Problem Statement

Although SAS software provides very powerful means of developing summary and inferential statistics, the use of SAS in batch mode presents some limitations in easily updating SAS datasets with corrections and changes to data. In some applications, transaction records can be used to update a dataset on a scheduled basis. Such situations usually have a uniform set of variables which are unchanged in value. An example is the classic payroll application where values for year-to-date wages, taxes withheld, etc. are regularly made. The use of the UPDATE data step statement in the SAS system is optimal for such an application, as only the observations and variables which require changes are affected. Programs can be developed which rarely change over time, for those variable names that are updated fairly infrequently.

There are some applications, however, where changes need to be made to different variables from one time to another. This can occur because the data is later found to be simply incorrect or for other reasons. One technique that can be used in these applications to change such data is to create specific data code in a program to perform the update. In its simplest form this is the following:

```sas
DATA BASE.CURRENT;
SET BASE.CURRENT;
IF ID='123' THEN CITY='BOSTON';
IF ID='456' THEN FIRSTAGE=23;
IF ID='799' THEN
ID;
DIAG1=889;
DIAG2=340;
END;
```

This "hard-coding" approach certainly works in updating a dataset with new data. However, it is computationally inefficient. For updating only those observations, each observation in the entire dataset must be read and tested for whether it contains a particular ID value. Each observation of the data set is also written from one place on disk to another.

A further weakness of this approach is that a program has to be created or edited each time a set of updates needs to be made. Changing a whole program introduces the usual risks of making typographical errors in the program code, even inadvertent errors in parts of the program usually not changed.

A second technique could be to supply only the new data in text form and read it in using column and/or formatted INPUT. This transaction dataset is then sorted for use in an UPDATE statement within a subsequent DATA step. Assuming that the BASE.MASTER dataset is already in sorted order by ID, the following example illustrates this technique:

```sas
DATA TRANSACTION;
INPUT ID 1-4;
CITY $ 5-14;
820 FIRSTAGE 2;
825 (DIAG1 = DIAG2) (4.) ;
PROC SORT DATA=TRANSACTION;
BY ID;
DATA BASE.MASTER:
UPDATE BASE.MASTER TRANSACTION
BY ID;
```

This method, however, makes the task of creating the raw text data file a tedious one of entering the correct data in the correct columns. Errors are highly likely while performing this data entry "freehand" even if there are columnar templates in the text editor. Using other software is a possibility for creating the raw data but this would incur an additional use of resources in a data system shop. Further, a fairly extensive INPUT statement may be required in datasets with large numbers of variable names, as there must be delineation of any variable name to be changed in a batch, even if data is to be changed in that variable for only one observation.

SAS software provides a convenient procedure (PROC EDITOR) which avoids some of the problems inherent in techniques such as those used above. The following code illustrates an example of use of this procedure.

```sas
PROC EDITOR DATA=BASE.CURRENT;
FIND VER 1, LAST ID=423;
REPLACE CITY='BOSTON';
VERIFY RESET;
FIND VER 1, LAST ID=456;
REPLACE FIRSTAGE=23;
VERIFY RESET;
FIND VER 1, LAST ID=789;
REPLACE DIAG1=889 DIAG2=340;
END;
```

The procedure uses the FIND command to locate an observation by specifying a condition (e.g. ID=456) which the observation must satisfy. In

---

BATCH USE OF THE EDITOR PROCEDURE FOR UPDATING SAS® DATASETS

Robert Rosofsky, Massachusetts Department of Public Health
the above cases, we are specifying a range of first to last observations.

By default, if the FIND command is unsuccessful, the observation "pointer" is automatically positioned to the last observation in the dataset and the subsequent REPLACE command could be executed incorrectly on that last observation. We want to execute the REPLACE command only if the FIND is successful. The VERIFY option of the FIND command is used to put this condition into effect. Such a combination of commands and options of the PROC EDITOR prevents any updating to the dataset with an invalid ID.

However, if the FIND command fails at any one time, by default PROC EDITOR will not "go" to make any subsequent changes (the NOGO switch is turned on). In order to turn off the NOGO switch for subsequent changes a "VERIFY RES" statement is required.

Although this method using PROC EDITOR repeatedly searches the dataset for the correct ID, it is more efficient in operation than the earlier approaches. It avoids the use of multiple data steps, possible large uses of storage for numerous missing variables, and resource intensive sorting. But as in the first approach, a weakness of this method is that a program has to be created or edited each time a set of updates need to be made.

A more ideal approach is to eliminate the use of this "hard-coding." The corrections required here are data changes and they should be utilized by a program in the form of data. Such "data-driven" approaches allow a program to be written, debugged, tested, and used for different datasets as a utility, where the corrections exist as data to be fed into the program, rather than as program code.

Proposed Technique

Using the example above, if there is a variable called "ID" as the key into a dataset, we can then imagine a text file which contains the values of the ID, the variable name(s) to be changed and the new value(s). These three elements are logically all of the information needed for any change to a dataset; i.e., identify the record to be changed, identify the variable to be changed, and identify the new value. This would look like the following:

<table>
<thead>
<tr>
<th>ID</th>
<th>Variable name</th>
<th>New Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>123</td>
<td>CITY</td>
<td>BOSTON</td>
</tr>
<tr>
<td>456</td>
<td>FIRSTNAME</td>
<td>23</td>
</tr>
<tr>
<td>789</td>
<td>MNAME</td>
<td>889</td>
</tr>
<tr>
<td>789</td>
<td>MNAME</td>
<td>340</td>
</tr>
</tbody>
</table>

where each line represents one change to be made. This data is suitable for "list INPUT" and is quite easy to create using text editors in micro, mini, or mainframe environments. No formatting is required.

We can then read this data and use it to "drive" a PROC EDITOR in order to update a dataset. This is illustrated by the following steps, using a file of birth certificate data as an example.

STEP 1: Using the output data set feature of PROC CONTENTS, we create a data set of the contents of the master file to be updated. This data set is used exclusively as a lookup table for checking that the variable names in the correction transactions are valid.

    PROC CONTENTS DATA=BASE.MASTER NOPRINT OUT=VARLIST(KEEP=NAME);

The VARLIST data set looks as follows:

<table>
<thead>
<tr>
<th>OBS</th>
<th>NAME</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>BM</td>
</tr>
<tr>
<td>2</td>
<td>DB</td>
</tr>
<tr>
<td>3</td>
<td>FAGE</td>
</tr>
<tr>
<td>4</td>
<td>FEAGE</td>
</tr>
<tr>
<td>5</td>
<td>ID</td>
</tr>
<tr>
<td>6</td>
<td>INAME</td>
</tr>
<tr>
<td>7</td>
<td>MAGE</td>
</tr>
<tr>
<td>8</td>
<td>MEDUC</td>
</tr>
<tr>
<td>9</td>
<td>PLURAL</td>
</tr>
<tr>
<td>10</td>
<td>RES</td>
</tr>
<tr>
<td>11</td>
<td>SEX</td>
</tr>
</tbody>
</table>

STEP 2: The corrections are input using "list" style INPUT. All VALUES come in as character data regardless of their type in the dataset.

    DATA TRANS;
    INFILE TRANS;
    INPUT ID NAME $ VALUE $;

This data set is sorted by name for the correct collating sequence, to be used in a subsequent MERGE step.

    PROC SORT DATA=TRANS;
    BY NAME;

and a print of this data set produces:

<table>
<thead>
<tr>
<th>OBS</th>
<th>ID</th>
<th>NAME</th>
<th>VALUE</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>1</td>
<td>BM</td>
<td>1989</td>
</tr>
<tr>
<td>2</td>
<td>5</td>
<td>FAGE</td>
<td>35</td>
</tr>
<tr>
<td>3</td>
<td>10</td>
<td>FAGE</td>
<td>34</td>
</tr>
<tr>
<td>4</td>
<td>25</td>
<td>LASTNAME</td>
<td>JONES</td>
</tr>
<tr>
<td>5</td>
<td>20</td>
<td>INAME</td>
<td>JOHNSTON</td>
</tr>
<tr>
<td>6</td>
<td>18</td>
<td>INAME</td>
<td>RENIS</td>
</tr>
<tr>
<td>7</td>
<td>20</td>
<td>MAGE</td>
<td>88</td>
</tr>
<tr>
<td>8</td>
<td>10</td>
<td>PLURAL</td>
<td>8</td>
</tr>
<tr>
<td>9</td>
<td>40</td>
<td>PLURAL</td>
<td>3</td>
</tr>
<tr>
<td>10</td>
<td>15</td>
<td>SEX</td>
<td>7</td>
</tr>
</tbody>
</table>

STEP 3: A MERGE is then performed of the dataset of variable names from step 1 (VARLIST) and the dataset of input corrections from step 2. Two data sets are created in this one step: one for valid variable names and one for invalid variable names.
DATA VALIDVAR /* Valid variable names */
   INVALIDVAR /* Invalid variable names */
   MEK2 TRANS (IN=IN_TRANS)
   VARLIST (IN=IN_VARS)
   BY NAME;
IF IN_TRANS AND IN_VARS THEN OUTPUT
   VALIDVAR;
ELSE if IN_TRANS AND NOT IN_VARS THEN
   OUTPUT INVALIDVAR;
Using PROC PRINT, the INVALIDVAR dataset can be
printed and examined for subsequent
Corrections of the source data as shown.

Step 4: The VALIDVAR data set is used to write
cut PROC EDITOR statements to a temporary text
file using PUT statements. This file is later
"INCLINED" when PROC EDITOR is run.

**DATA_NULL;**
SET VALIDVAR:
PUT 'FIND VER 1, LAST ID = ID 4. ;'
/' REPLACE 'NAME' = 'VALUE' ;'
/' VERIFY RESET ;';

The environment and operating system determine
how to set up a temporary file which SAS
writes to for the Dname "INCLINDEX" above.
Under OS/360 typical JCL lines would be:
//INCLINDEX DD DSN=WMPASS,DISP=(NEW,PASS),
// SPACE=(CY6,11),RULE,UNIT=SYSDA,
// DCB=(RECFM=FB,RECL=80,LRECL=6160)

The text dataset consisting exclusively of
SAS program statements for the current example
would look like the following:

FIND VER 1, LAST ID = 5 ;
REPLACE PACE = 35 ;
VERIFY RESET ;
FIND VER 1, LAST ID = 10;
REPLACE PAGE = 34
VERIFY RESET
FIND VER 1, LAST ID = 20;
REPLACE ADAME = JOHNSTON ;
VERIFY RESET ;
FIND VER 1, LAST ID = 18;
REPLACE NAME = RONIG ;
VERIFY RESET ;
FIND VER 1, LAST ID = 7;
REPLACE NAME = 88 ;
VERIFY RESET ;
FIND VER 1, LAST ID = 10;
REPLACE PLURAL = 8 ;
VERIFY RESET ;
FIND VER 1, LAST ID = 40;
REPLACE PLURAL = 3 ;
VERIFY RESET ;
FIND VER 1, LAST ID = 15;
REPLACE SEX = 7 ;
VERIFY RESET ;

**Step 5:** PROC EDITOR is run to update the dataset. No further data steps are required.

**PROC EDITOR DATA=BASE.MASTER;**
INCLUDE INCLINDEX;
END;

The INCLUDE line executes the program code written
by the earlier DATA step. The END
statement is required to shut off the PROC
EDITOR.

Sample output from the run of the PROC
EDITOR step looks like the following. With SAS
Version 5 PROC EDITOR, the NOTES are printed on
the log just prior to the statement line.

Note: WELCOME TO THE EDITOR PROCEDURE. BEGIN
ENTERING COMMANDS.
53 PROC EDITOR DATA=BASE.MASTER;
54 INCLUDE INCLINDEX;
NOTE: FOUND AT OBS 5.
55 +FIND VER 1, LAST ID = 5 ; 1//INCLINDEX
56 +REPLACE PAGE = 35 ; 1//INCLINDEX
57 +VERIFY RESET ; 1//INCLINDEX
NOTE: FOUND AT OBS 10.
58 +FIND VER 1, LAST ID = 10; 1//INCLINDEX
59 +REPLACE PAGE = 34 1//INCLINDEX
60 +VERIFY RESET ; 1//INCLINDEX
NOTE: FOUND AT OBS 22.
61 +FIND VER 1, LAST ID = 20; 1//INCLINDEX
62 +REPLACE NAME = JOHNSTON ; 1//INCLINDEX
63 +VERIFY RESET ; 1//INCLINDEX
NOTE: FOUND AT OBS 10.
64 +FIND VER 1, LAST ID = 18; 1//INCLINDEX
65 +REPLACE NAME = RONIG ; 1//INCLINDEX
66 +VERIFY RESET ; 1//INCLINDEX
NOTE: FOUND AT OBS 20.
67 +FIND VER 1, LAST ID = 7; 1//INCLINDEX
68 +REPLACE NAME = 88 ; 1//INCLINDEX
69 +VERIFY RESET ; 1//INCLINDEX
NOTE: FOUND AT OBS 10.
70 +FIND VER 1, LAST ID = 10; 1//INCLINDEX
71 +REPLACE PLURAL = 8 ; 1//INCLINDEX
72 +VERIFY RESET ; 1//INCLINDEX
NOTE: NOT FOUND. OBS=30.
73 +FIND VER 1, LAST ID = 40; 1//INCLINDEX
ERROR: NO CHANGES MADE BECAUSE NOO DSWITCH IS
SET.
74 +REPLACE PLURAL = 3 ; 1//INCLINDEX
75 +VERIFY RESET ; 1//INCLINDEX
NOTE: FOUND AT OBS 15.
76 +FIND VER 1, LAST ID = 15; 1//INCLINDEX
77 +REPLACE SEX = 7 ; 1//INCLINDEX
78 +VERIFY RESET ; 1//INCLINDEX
79
NOTE: EXITING FROM THE EDITOR PROCEDURE.
79 END;

An examination of the log indicates the successful and unsuccessful edits in the
dataset. Tables 1 and 2 at the end of this article illustrate the sample dataset before
and after this technique is used.

Additions can be made to this approach for
further data integrity, but at some cost. For ex
ample, the dataset should be backed up with
at least one generation before editing in case
of system failure, incorrect correction data,
or other problems. More complicated keys into a dataset can be handled by the technique. And checks for the validity of the transaction data can be developed.

## Table 1

Data Set "MASTER"

Before Updating With "PROC EDITOR"

<table>
<thead>
<tr>
<th>OBS</th>
<th>ID</th>
<th>NAME</th>
<th>BW</th>
<th>MAGE</th>
<th>FAGE</th>
<th>PRSN</th>
<th>SEX</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>1</td>
<td>SHINE</td>
<td>3062</td>
<td>29</td>
<td>34</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>2</td>
<td>2</td>
<td>SHAPERO</td>
<td>3402</td>
<td>19</td>
<td>20</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>3</td>
<td>3</td>
<td>SCHNEITZEL</td>
<td>3912</td>
<td>24</td>
<td>25</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>4</td>
<td>4</td>
<td>WISNIESKI</td>
<td>3912</td>
<td>16</td>
<td>99</td>
<td>0</td>
<td>2</td>
</tr>
<tr>
<td>5</td>
<td>5</td>
<td>FARRELL</td>
<td>3912</td>
<td>26</td>
<td>33</td>
<td>0</td>
<td>2</td>
</tr>
<tr>
<td>6</td>
<td>6</td>
<td>HELSING</td>
<td>2892</td>
<td>40</td>
<td>62</td>
<td>0</td>
<td>2</td>
</tr>
<tr>
<td>7</td>
<td>7</td>
<td>TORDJERAN</td>
<td>3175</td>
<td>19</td>
<td>21</td>
<td>0</td>
<td>2</td>
</tr>
<tr>
<td>8</td>
<td>8</td>
<td>RANSEN</td>
<td>3714</td>
<td>26</td>
<td>30</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>9</td>
<td>9</td>
<td>SHEFF</td>
<td>3245</td>
<td>26</td>
<td>29</td>
<td>0</td>
<td>2</td>
</tr>
<tr>
<td>10</td>
<td>10</td>
<td>ROSENFELD</td>
<td>3317</td>
<td>38</td>
<td>48</td>
<td>0</td>
<td>2</td>
</tr>
<tr>
<td>11</td>
<td>11</td>
<td>PETERSON</td>
<td>3260</td>
<td>25</td>
<td>32</td>
<td>0</td>
<td>2</td>
</tr>
<tr>
<td>12</td>
<td>12</td>
<td>FRANKLIN</td>
<td>3657</td>
<td>25</td>
<td>27</td>
<td>0</td>
<td>2</td>
</tr>
<tr>
<td>13</td>
<td>13</td>
<td>KNOWLES</td>
<td>3717</td>
<td>21</td>
<td>23</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>14</td>
<td>14</td>
<td>AGERBERG</td>
<td>4167</td>
<td>29</td>
<td>30</td>
<td>0</td>
<td>2</td>
</tr>
<tr>
<td>15</td>
<td>15</td>
<td>RALPH</td>
<td>2750</td>
<td>31</td>
<td>30</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>16</td>
<td>16</td>
<td>KLEIN</td>
<td>3430</td>
<td>29</td>
<td>31</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>17</td>
<td>17</td>
<td>DAVIS</td>
<td>3147</td>
<td>26</td>
<td>30</td>
<td>0</td>
<td>2</td>
</tr>
<tr>
<td>18</td>
<td>18</td>
<td>SWANSON</td>
<td>3175</td>
<td>30</td>
<td>31</td>
<td>0</td>
<td>2</td>
</tr>
<tr>
<td>19</td>
<td>19</td>
<td>AZZARA</td>
<td>1928</td>
<td>33</td>
<td>33</td>
<td>0</td>
<td>2</td>
</tr>
<tr>
<td>20</td>
<td>20</td>
<td>COMERFORD</td>
<td>3005</td>
<td>22</td>
<td>59</td>
<td>0</td>
<td>2</td>
</tr>
</tbody>
</table>

The asterisks indicate updated observations.

## Table 2

Data Set "MASTER"

After Updating With "PROC EDITOR"

<table>
<thead>
<tr>
<th>OBS</th>
<th>ID</th>
<th>NAME</th>
<th>BW</th>
<th>MAGE</th>
<th>FAGE</th>
<th>PRSN</th>
<th>SEX</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>1</td>
<td>SHINE</td>
<td>3062</td>
<td>29</td>
<td>34</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>2</td>
<td>2</td>
<td>SHAPERO</td>
<td>3402</td>
<td>19</td>
<td>20</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>3</td>
<td>3</td>
<td>SCHNEITZEL</td>
<td>3912</td>
<td>24</td>
<td>25</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>4</td>
<td>4</td>
<td>WISNIESKI</td>
<td>3912</td>
<td>16</td>
<td>99</td>
<td>0</td>
<td>2</td>
</tr>
<tr>
<td>* 5</td>
<td>5</td>
<td>FARRELL</td>
<td>3912</td>
<td>26</td>
<td>35</td>
<td>0</td>
<td>2</td>
</tr>
<tr>
<td>6</td>
<td>6</td>
<td>HELSING</td>
<td>2892</td>
<td>40</td>
<td>62</td>
<td>0</td>
<td>2</td>
</tr>
<tr>
<td>* 7</td>
<td>7</td>
<td>TORDJERAN</td>
<td>3175</td>
<td>88</td>
<td>21</td>
<td>0</td>
<td>2</td>
</tr>
<tr>
<td>8</td>
<td>8</td>
<td>RANSEN</td>
<td>3714</td>
<td>26</td>
<td>30</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>9</td>
<td>9</td>
<td>SHEFF</td>
<td>3245</td>
<td>26</td>
<td>29</td>
<td>0</td>
<td>2</td>
</tr>
<tr>
<td>* 10</td>
<td>10</td>
<td>ROSENFELD</td>
<td>3317</td>
<td>38</td>
<td>48</td>
<td>0</td>
<td>2</td>
</tr>
<tr>
<td>11</td>
<td>11</td>
<td>PETERSON</td>
<td>3260</td>
<td>25</td>
<td>32</td>
<td>0</td>
<td>2</td>
</tr>
<tr>
<td>12</td>
<td>12</td>
<td>FRANKLIN</td>
<td>3657</td>
<td>25</td>
<td>27</td>
<td>0</td>
<td>2</td>
</tr>
<tr>
<td>13</td>
<td>13</td>
<td>KNOWLES</td>
<td>3717</td>
<td>21</td>
<td>23</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>14</td>
<td>14</td>
<td>AGERBERG</td>
<td>4167</td>
<td>29</td>
<td>30</td>
<td>0</td>
<td>2</td>
</tr>
<tr>
<td>* 15</td>
<td>15</td>
<td>RALPH</td>
<td>2750</td>
<td>31</td>
<td>30</td>
<td>0</td>
<td>2</td>
</tr>
<tr>
<td>16</td>
<td>16</td>
<td>KLEIN</td>
<td>3430</td>
<td>29</td>
<td>31</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>17</td>
<td>17</td>
<td>DAVIS</td>
<td>3147</td>
<td>26</td>
<td>30</td>
<td>0</td>
<td>2</td>
</tr>
<tr>
<td>* 18</td>
<td>18</td>
<td>SWANSON</td>
<td>3175</td>
<td>30</td>
<td>31</td>
<td>0</td>
<td>2</td>
</tr>
<tr>
<td>19</td>
<td>19</td>
<td>AZZARA</td>
<td>1928</td>
<td>33</td>
<td>33</td>
<td>0</td>
<td>2</td>
</tr>
<tr>
<td>* 20</td>
<td>20</td>
<td>JOHNSTON</td>
<td>3005</td>
<td>88</td>
<td>99</td>
<td>0</td>
<td>2</td>
</tr>
</tbody>
</table>

The asterisks indicate updated observations.

### Acknowledgements:

To Cynthia Wisniewski, Programmer/Analyst at the Massachusetts Department of Public Health, for initially suggesting PROC EDITOR as a possible solution to the problem addressed by this paper.

### References:


SAS is the registered trademark of SAS Institute Inc., Cary, NC, USA.

Please address questions, suggestions, or discussions of similar work and issues to:

Robert Rosofsky
Bureau of Parent, Child, and Adolescent Health
Department of Public Health
150 Tremont Street, 4th Floor
Boston, MA 02111