

The Data Warehouse Manager:

An implementation of some organizational ideas in the SAS system

Paul A. Thompson, Ph.D.
Case Western Reserve University
Cleveland, OH 44106

August 4, 1995

Abstract

The notions of a data warehouse are currently a very hot topic in a variety of areas. The DWM, an implementation of several notions inherent in data warehousing, uses SAS/AF control structures to run data warehouses. It uses an extremely sophisticated macro method for updates of master system datasets, an integrated approach to setting data types, an integrated system for data entry, and overall control using SAS data management tools. The system is self-installing and self-removing. It is platform-independent, as all machine-dependent details are added using macro constructions.

Introduction

The notion of the data warehouse (DW) in the SAS[®] system for data management (SAS) is one in which data is stored in a canonical form (similar in many ways to the basic notion of a SAS dataset) and can be moved out and in from the storage medium with great flexibility.

There are some important ideas which are basic to the DW. In a DW, the master system dataset (MSD) is read-only, thus allowing concurrent access to such basic central datasets. Methods for data access, updating and control should be "transparent to the user," as well as should the methods for moving data to and from the MSD. Data should then be able to be analyzed in other platforms and on other systems (packages, etc.), while storing and abstracting them in the SAS system. While a lot of this process depends on some external programs, there is a certain amount which can and should be able to be managed internally by SAS itself.

The Data Warehouse Manager (DWM) is a system which runs a DW in SAS.

- It allows the data manager to define an overall structure for a set of dataset type(s) (DST). This includes setting up canonical screens, overall audit characteristics (frequency, etc.), storage systems and other details.
- It allows the creation of DST in a uniform manner, in which a whole range of variable characteristics can be defined in an equivalent fashion.
- It performs updates in a consistent fashion, saving a variety of performance statistics to enable the data manager to determine success of the update process.
- It forces data entry to be done in a double-entry/compare algorithm.
- It enables corrections in entered data to be entered and applied simply.
- It enables the data entry personnel to perform audits of data on an ongoing basis in a simple and straightforward manner.

Some terms for the DWM

The DWM uses a few terms, which will be defined here.

- **MSD:** a master dataset. It may be considered to be *read-only*, should be concurrently accessible for all system users. In addition, all MSD are uniquely keyed with certain variables, which in the SAS system can be considered to be BY variables. Data is never entered directly into a MSD, for several reasons. First, this is a somewhat dangerous process, as the act of opening the dataset in a write mode may damage it. In addition, it is very difficult to track data when it is directly entered into the MSD. Finally, it is important to have an unaltered copy of entered data for later use.

- interim dataset(s) (ID) : a dataset in which data is entered directly, using either a full-screen method such as FSEdit, translation from some other format (using something such as DBMS/COPY) or other forms of direct transference. Other methods for data entry into ID include ASCII data entry (definitely not preferable), scanner methods and others which are developing rapidly as the computer revolution proceeds.
- update process (UP) : a method for getting data from a ID into a MSD. It involves 1) determining that data needs to be added, 2) obtaining this data and putting it into a file, 3) transforming of the data if necessary, 4) checking the data for range violations and other problems, 5) checking the MSD to ensure that keys are unique, 6) addition of the data, 7) restoration of the MSD after adding in the new data. In addition, other problems must be considered, such as the need for audits of data for regulatory purposes, corrections of data when errors are found, routine computation of summary measures from the data and the processing of routine, systematic, useful and easy-to-read statistical summaries of the UP.
- DST (see Page 1): the basic conceptual unit of the DWM. It consists of 1) the MSD for the data; 2) the code for summarizing, transforming and otherwise altering initial data into its ready-to-run form; 3) ID used to enter data for addition to the MSD; and 4) other information needed to make sensible use of the data.
- short type name (STN) : a short-hand method for referring to a DST. It is a three-letter acronym for the DST, and is used to uniquely key macros for use in the system.
- numbers dataset (NDS) : the place where all information about the DST is stored. It retains records about current ID numbers for both new and corrections ID, numbers for backup datasets, BY variables, MSD name, merge-in dataset (MID) dataset IDs and other information. It is managed by one module of the DWM
- records dataset (RDS) : the dataset in which all information about each ID is stored. Each ID is keyed by number within the DST context. In addition, the RDS stores the dates of creation, audit, update, and second update.

Core ideas of the DWM

The DWM is based on several core ideas.

1. Storage location for data should be transparent to the user. Users should not need to define them, nor should they need to be responsible for the tracking of such data.
2. The SAS system is excellent at the tracking of sequence numbers, retention of dates and determination of status of datasets. data manager (DM) and other users should not need to remember the status of ID, the current condition of the MSD, the location of backup datasets, and other important information.
3. The SAS macro system is quite sophisticated and can be tailored with great sensitivity for a variety of changes in the process.

Components of the DWM

The DWM system includes the following components.

1. NDS: The NUMBERS dataset contains information which controls processing of each DST (see Page 1).
2. RDS: The RECORDS dataset contains information about each ID.
3. UP: The update process is the method by which new and corrections ID are added to MSD.
4. The DST creation process: DST are created in a unified and simple fashion.
5. The data entry process: data entry is tightly controlled in DWM

Each of these components is briefly discussed below.

NDS: DST registration and MSD tracking

One of the two core datasets for the system is the NDS. This dataset stores all information about all DST (see Page 1). This includes 1) BY variables; 2) MSD name; 3) long or descriptive name of the DST; 4) current sequence numbers for new and corrections datasets; 5) backup dataset number; and 6) if needed, MID dataset number and STN, BY variable and other variables. In addition, editing choices for variable names are defined for the creation of FSEdit screens. When MSD are processed, the NDS is used to provide the controlling information.

RDS: ID registration and tracking

As datasets are created using the DWM, information about them is stored in the RDS. This dataset includes STN, sequence number, number of observations, and dates of creation, audit, update and (occasionally) reupdate. Indicator variables record whether the ID has been audited, updated and range-checked.

The update process

Updates are done in a common update approach, using a very complex macro. The macro has some very interesting details which bear some examining from the standpoint of the understanding of SAS macro coding.

In performing this process, there are several things that must be taken into account:

- The UP (see Page 2) must be done without too much maintainance.
- The process must be general.
- Each UP for an individual MSD is different, but all are somewhat similar.

The update macro has the following features (see the flow-chart for this process in Figure 1):

- All information to tailor the macro is stored in NDS, in terms of the components of the process (like the variable names which are used in the BY statements). It is consulted at the start of the UP (see Page 2). A pass through a DATA _NULL_ step converts the variable information in the NDS into macro variables. This includes 1) MSD name; 2) BY variables for the DST; 3) long or descriptive name of the DST; 4) merge dataset number and STN and related BY variables.
- The UP macro update is written with a number of "hooks" to hang code on for individual DST. Every DST has code to modify the add dataset and code to modify the final MSD. However, several datasets have very specialized need, and so the macro update has seven "hooks" written in.
- Each DST has a separate macro dataset with its version of the "hook" sections listed. Each "hook" code section is a fragment consisting of those statements needed at each point.

The DST creation process

DST are both created and revised in a single general method. First, the STN is chosen. If it exists, old information about the DST is recalled and presented for revision. If it does not exist, new information about the DST is entered. After the DST is defined (by indicating the MSD, BY variables, and other required information), the variables for the dataset are defined. These are defined sequentially, but variables which have identical characteristics are defined at the same time. After variable creation has finished, dataset screens are defined, using an algorithm which allows several types of screens to have common characteristics. Finally, the macro dataset is generated and presented to the DM (see Page 2) for overall editing. At this point, statements are added to summarize, transform, collapse and otherwise alter raw data to the point where it has the proper form for statistical and other data handling in standard SAS procedures and procedures in other languages.

The data entry process.

The DWM supports four types of data entry process.

FSEDITdata entry. The most commonly used approach is the FSEDITmethod. Here, the DWM runs by 1) obtaining a sequence number from the NDS, 2) DWM generates a new ID from a pattern dataset incorporating the sequence number (as well as a compare ID); 3) opening the ID for data entry, 4) opening the compare ID for entry 5) comparing both. After they are the same, 6) a record is entered into RDS and 7) an audit dataset is prepared.

Other methods. Other methods for data entry include ASCII translation, SAS dataset registration and SASPROC CIMPORT dataset translation. In each case, the approach involves:

1. obtaining the location of data;
2. using the data port method to translate the data into SAS datasets;
3. entering information into the RDS dataset to ensure that the data will be updated at the next round of data update.

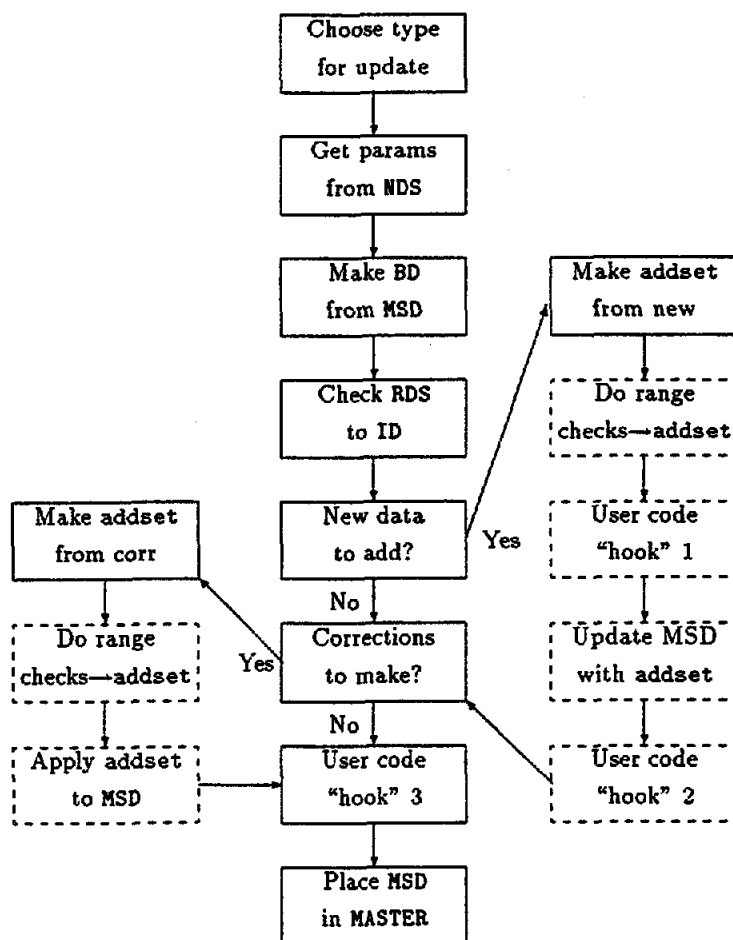


Figure 1: Flowchart of the generic update procedure

Macro techniques used in the DWM

Conversion of numeric values

In using PROC SYMPUT when the value being put into a macro variable is a number, it is sometimes difficult to determine the number of spaces for the conversion into character which is then put into a macro variable. Here are three choices:

```

DATA CHLK;
NV=23;
CALL SYMPUT("_maval", NV); /* 1 */
CALL SYMPUT("_mbval", PUT(NV,5.)); /* 2 */
CALL SYMPUT("_mcval", COMPRESS(PUT(NV,5.))); /* 3 */
RUN;
%put _maval[&_maval.], NAME&_maval.END;
%put _mbval[&_mbval.], NAME&_mbval.END;
%put _mcval[&_mcval.], NAME&_mcval.END;

```

This results in the following:

```

_maval[      23.1], NAME      23.1END
_mbval[   23], NAME   23END
_mcval[23], NAME23END

```

When there are integers only, cases 1, 2 or 3 may be tried. When the integer is to be appended to a string, in macro code, Cases 1 or 2 will have poor results, with the implicit blanks embedded at the front of the string. Note that the

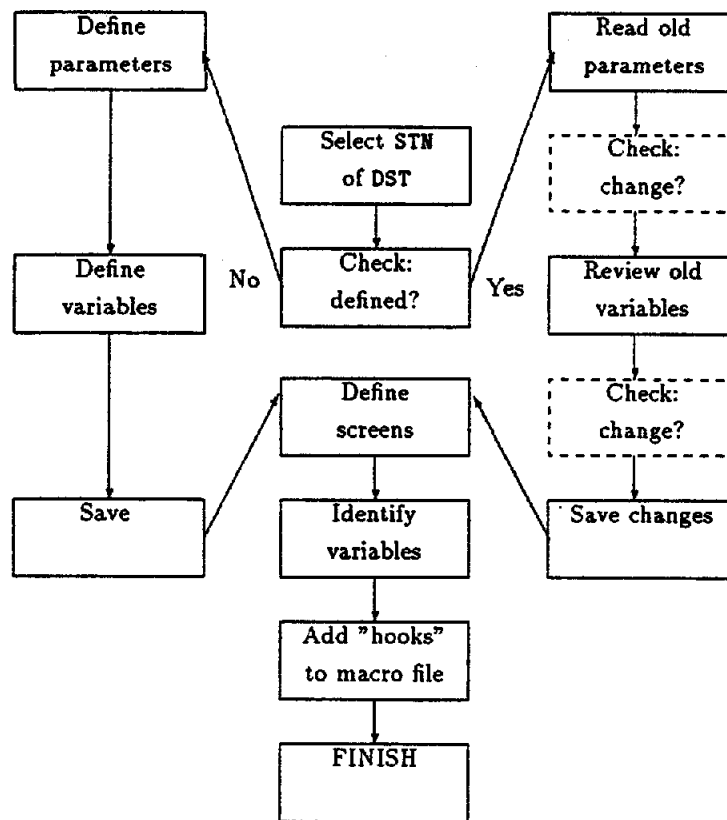


Figure 2: Flowchart of the DST definition process

TRIM function, which only removes blanks at the end of the string, will not be correct in this case. When a decimal is present, Case 5 is the best. Regardless, when the implicit blanks are not handled explicitly, the conversion to macro code will not be handled exactly as most persons would expect.

Macro calls with large numbers of arguments

In the macro which forms the core of the UP (see Page 2) in the DWM, there are 40-50 arguments needed for several of the macros. Considering that there are, in addition, more than 100 different DST being managed by the DWM at this time, it is a daunting task to consider writing the macro calls for the system, and, more importantly, *revising* said calls. Anyone can write a system, but it takes a real person to maintain one!

This problem is solved in a relatively simple, but somewhat roundabout way, in the DWM, and some associated systems. In these systems, the different DST are all entered into a single dataset, the NDS. This dataset maintains all information about each DST, including information about BY variables, MSD name, MID name and other important information which is used during the UP. When calling the UP macro, the call requires essentially one parameter: the STN or an associated sequence number called `_DSNMB`.

Compare these two situations:

Situation 1:	<pre> %_update(_master=namex, _stn=nm, _bys=IDN datea, _long=Main biological data, _mstn=ids, _updtype=1) %_update(_master=arcxx, _stn=arc, _bys=IDN dateb, _long=Main architecture vx, _mstn=ids, _updtype=1) </pre>
Situation 2:	<pre> %_update(_stn=nm); %_update(_stn=arc); </pre>

It is clear that Situation 2 is easier to follow, and easier to understand.

Of course, in Situation 2, there is an additional step through the NDS. In this step, we place all relevant variable values into macro variables using CALL SYMPUT:

This approach is somewhat more tangential to set up. It requires a mechanism to handle the dataset containing the controlling information. In this approach, this dataset is very important, and any threat to this dataset could bring your entire operation to a crashing halt!

The main payoff for this approach comes when you wish to run a variety of updates in a row, of course. This can be done by the following:

```
%macro _overrun(_st, _fi);  
%do _zi=&_st. %to &_fi.;  %_update(_dsnmb=&_zi.);%end;  
%mend _overrun;  
%_overrun(12, 14);
```

In this case, the WHERE statement above is modified slightly, replacing the where condition (STN="&_stn.") with a numeric version (DSNMB=&_dsnmb.).

Tailoring the macro with different calls

In this approach, each of the different DST have different portions of the UP macro. For instance, when the ID are combined together, prior to the update process which places them into the MSD, certain ranges must be checked, and in some cases, values must be transformed. This is done by placing each of these code sections into a specific set of macros, one for each DST. During the UP, the files with these fragments are read in and processed. To ensure that the correct macro is used when the particular DST is being processed, calls are handled as follows:

```
DATA NEWVAL;SET X&_stn..NEW&_cn.;%&_stn.c1;RUN;
```

When the macro is run, the macro variables are resolved during execution. For instance, using the following values for variables and macros:

```
%let _stn=xxa; %let _cn=012;  
%macro _xxac1;ARRAY VALS XX1-XX12;DO I=1 TO 12; VALS{I}=VALS{I}-1; END;%mend _xxac1;
```

the final code which would be run in SAS would consist of:

```
DATA NEWVAL;SET Xxxa.NEW012;ARRAY VALS XX1-XX12;DO I=1 TO 12; VALS{I}=VALS{I}-1; END;RUN;
```

Installing the DWM

The DWM is installed quite easily. There is a single screen which tailors the system for any place that SAS runs on. The porting process involves indicating the names of 5 or 6 datasets, the names of 6 or 7 subdirectories and several other pieces of information. After the information is entered, a second button is toggled to install the system, which occurs in about 15 minutes.

Conclusion

The DWM is a method for controlling the processing of information in a SAS environment. It enables the DM to easily define DST, add data to ID, update data from ID into MSD and control the entire process from a single, well-designed interface. In addition, the process of adding new DST to the system is easily understood by anyone who can toggle a button. The system controls the entire process in one place, so that data can be far more simply and directly controlled.

Contacting the author. For more information about the DWM, or to arrange a demonstration, please contact:

Paul A. Thompson, Ph.D.
Department of Psychiatry
Laboratory of Biological Psychiatry
Program in Basic and Clinical Neuroscience
Case Western Reserve University
Cleveland, OH 44106
Phone: (216) 844-8946
Fax: (216) 844-5403

SAS is a registered trademark or trademark of SAS Institute Inc. in the USA and other countries. ® indicates USA registration.

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