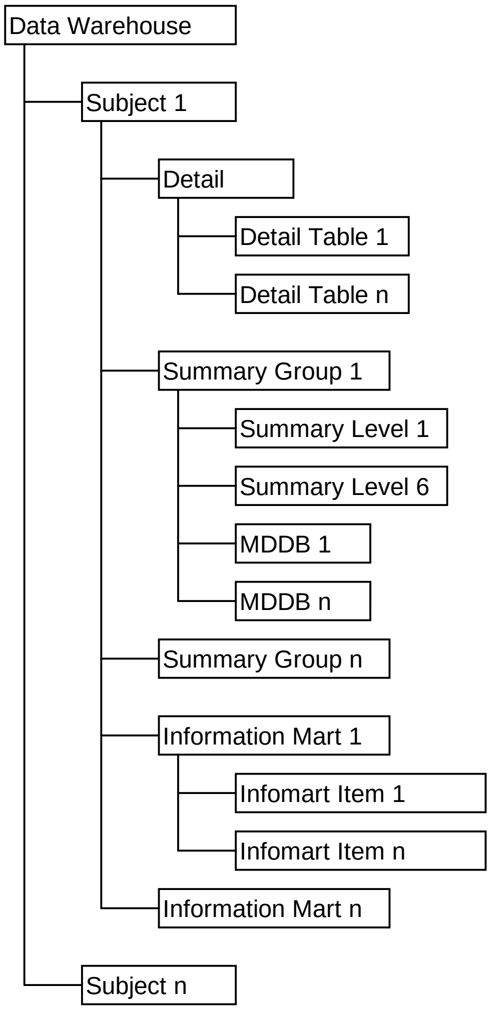


Figure 7 - Structure within a Subject

The detail element can function simply as a grouping element like an ODD group, or it can actually be defined as a table or view on to multiple, related detail tables. You may only have one detail element per subject.

Detail tables are the most granular tables in a subject and are the elements that contain the initial load of data that comes from the operational data sources (after it has been transformed).

Summary groups are groups of related summary tables and/or MDDBs and are derived from their related detail tables. You may have any number of summary groups. A summary group defines the default class and analysis variables that are used when building the dimensions for each summary level within the group.

Summary levels (also known as summary tables) are time-based tables and are summarized by the default variables identified in the summary group. Each summary level corresponds to a particular time dimension such as day, week, month, year, etc. Six different time periods are available for summarization.

Information marts, or infomarts, are logical groupings of information mart items. An information mart item contains or displays information derived from data in a detail or summary level. Typically, they are charts, reports or graphs stored in SAS catalog entries. However, they also may be stored queries, text files, or even other executable applications that are invoked when opened.

Figure 8 shows the logical structure within Data Mart groups:

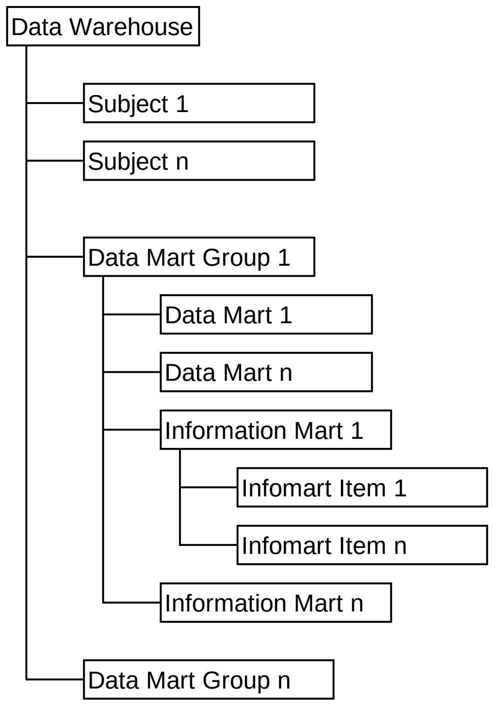
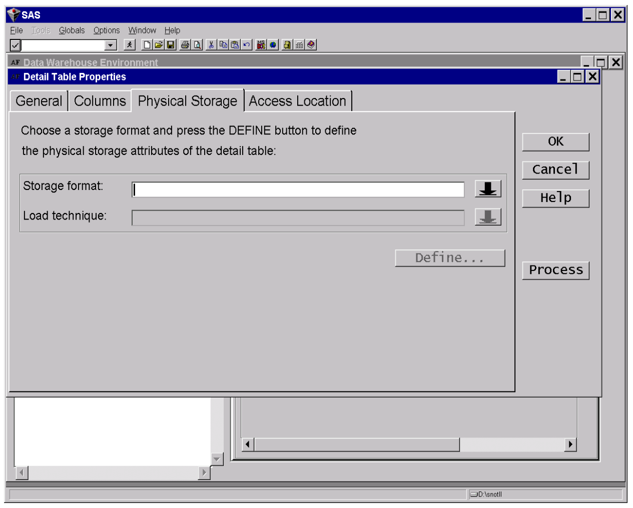


Figure 8 - Structure within a Data Mart Group

Data Marts are tables of data that are typically subsets of one or more subject tables and are designed to address the information requirements of a specific end-user community, such as a department or an individual. Data marts may also be used to store the results of ad hoc queries or cross-subject analyses and usually reside closer to the end user, for example, a Data Warehouse may reside on an enterprise-wide Unix Server, however, a data mart may reside on the end user's desktop PC.

Defining warehouse elements is a similar process to defining the ODD groups and ODDs. You use the pull-down menus or the right mouse button in the Explorer window to add warehouse elements to the hierarchy. You then edit those elements via their Properties windows. Many of the Property window tabs are similar, if not identical, to other tabs you have already seen when defining ODDs. However, there are some new tabs that pertain to the definition of the physical storage characteristics of warehouse tables. Figure 9 shows the physical storage tab that is displayed when defining or editing the properties of a detail table:



Choose a storage format and press the DEFINE button to define the physical storage attributes of the detail table:

Storage format:

Load technique:

Figure 9 - Physical Storage Tab

The storage format of a detail (or summary) table may be either SAS or DBMS format. The Load Technique may be either "Refresh" to reload the entire table or "Append" to append new transactions coming from the operational data source to the detail table. Once those two fields have been completed, click on the `Define` button to display the appropriate Table Properties window. Different windows are displayed based on what you selected as the storage format.

If you selected the SAS storage format, a SAS Table Properties window will be displayed that will allow you to define the physical location of the SAS table, any read/write/alter passwords, SAS data set options like compression and encryption, and any desired data set indexes. The index tab is a relatively new addition that allows you to define single or multiple indexes as well as update and delete those indexes:

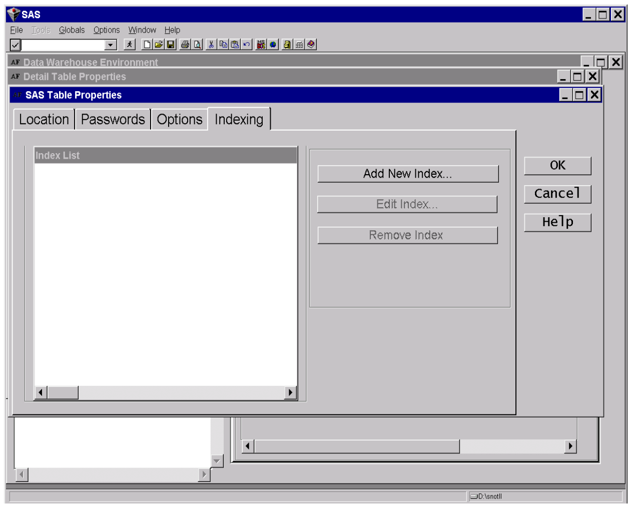


Figure 10 - The Index Tab

If you chose instead to store your detail table in a DBMS storage format, clicking on the `Define` button will display the DBMS Table Properties window as shown in Figure 11:

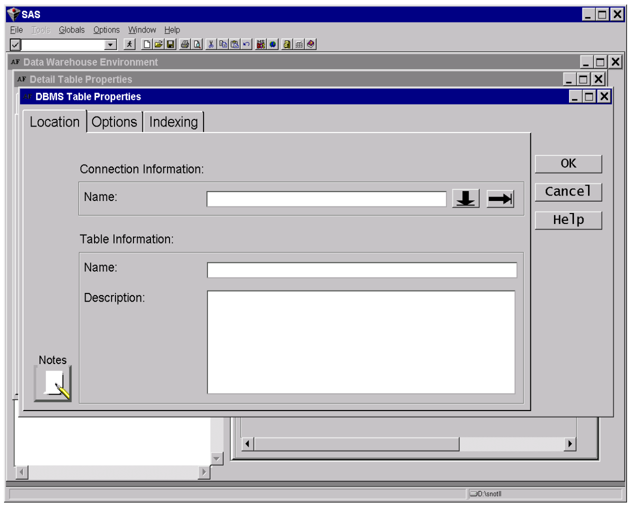


Figure 11 - DBMS Table Properties Window

The Location tab contains important information about how the DBMS table is to be accessed. The table is accessed by defining a DBMS connection object that contains information about the DBMS format, e.g., Oracle, as well as information on DBMS userids, schemas, passwords, and other options as shown in Figure 12:

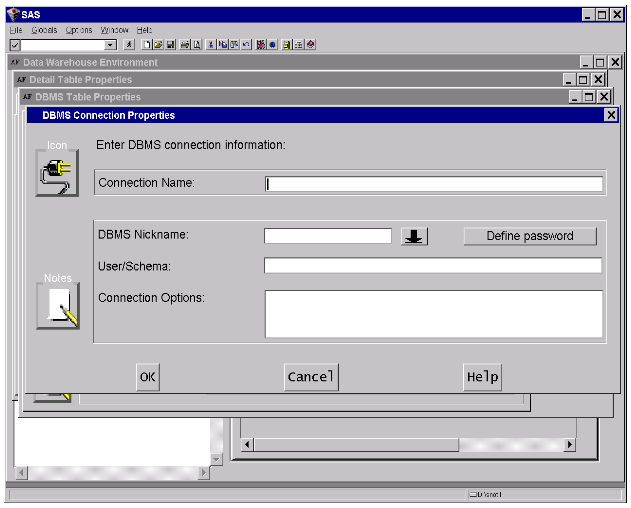


Figure 12 - DBMS Connection Properties Window

This information is used to generate the appropriate SQL to create and load the DBMS table in the load job.

Once you have initially defined your detail and detail tables, it's a good idea to proceed with defining the process that will populate those tables (via the Process Editor), rather than defining more warehouse elements such as summary tables and infomarts. Once you have tested the loading of the detail tables, then other warehouse elements, such as summary tables and data marts, may be defined easily via similar techniques.

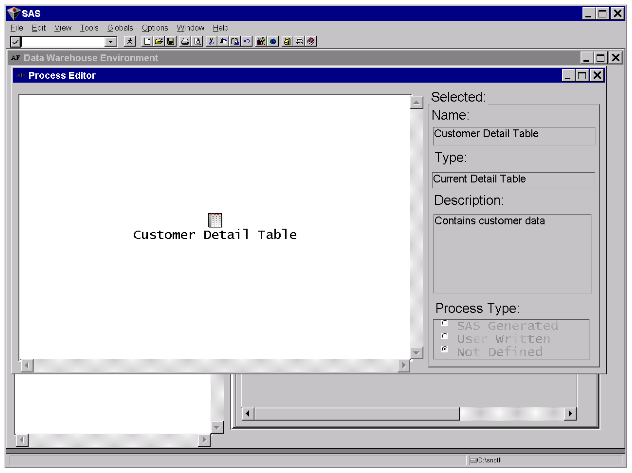
DEFINING WAREHOUSE PROCESSES

Once the desired warehouse elements have been defined with the Explorer and properties windows, you are now ready to determine how those elements are to be populated. This is done by using a new tool known as the Process Editor. The Process Editor is accessible from many different windows by way of pull-down menus, pop-up menus, the toolbox, and push buttons, all via the `Process…` or `Process Editor` selection.

The Process Editor enables you to define how data flows from your Operational data sources, through any optional transformation steps, and into the load step for the target table. Additionally, the Process Editor defines how other warehouse elements, such as summary tables or data marts, are populated.

The easiest way to explain how the Process Editor works is to go through a simple example of defining a load process for a detail table.

In our example, assume you have defined a detail table called CUSTOMER DETAIL TABLE that contains information about customers. Invoking the Process Editor on the Customer detail table shows only a single icon, since no process has yet been defined:

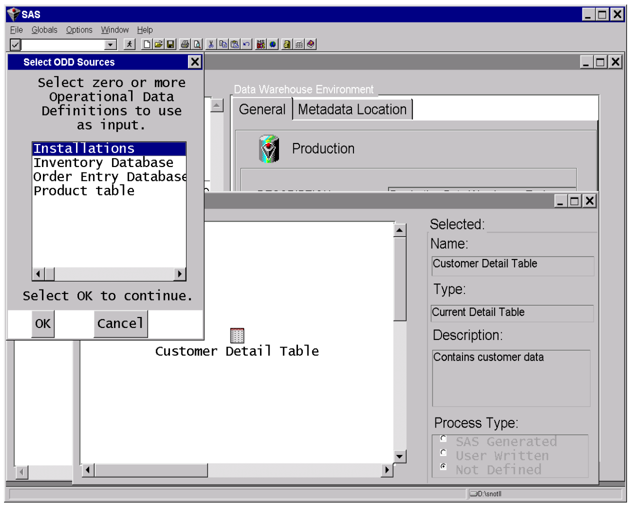


Name: Customer Detail Table
Type: Current Detail Table
Description: Contains customer data

Figure 13 - Process for New Detail Table

Note that the Process Editor displays the name, type of warehouse element, and description that you completed on the properties window for that detail table. Additionally, the Process Type radio box indicates whether or not a process step has been defined for that object, and, if so, whether or not the code will be generated by SAS/Warehouse Administrator or supplied by you.

The first thing you will want to do is to decide what ODDs are input to that Customer Detail table. To do that, you define an Operational Data Mapping for the table. The Operational Data Mapping defines how columns in the ODD are mapped to columns in the detail table. To add a Mapping, select the Customer Detail table and open the pop-up menu with the right mouse button and select `Add>Operational Data Mapping`. SAS/Warehouse Administrator displays a small selection box that requires you to select which ODDs are input to the Customer table. In our example, we select the INSTALLATIONS ODD:



Select zero or more Operational Data Definitions to use as input.

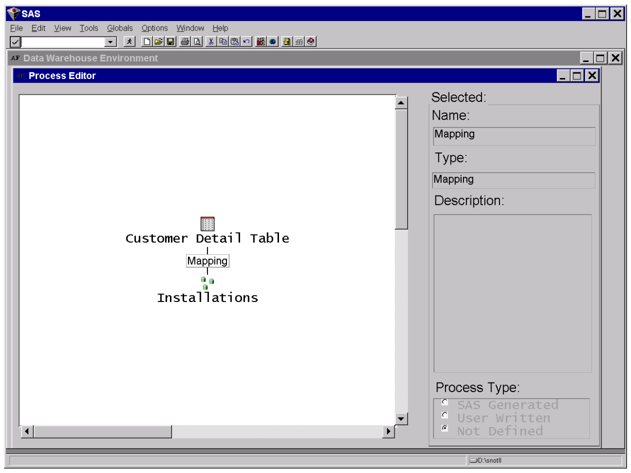
Installations
Inventory Database
Order Entry Database
Product table

Select OK to continue.

Selected:
Name: Customer Detail Table
Type: Current Detail Table
Description: Contains customer data
Process Type: SAS Generated / User Written / Not Defined

Figure 14 - Selecting Operational Sources

After you choose your input ODDs, you will notice that the process diagram has been updated as shown in Figure 15:


SAS
File Edit View Tools Globals Options Window Help
Data Warehouse Environment
Process Editor
Customer Detail Table
Mapping
Installations

Selected:
Name: Mapping
Type: Mapping
Description:
Process Type: SAS Generated / User Written / Not Defined

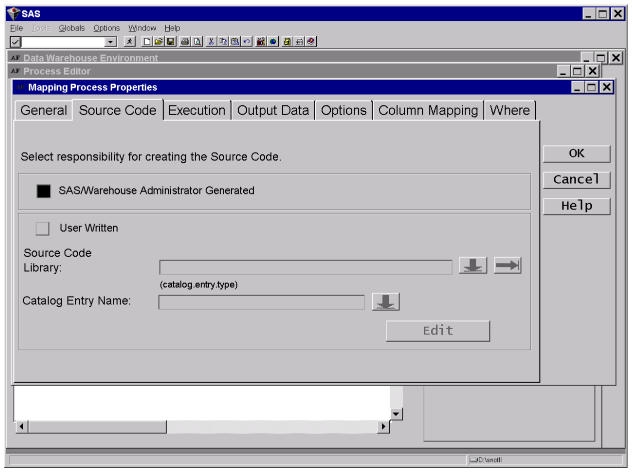
Figure 15 - Updated Process Diagram

You read the process diagram from bottom to top, i.e., the sources are at the bottom and the data flow proceeds upward towards the target table at the top.

Note that the ODD and the detail table are depicted as icons where the Mapping object is depicted as a simple text box. Iconed objects denote “loadable” entries such as tables. They represent both data (the table itself) and the code used to load that table. The pop-up menu for iconed objects contain an “Edit Load Step” selection that is used to specify how the load code for that table is to be generated.

Text box objects, like the mapping object, are “process” entries that primarily represent source code only, i.e., there are no pre-defined warehouse objects (like detail tables or ODDs) associated with process objects.

To define the process for the Mapping object, we select the object and open the pop-up menu with the right mouse button and select `Properties`. The following Mapping Process Properties window is displayed:



Select responsibility for creating the Source Code.

■ SAS/Warehouse Administrator Generated

☐ User Written

Source Code Library:
(catalog.entry.type)
Catalog Entry Name:

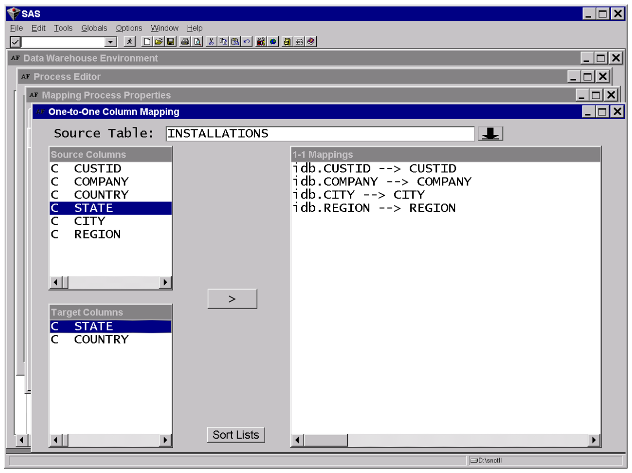
Figure 16 - Mapping Process Properties Window

The Source Code tab, as shown in Figure 16, is where you specify whether or not you want SAS/Warehouse Administrator to generate the code to do the mapping of columns, or if you will be supplying the code via a SAS catalog source entry.

The Execution tab specifies the host where the mapping source code should run. If this is a remote host, then the source code is automatically enclosed in an RSUBMIT block.

The Output Data tab specifies the output data set that will be produced as a result of the mapping process.

One of the most important tabs on this window is the Column Mapping tab. This tab allows you to define how the detail table columns are to be mapped to the ODD columns. You can specify that you want the detail table columns to be mapped directly, one-to-one, with the ODD columns, i.e., column values are directly copied without transformation. To do this, you click the button labeled `1 to 1 Mappings…` This will display the One-to-One Column Mapping window that will allow you to choose which columns map to each other:


SAS
File Edit Tools Globals Options Window Help
Data Warehouse Environment
Process Editor
Mapping Process Properties
One-to-One Column Mapping
>
Sort Lists

Source Table: INSTALLATIONS

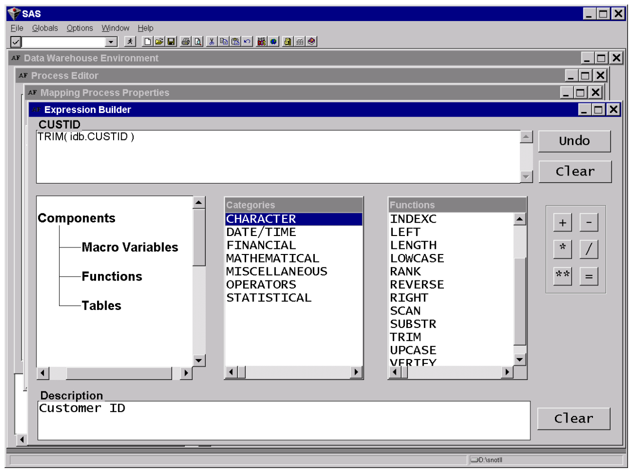
Source Columns
```
C  CUSTID
C  COMPANY
C  COUNTRY
C  STATE
C  CITY
C  REGION
```

Target Columns
```
C  STATE
C  COUNTRY
```

1-1 Mappings
```
idb.CUSTID --> CUSTID
idb.COMPANY --> COMPANY
idb.CITY --> CITY
idb.REGION --> REGION
```

Figure 17 - One-to-One Mapping Window

Alternatively, you can specify that detail table columns can be derived from ODD columns by some formula or other type of translation. By pressing the `Derive Mapping` button, SAS/Warehouse Administrator displays an Expression Builder window that allows you to construct many different types of transformations on column values as they are being extracted from the ODD:



CUSTID
TRIM(idb.CUSTID)

Categories	Functions
CHARACTER	INDEXC
DATE/TIME	LEFT
FINANCIAL	LENGTH
MATHEMATICAL	LOWCASE
MISCELLANEOUS	RANK
OPERATORS	REVERSE
STATISTICAL	RIGHT
	SCAN
	SUBSTR
	TRIM
	UPCASE
	VERIFY

Description
Customer ID

Figure 18 - Expression Builder Window

Once you have completed the information on those four tabs, your mapping is complete and may be saved by pressing OK. You may optionally set some various options via the Options tab, or supply a WHERE clause for the generated mapping code via the WHERE tab. The General tab is used to change the name of the Mapping object as it appears in the Process Editor.

There are currently three other types of process objects besides Operational Data Mappings:

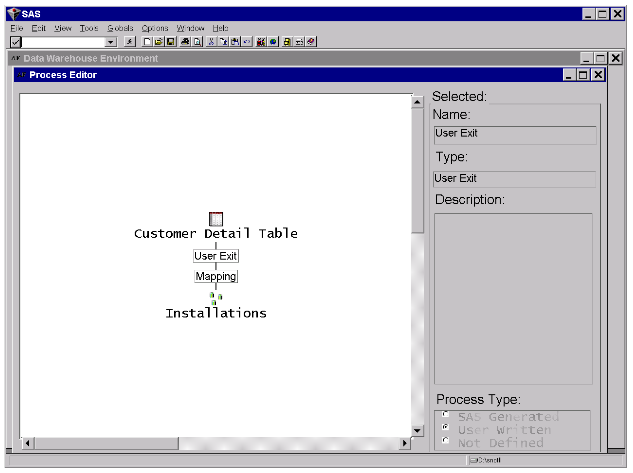
- Data Transfers
- Record Selectors
- User Exits

Data Transfer process objects generate code that is used to transfer data from one host to another. SAS/Warehouse Administrator generates code that uses PROC UPLOAD or PROC DOWNLOAD to move the data.

Record Selector process objects are used to subset data according to the criteria you specify. Record selectors are frequently used to extract just the changed data from any input ODDs.

User exit process objects are code fragments that you want SAS/Warehouse Administrator to insert into the process and run. They are frequently used to transform data before loading into the target table, or to generate reports.

For example, perhaps you wanted to include a user exit between the Mapping object and the load step that would print a PROC CONTENTS of the extracted data from the Mapping step. You would select the Mapping object, click the right mouse button to open the pop-up menu and select `Insert>User Exit`. A User Exit process object would be inserted in the process flow as shown in Figure 19:



Name:
User Exit
Type:
User Exit
Description:

Process Type:
SAS Generated
[✓ User Written]
Not Defined

Figure 19 - Process after adding a User Exit

You would then invoke the Properties window on the new User Exit and complete the Source Code tab so that it points to your PROC CONTENTS source code in a SAS catalog SOURCE entry.

Note that we used the `Insert>` selection in the previous example, rather than `Add>`. `Insert>` means to add a "target" to the currently selected entry, i.e., closer to the top of the process flow diagram. `Add>` means to add a "source" to the current entry, i.e., closer to the bottom of the diagram.

The last thing you will want to do is to define how the detail table is to be loaded. Up until now, we have only defined how the input data are accessed and prepared, i.e., transformed into the correct format in preparation for loading into the target table. To define the details of the Load step, select the detail table and open the pop-up menu with the right mouse button and select `Edit Load Step…` This will invoke the Load Process Attributes window that contains some tabs that you saw before on the Mapping Properties window. You can specify how the source code for the load step is to be generated, where the code should run, what options are in effect, and if any post-processing step should run after the load step. The post-processing step can be used to perform statistics on the loaded table, send completion messages or reports to the DWA, or for any other purpose.

Since the processes for all objects in our example have been defined, you are now ready to generate code to load the Customers detail table.

There are many other facilities in the Process Editor that are beyond the scope of this paper. The Process Editor is an incredibly flexible tool that can be used to define practically any kind of warehouse process. It also serves as an effective documentation tool that describes how warehouse tables and columns are populated. At SAS Institute, we have even used the Process Editor to document data flows and source-target relationships for processes that are not even actual warehouse applications.

GENERATING CODE

Once you have completed the definition of the process metadata for the warehouse element you are working on, you may start generating the code that will be used to load that element. To view the code that will be generated, go to the Process Editor for the warehouse element you are loading and open the pop-up menu for that selected element with the right mouse button. Select the `View`

`Code…` function to view the generated code. You have the option of viewing code for the selected element or for the entire process tree.

It is recommended that you view the generated code for many of the process editor objects, such as ODDs, Data Transfer objects, Mapping objects, etc. to get a sense of what those objects do and how they can be used in the process diagram. For example, if an ODD has been defined as remote, an RSUBMIT block is automatically generated by SAS/Warehouse Administrator:

```
        options comamid=tcp;
        filename rlink "tcptso.scr";
        %let MVS=mvs.mycompany.com;
        signon MVS;
        rsubmit MVS;
        libname mvsdata
        "mvsdata.test.sasdata"
        ;
        endrsubmit;
```

SAS/Warehouse Administrator generates the code in the Process Editor from bottom to top, left to right. Whenever you generate code to load a table, SAS/Warehouse Administrator generates the code needed to access any warehouse elements that are input to the table, as well as the code to load the table itself. To be more specific, when `View Code` is selected for a "loadable" element, such as a detail or summary table, the code that is generated contains all of the processing down to the next "loadable" element in the process diagram. When another "loadable" element is encountered, only the code needed to access that element is generated, i.e., it's associated Load Step code is not generated. For example, if you were generating code to load a summary table, SAS/Warehouse Administrator would generate the code to access the detail tables that are input to the summary table, but it would not generate code to load those detail tables.

User Exits are inserted in the generated code without modification.

Each individual step in the process, by default, sets the &SYSLAST macro variable to communicate the last data set created to the next step in the process. For example, this code fragment below was generated for a mapping object and detail table. It shows how the WORK.EXTPROD view created in the mapping step is accessed in the next step by the SET _LAST_ statement:

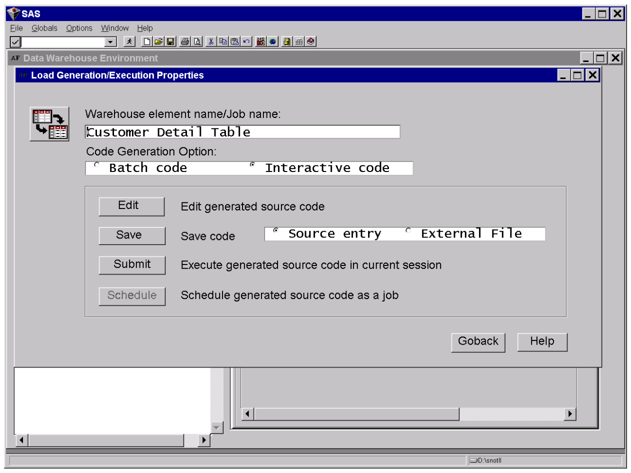
```
 /**********************************************/
/* Name: mapping */
/* Description: Execute the Process for this
step */
/* Generated:  08JAN97:10:02:58 */
/**********************************************/
PROC SQL;
CREATE VIEW WORK.extprod AS
SELECT
pdb.PRODNUM AS PRODNUM length=8
,
pdb.PRODNAME AS PRODNAME length=15
,
pdb.PRODID AS PRODID length=9
FROM
PDB.pdb
;
QUIT;
%let syslast=WORK.extprod;

/**********************************************/
/* Name: Product detail table */
/* Description: Execute the Process for this
step */
/* Generated:  08JAN97:10:02:59 */
/**********************************************/
libname _whdata
"dwdemo\_whdata"
;
DATA WORK.A00000W8 / VIEW=WORK.A00000W8;
SET _LAST_;
       length _loadtm 8;
       _loadtm=input("&SYSDATE"||'
'||"&SYSTIME",DATETIME.);
RUN;
```

There is an option to remove the definition of the &SYSLAST macro variable, however, it is then up to you to ensure that the flow of data through the process is correct.

EXECUTING GENERATED CODE

Once you have reviewed the code for the warehouse element you wish to load, select that element (either in the Explorer window or Process Editor window) and select `Run…` from the pop-up menu, `File` pull-down menu or toolbox. SAS/Warehouse Administrator opens the Load Generation/Execution Properties window:


SAS
File Globals Options Window Help
Data Warehouse Environment
Load Generation/Execution Properties

Warehouse element name/Job name: Customer Detail Table

Code Generation Option: Batch code [✓ Interactive code]

Edit — Edit generated source code

Save — Save code [✓ Source entry] External File

Submit — Execute generated source code in current session

Schedule — Schedule generated source code as a job

Figure 20 - Load Generation/Execution Properties Window

This window gives you the option of generating code that is tailored to be run in either batch or interactive mode. Interactive mode assumes you will be running the generated code in the current session, so does not allocate the metadata data sets. Interactive mode is useful for testing the code, however is not practical if you are working with large amounts of data. If you select batch mode, SAS/Warehouse Administrator generates the LIBNAME statements to access metadata.

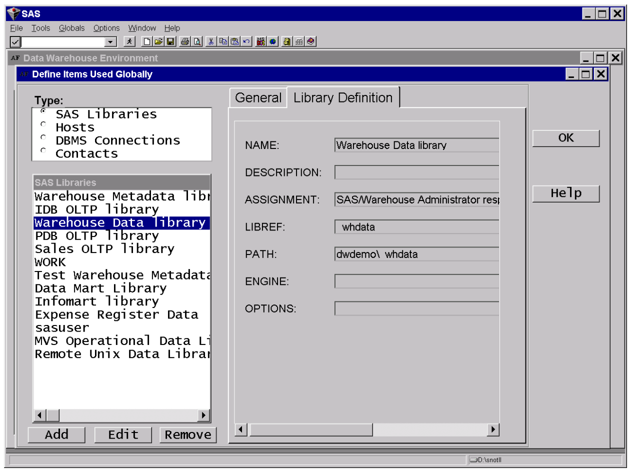
Additionally, you have the option of either editing the job, saving the job source code to a catalog source entry or external file, or passing the generated source code to the Job Scheduler.

ADDITIONAL TOOLS

SAS/Warehouse Administrator also offers other new tools and interfaces that assist you in building warehouses.

Setup Window

The Setup Window allows you to define, in one interface, any items that are used globally. This includes SAS Libraries, Hosts, DBMS Connections, and Contacts:



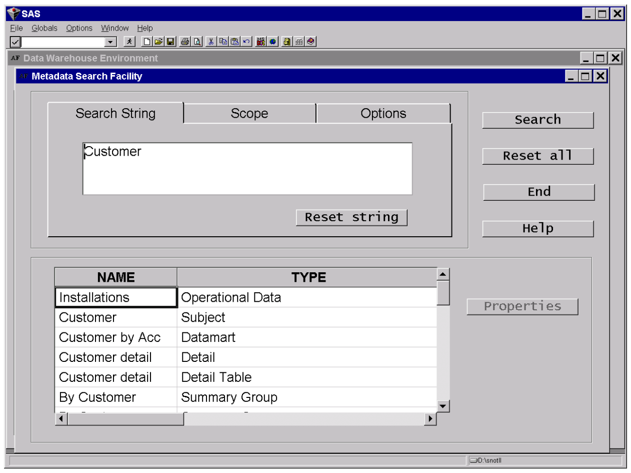
NAME: Warehouse Data library
DESCRIPTION:
ASSIGNMENT: SAS/Warehouse Administrator res
LIBREF: whdata
PATH: dwdemo\ whdata
ENGINE:
OPTIONS:

Figure 21 - The Setup Window

Items may be either added, edited or removed. This window is useful if you have a large number of global items to enter at once.

Metadata Search Capabilities

Once you have a large warehouse already built, you may find it necessary to locate a particular item of information. The Metadata Search window, shown in Figure 22, makes this easy to perform:



Customer

NAME	TYPE
Installations	Operational Data
Customer	Subject
Customer by Acc	Datamart
Customer detail	Detail
Customer detail	Detail Table
By Customer	Summary Group

Figure 22 - The Metadata Search Window

You access the Metadata Search window by opening the `Tools` pull-down menu and selecting `Search Metadata…` This window provides the ability to search for a character string in the NAME and/or DESCRIPTION fields of any warehouse element. Various options are available to either limit the search or modify the search criteria.

When the list of matching warehouse elements is displayed, you can select individual items and click on the `Properties` button to view the appropriate properties window for that selected item. If the item is a column object, the properties window for the owning warehouse table is displayed.

Metadata Export Capabilities

SAS/Warehouse Administrator provides the ability to export the metadata to either an EIS metabase or SAS data sets. You can then use the exported metadata in other SAS or SAS/EIS® applications.

Miscellaneous

SAS/Warehouse Administrator contains a number of useful data utilities from the Explorer window. To access those utilities, select a warehouse element that represents a table, such as an ODD or detail table. Open the pull-down menu under `Tools` or the pop-up menu via the right mouse button and select `Data Utilities >`. You should see a list of actions as shown in Figure 23:

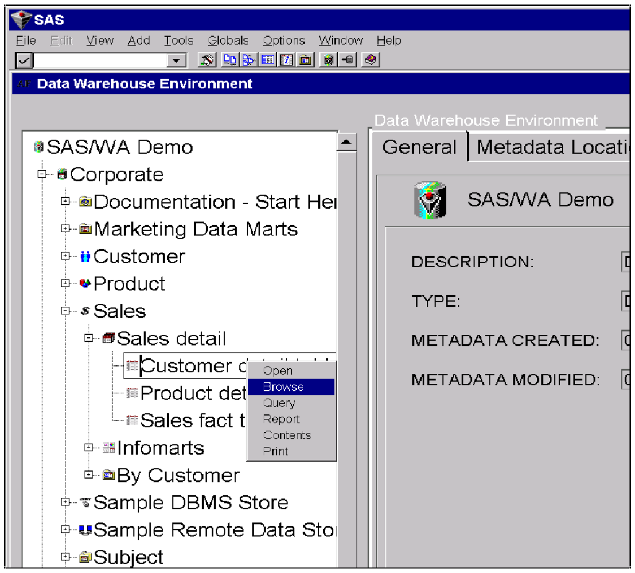


Figure 23 - Data Utilities

For example, selecting `Contents` will issue a PROC CONTENTS against the data set, displaying the complete attributes of the table.

As with many other SAS applications, you can drag and drop tables. For example, you could add a SAS/EIS application on your SAS/Warehouse Administrator folder. You could then drag a table from the Explorer window and drop it on top of that application. This will invoke the SAS/EIS application with the table as its primary input.

CONCLUSION

SAS/Warehouse Administrator software provides a flexible framework for effective warehouse management through a metadata-driven architecture. It provides business subject definition, table and column definition, summarization, process editing and numerous other capabilities that facilitate the building, maintenance, and exploitation of data warehouses. SAS/Warehouse Administrator further strengthens the ability of the SAS System to be used as an effective data warehouse tool.

REFERENCES

Lewis, Terry, SAS Institute Inc. (1996), "*Data Warehousing with the SAS System*", Western Users of SAS Software Proceedings of the Fourth Annual Regional Conference

SAS Institute Inc. (1996), *SAS/Warehouse Administrator User's Guide,* Cary, NC: SAS Institute Inc.

AUTHOR CONTACT

Terry Lewis, SAS Institute Inc., 100 SAS Campus Dr., Cary, NC 27513, (919)677-8000, ext. 7778, email snotll@mvs.sas.com