

# Incorporating External Data Into the Data Warehouse

Tim Walters, InfoTech Marketing, Littleton, CO

## ABSTRACT

Most data warehouses focus solely on extracting data from internal operating systems. While this is valuable, additional data is available externally that can significantly enhance the value of the data warehouse. This paper presents ways that an intermediate user of Base SAS® (with a little SAS/ACCESS® thrown in) can incorporate external data into a data warehouse and report on such data. External data is defined as all data outside the organization's operating systems. Data from the company's spreadsheets can be directly included through SAS/ACCESS. Data from outside the company can be included at a summarized or detailed level by creating match variables and using the MERGE statement.

## INTRODUCTION

Most data warehouse development efforts focus on freeing the company's data from its operating systems and allowing users to query against the data. While this is a necessary first step in designing a data warehouse, and most users are happy that they can finally access the company's internal data, a data warehouse with only internal data is like undeveloped land: it has intrinsic value, but its value increases as it is built upon. External data is like that building: it adds much additional value to the warehouse. This is particularly true for marketing, which may need external data to analyze the company's customers, evaluate market share, compare actual results to plan, etc. While this paper has a marketing slant, the techniques can be applied to any external data.

## DEFINITION

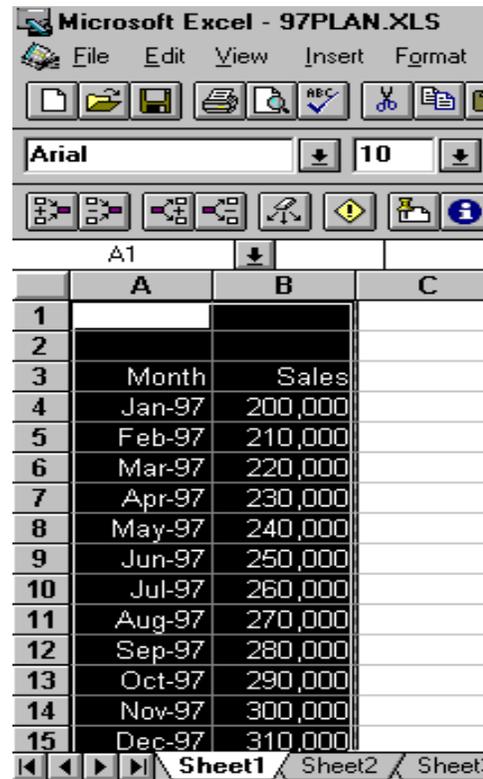
External data is defined as any data not contained in the company's operating systems. It can be data that the company has, but is not in an operating system. For example, budgeting data may be kept in spreadsheets in a P.C. and not be included in the company's accounting system. In this case, the company already has the data but needs it in the data warehouse for analysis purposes.

External data may also come from outside the organization at either a summary or detailed level. You may want to estimate penetration of accounts by sales territory. To do so, you can purchase summarized data from another company, like The Business Marketing Files from InfoTech Marketing. If you want to analyze your customer base, detailed "overlay" data containing

individual demographic or firmographic information can be purchased for consumers or businesses, respectively. This overlay data is merged with the company's individual account records to append information on each account. The accounts can then be segmented, for example, into high and low profitability groups and the market can be targeted to obtain additional high value customers.

## LINKING DATA THE COMPANY ALREADY HAS

Data from internal sources the company already has, such as forecast data, can be included in the data warehouse through ODBC drivers using SAS/ACCESS. These drivers can be used, for instance, to incorporate plan data or the most recent data on operating results from a spreadsheet. Suppose the company's monthly financial plan is in a spreadsheet and you want to compare actuals stored in the data warehouse to the plan. Let's say the Excel spreadsheet is arranged with months in column A and planned sales in column B, with the data specifically in cells a4:b15. See Figure 1.



	A	B	C
1			
2			
3	Month	Sales	
4	Jan-97	200,000	
5	Feb-97	210,000	
6	Mar-97	220,000	
7	Apr-97	230,000	
8	May-97	240,000	
9	Jun-97	250,000	
10	Jul-97	260,000	
11	Aug-97	270,000	
12	Sep-97	280,000	
13	Oct-97	290,000	
14	Nov-97	300,000	
15	Dec-97	310,000	

Figure 1. Spreadsheet Data

If you have SAS/ACCESS for PC File Formats, you can access the data directly quite simply by using an access

descriptor and view descriptor in the PROC ACCESS procedure. The view descriptor creates a temporary or permanent dataset that can then be used like any other file (see Figure 2).

Figure 2. SAS Code to Access Spreadsheet Data

```
LIBNAME STORE "C:\PLAN\";
PROC ACCESS DBMS=XLS;
*CREATE ACCESS DESCRIPTOR;
  CREATE STORE.HELP.ACCESS;
  PATH="C:\INFOTECH\97PLAN.XLS";
  WORKSHEET="SHEET1";
  RANGE="A4..B15";
  ASSIGN=YES;
  RENAME VAR0=MONTH VAR1=SALES;

*CREATE VIEW DESCRIPTOR;
  CREATE STORE.NEW.VIEW;
  SELECT ALL;
RUN;
PROC PRINT DATA=STORE.NEW;
RUN;
```

In this example, an access descriptor is created and named store.help.access, the path of the spreadsheet is defined, the worksheet containing the data within the spreadsheet is identified, the range of the data is defined

(a4.b15), and the default variables are renamed to month (for column A) and sales (for column B). The view descriptor is then defined as store.new.view with all (two) the variables selected. This view descriptor actually creates a permanent dataset that can be referred to as Store.New. The printout in Figure 3 shows the Month and Sales variables in Store.New.

Figure 3. Printout of Store.New Dataset

OBS	MONTH	SALES
1	JAN1997	200,000
2	FEB1997	210,000
3	MAR1997	220,000
4	APR1997	230,000
5	MAY1997	240,000
6	JUN1997	250,000
7	JUL1997	260,000
8	AUG1997	270,000
9	SEP1997	280,000
10	OCT1997	290,000
11	NOV1997	300,000
12	DEC1997	310,000

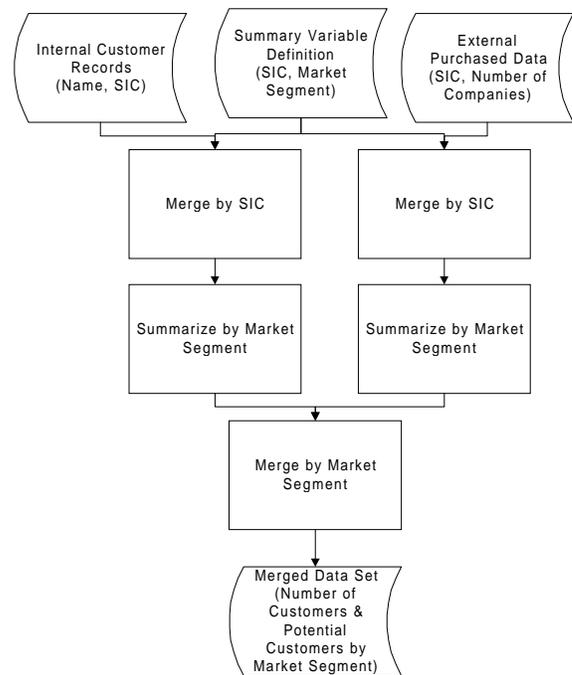
The file Store.New can then be used like any other data file. It can be merged by month with actual data, appended, summarized, etc.

If you do not have SAS Access for PC File Formats, you can output the spreadsheet as a comma separated (.CSV) file using Excel's File Save As command. You can then use the INFILE and INPUT statements to read in the resulting file. This may work for static data that doesn't change very much, such as annual plan data, but you can have problems ensuring you have the most recent data (which depends on users creating output files) with dynamically changing data.

## INCORPORATING EXTERNAL SUMMARIZED INFORMATION

External summarized information can fairly easily be included in the data warehouse using SAS summarization and merge capabilities. The key is to establish a common variable for merging the data. This variable can be sales territory code, market segment, state, etc. Both your internal and external data often need to be summarized by this common variable. A typical flowchart for adding this type of data is shown in Figure 4.

Figure 4. Flowchart for Adding Summarized Data



The company has three databases. The first is its customer master file that contains the customer's name and descriptive information. In this case, the descriptor is the SIC (Standard Industrial Classification) code, a standard descriptor of the type of business the customer conducts. The middle database defines how the company aggregates its customers. In this case, SIC codes are grouped into market segments. The third database is external purchased data such as the Department of Commerce's County Business Patterns or InfoTech

Marketing's The Business Marketing Files. This data reveals the number of businesses by SIC code. All three files are flat files that can be read with INFILE and INPUT statements.

For meaningful analysis, both the internal data and external data needs to be summarized and merged together. To begin, the database that defines the company's market segmentation scheme is merged with both the internal customer master file and external summarized data file to add the segments to each using PROC SORT and the MERGE and BY statements in the data step. The following code is an example of how to do this:

Figure 5. Code to Merge Two Files

```

*SORT THE INTETNAL CUSTOMER
FILE BY THE COMMON VARIABLE;
PROC SORT DATA=CUSTOMER;
  BY SIC;
*SORT THE SEGMENT DEFINITION
FILE BY THE COMMON VARIABLE;
PROC SORT DATA=MARKET;
  BY SIC;
*COMBINE THE FILES BY THE
COMON VARIABLE;
DATA COMBINE;
  MERGE CUSTOMER(IN=A)
  MARKET;
  BY SIC;
*KEEP THE OBSERVATIONS IN THE
CUSTOMER FILE;
IF A;

```

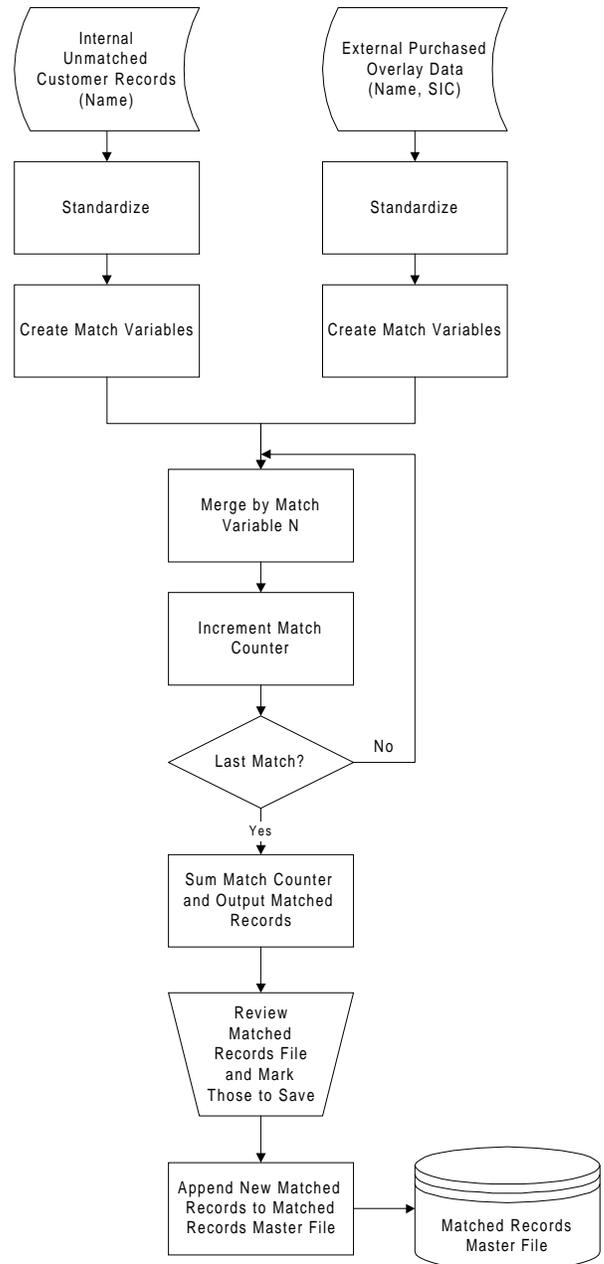
Both files are then summarized by the market segments, either by using PROC MEANS (or SUMMARY) or by aggregating the data in the data step. Because the market segment is a variable in each file, they are then merged (again by using PROC SORT and MERGE BY functionality) and one file is written. This file includes a count of the number of customers the company has by segment and the total market potential by segment. Numerous analyses of the segments can then be conducted.

### ADDING EXTERNAL OVERLAY DATA

External detailed data can be merged into the data warehouse to enhance your customer data. In the previous example, it was assumed that the customer master file had the SIC for each customer. Often, this is not the case. You may not collect the information you need, or you may collect it but not have it on all your observations. Overlay data purchased externally can add completely new variables to your internal systems or fill in the gaps of those observations with missing data.

Adding overlay data is often a multi-step process requiring human intervention. It is multi-step in order to conserve resources. Human intervention is required because the match process is inexact. A typical overlay process begins with standardizing data from both the internal and external databases. Various match routines are used and reviewed for accuracy. As matches occur, the observation is excluded from further matching. The following flow chart illustrates the process:

Figure 6. Flowchart for Adding Overlay Data



### Choosing the Internal Data to Overlay

Two of the first decisions in adding overlay data is the time period to perform the match and exactly what

internal records to try to overlay. This decision depends primarily on two factors: dynamics of the company and the number of customers. Dynamics of the company influence the regularity of performing the match. Start-up companies, for instance, may want to perform the match process monthly or even more frequently. More stable companies may want to add the data only quarterly. The sheer volume of the number of customers to match also influences the timing selection.

In addition, you need to decide how many times to try match a record. Because the overlay database changes, you probably want to try to match any unmatched records more than once to see if the record has been added to the external database. If you perform the match monthly, you may want to limit the records you match to those that haven't been matched in three months. After three months, chances are good that the record won't be added and there's no sense in wasting time trying to add overlay data that doesn't exist. Because of the changes in businesses, you should update all your internal data once per year to obtain the latest SIC code, business size, and parent data.

### **Standardizing the Data**

To increase the match chances, you should standardize the data in both the internal and external data. Techniques to standardize data include:

- Eliminating data that could be confusing. The SAS INDEX function finds the first occurrence of a string and can be used with the SUBSTRING function to eliminate any data beyond the occurrence of the data you want to eliminate. For instance, you may want to eliminate the suite number from address in order to match only on the physical address. The following code accomplishes this (against an address variable named UADD1) for either the abbreviation of "Ste" or "Suite":

```
*GETS RID OF SUITE NUMBERS;  
  
STEADD=INDEX(UADD1, 'STE' );  
  
IF STEADD EQ 0 THEN  
STEADD=INDEX(UADD1, 'SUITE' )  
;  
  
IF STEADD GT 1 THEN  
UADD1=SUBSTR(UADD1, 1, (STEAD  
D-1) );
```

- Converting data to the same (upper) case to eliminate any case sensitivity problems. The UPCASE function ensures that the data is converted to upper case.

- Eliminating vowels, characters, and spaces to perform phonetic matching. The COMPRESS function eliminates spaces or whatever is specified, as seen in the following code:

```
*ELIMINATES VOWELS AND
CHARACTERS;

FCOMCOMP=COMPRESS(UCOMPANY,
'AEIOU-%.&,' );

*ELIMINATES SPACES;

SCOMCOMP=COMPRESS(FCOMCOMP)
;
```

### Creating the Match Variables

After the data is standardized, common match variables are created. Some variables, such as telephone number, are used in their original form. Other variables are created using the SAS concatenation capabilities and the TRIM and SUBSTRING functions. A company involved in the facsimile services business chose to perform six different matches:

- 5 characters of contact name, 10 characters of company name, and 5 characters of city
- 6 characters of telephone number (area code and exchange) and 10 characters of company name
- 10 characters of company name, 5 characters of address, and 5 characters of city
- 10 characters of company name, 15 characters of address
- 5 characters of ZIP code and 20 characters of company name
- Telephone number

The following code shows how the first variable is created by trimming substrings of contact, company, and city, and then concatenating them together:

```
PHON1=TRIM(SUBSTR(CONTACT,1,5))||
TRIM(SUBSTR(COMPANY,1,10))||
TRIM(SUBSTR(CITY,1,5));
```

Similar statements are used to create the other variables.

### Performing the Matching

After the match variables are created, the internal and external data sets are both sorted by the first match variable and then merged together to create a new data set. If records with the same match variable are found in both data sets, by using the IN= option on both data sets, a new variable to count the number of matches is created and set equal to 1, as shown in the following code:

Figure 7. Code for Matching Overlay Data

```
PROC SORT DATA=OVERLAY;
  BY PHON1;
PROC SORT DATA=CUSTOMER;
  BY PHON1;
DATA NEW;
  MERGE CUSTOMER (IN=A)
        OVERLAY (IN=B
                KEEP=DUNSNO PHON1);
  BY PHON1;
* KEEPS CUSTOMER RECORDS;
  IF A;
* IF FOUND IN BOTH FILES AN
  NOT BLANK, SET
  MATCH VARIABLE EQUAL TO 1;
  IF A AND B AND PHON1 NE ""
  THEN MATCH1=1;
  ELSE MATCH1=0;
RUN;
```

The new data set and external data set are sorted with the next match variable and merged again. The match count variable is incremented by 1 if matching records are found in each data set. This process is repeated for each of the match variables.

### Saving the Matched External Records

After the matching is performed, any record with a match is output to a file for manual review and permanent selection. The output data set lists internal record components as record 1 and the matching external data as line 2. The analyst then indicates that they want to include the external record as a permanent match by placing an X beside it. A program is then ran to add the external record to a data file of previously matched external records.

The manual review process can be eliminated if you set a criteria above which you will automatically select the record. For instance, if you have six match criteria and an external record matches on two of them, you could automatically add the external record to the matched data set. This does run the risk of adding records that are not truly matches and excluding records that are matches but may not match exactly. In many cases, the manual review process can limit these risks. The risk of erroneous

matches should be weighed against the time required for manual review.

## **CONCLUSION**

Incorporating external data can dramatically increase the value of the data warehouse. This paper presented ways of using SAS to include three different types of external data:

1. Data internal to the company but not in an operating system;
2. Data external to the company but in a summarized format;
3. Data external to the company in a detailed, individual record format.

Although the focus has been on marketing data, these techniques can also be used to include other types of data. With these relatively easy ways of including external data, you should not shy away from including these rich data sources in your data warehouse.

SAS and SAS/ACCESS are registered trademarks or trademarks of SAS Institute Inc. in the USA and other countries. ® indicates USA registration.

Tim Walters, InfoTech Marketing, 9350 W. Cross Dr., Suite 203, Littleton, CO 80123, voice: 303-904-3045, fax: 303-727-4690, e-mail: InfoTecMkt@AOL.COM