

# Reading and Writing Data from Microsoft Excel/Word Using DDE

## Prepared by Destiny Corporation

### Dynamic Data Exchange to Excel

Dynamic Data Exchange is a means of transferring data between different Windows applications such as the SAS System and Microsoft Excel or Microsoft Word. The technique uses what is known as a Client / Server type relationship, where the Client application makes the requests and the Server application responds accordingly. When using DDE with the SAS System, SAS is always the Client.

The DDE technique may be used within a Data Step, a Macro or from inside a SAS/AF application.

Using the DDE technique involves specifying the following three components in what is called a DDE Triplet:

1. The path and name of the executable file for the server application. For example, with Microsoft Excel the name of the executable is EXCEL.
2. The full path and file name of the document or spreadsheet with which you wish to share data. This is referred to in SAS documentation as the Topic.
3. The location of data to be read or modified. With a spreadsheet application, this is a range of cells. In applications such as Microsoft Word, the location of data is defined by what are known as bookmarks.

An example of a DDE Triplet for an Excel Spreadsheet is as follows:

```
EXCEL | C:\COURSES\DATA\CLIN.XLS |  
R4C17:R17C8
```

The DDE Triplet is then incorporated into a Filename statement of the following form:

```
FILENAME fileref DDE 'DDE-Triplet' |  
'CLIPBOARD' <DDE-Options>;
```

where:

**fileref** is a valid fileref

**DDE** is the keyword that primes the SAS System for Dynamic Data Exchange.

'DDE-Triplet' is the three part specification of the DDE external file and takes the following form: 'Executable-file-name|Topic|Data-range'. Note that the triplet is application dependent and may differ between different releases of the same application.

**CLIPBOARD'** is an alternative to directly specifying the DDE triplet. To use this option, copy the required data from the Server application to the Clipboard and specify the 'CLIPBOARD' option in the Filename statement. SAS will then determine the Triplet for itself. You will be able to see the triplet definition by looking at the Log.

**DDE-Options** are as follows:

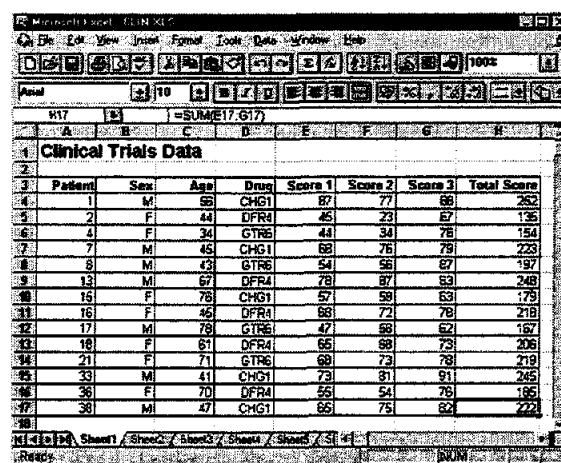
**HOTLINK** invokes the DDE HOTLINK facility, which causes the DDE link to be activated whenever data in the spreadsheet range is updated.

**NOTAB** makes the SAS System ignore tab characters between variables.

**COMMAND** allows remote commands to be sent to DDE server applications.

### Example of using DDE to read data from Excel

Consider the following Excel Spreadsheet:



| Patient | Sex | Age | Drug | Score 1 | Score 2 | Score 3 | Total Score |
|---------|-----|-----|------|---------|---------|---------|-------------|
| 1       | M   | 55  | CHG1 | 87      | 77      | 66      | 252         |
| 2       | F   | 44  | DPR4 | 45      | 23      | 67      | 135         |
| 4       | F   | 34  | GTR6 | 44      | 34      | 76      | 154         |
| 7       | M   | 45  | CHG1 | 68      | 76      | 79      | 223         |
| 8       | M   | 43  | GTR6 | 54      | 56      | 67      | 197         |
| 13      | M   | 67  | DPR4 | 78      | 87      | 83      | 248         |
| 15      | F   | 78  | CHG1 | 57      | 59      | 63      | 179         |
| 16      | F   | 45  | DPR4 | 68      | 72      | 76      | 218         |
| 17      | M   | 78  | GTR6 | 47      | 68      | 62      | 167         |
| 18      | F   | 61  | DPR4 | 65      | 69      | 73      | 206         |
| 21      | F   | 71  | GTR6 | 68      | 73      | 78      | 219         |
| 33      | M   | 41  | CHG1 | 73      | 81      | 91      | 245         |
| 36      | F   | 70  | DPR4 | 55      | 54      | 76      | 185         |
| 38      | M   | 47  | CHG1 | 65      | 75      | 82      | 222         |

With Excel running and the above spreadsheet open, a simple Data Step can be written to read the above data into a SAS Data Set.

The DDE Triplet components for the above example are as follows:

Name of Executable File: EXCEL

Full Path and Name of spreadsheet file:  
C:\COURSES\DATA\CLIN.XLS

Location of data (Row and Column coordinates):  
R4C1:R17C8

Note how the two sets of coordinates that define the range of data are separated by a colon.

The SAS statements required to read the data into a SAS data set are as follows:

```
1 INPUT PTHO SEX $ AGE DRUG $ SCORE1=SCORE2 TOTAL_SC ;
2 RUN;

NOTE: The infile XFILE is:
      FILENAME=\\file\\c:\courses\data\clin.xls;
      RECFM=U, LRECL=1056

NOTE: 14 records were read from the infile XFILE.
      The minimum record length was 35.
      The maximum record length was 35.
      The data set WORK.CLIN has 14 observations and 6 variables.
      NOTE: The DATA statement used 0.71 seconds.

3 /* Design a Filter to the Excel Spreadsheet and specify that
4    Dynamic Data Exchange is to be used */
5 FILENAME xfile DDE 'excel://c:\courses\data\clin.xls;R4C1:R17C8';
6
7 /* Create a Data Set called CLIN in the Work Library and read in
8    data from the Excel Spreadsheet */
9 DATA WORK.CLIN;
10 INFILE XFILE;
11 INPUT PTHO SEX $ AGE DRUG $ SCORE1=SCORE2 TOTAL_SC ;
12 RUN;
```

Running the above code generates the following data set:

| The SAS System |     |     |      |        |        |          |     |
|----------------|-----|-----|------|--------|--------|----------|-----|
| PTHO           | SEX | AGE | DRUG | SCORE1 | SCORE2 | TOTAL_SC |     |
| 1              | M   | 56  | CHG1 | 87     | 77     | 88       | 252 |
| 2              | F   | 44  | DFR4 | 45     | 58     | 57       | 185 |
| 3              | F   | 34  | DFR6 | 44     | 34     | 74       | 159 |
| 4              | M   | 45  | CHG1 | 69     | 76     | 73       | 223 |
| 5              | M   | 43  | GTR6 | 54     | 56     | 87       | 177 |
| 6              | F   | 57  | DFR4 | 79     | 87     | 83       | 249 |
| 7              | F   | 76  | CHG1 | 57     | 59     | 63       | 179 |
| 8              | F   | 45  | DFR4 | 68     | 72     | 78       | 218 |
| 9              | M   | 78  | DFR6 | 67     | 58     | 62       | 167 |
| 10             | F   | 51  | DFR6 | 65     | 68     | 73       | 206 |
| 11             | F   | 71  | DFR6 | 68     | 73     | 78       | 219 |
| 12             | M   | 51  | CHG1 | 73     | 81     | 91       | 245 |
| 13             | F   | 79  | DFR4 | 55     | 54     | 76       | 185 |
| 14             | M   | 47  | CHG1 | 85     | 75     | 82       | 222 |

### Example of using DDE to write data to Excel

To output data to an Excel spreadsheet, define a DDE Triplet which contains the co-ordinates of the cells to which data is to be written.

Then use a null data step containing 'Put' statements, to write out the values of required variables to the spreadsheet.

For example, the following code uses a Proc Summary to calculate simple statistics on the data that was originally read in from Excel. The statistics are output to a data set, which is used to update the Excel spreadsheet:

```
1 /* Derive simple statistics for the three score variables */
2 proc summary data=work.clin;
3   var score1=score2;
4   output out=work.summary;
5 run;

6 /* Design a Filter to the Excel Spreadsheet and specify that
7    Dynamic Data Exchange is to be used */
8 FILENAME xfile DDE 'excel://c:\courses\data\clin.xls;R20C1:R20C3';
9
10 /* Output statistics for Score variables to Excel Spreadsheet */
11 data _null_;
12   file xfile;
13   put _stat_ score1 score2 score3;
14 run;
```

Note that the only change to the Filename statement compared with the previous example, is that the DDE Triplet points at a different range of cells in the spreadsheet.

A null data step is used to read the data set containing the statistics and 'Put' statements write out the values of selected variables to the Excel spreadsheet. This is no different to writing out data to a text file.

Note that a null data step is used when you wish to avoid creating another data set.

Running the above code updates the Excel spreadsheet accordingly:

The updated Excel spreadsheet complete with summary statistics:

| Clinical Trials Data |     |     |      |         |          |          |             |
|----------------------|-----|-----|------|---------|----------|----------|-------------|
| Patient              | Sex | Age | Drug | Score 1 | Score 2  | Score 3  | Total Score |
| 1                    | M   | 56  | CHG1 | 87      | 77       | 88       | 252         |
| 2                    | F   | 44  | DFR4 | 45      | 29       | 87       | 135         |
| 3                    | F   | 34  | GTR6 | 44      | 34       | 76       | 154         |
| 4                    | M   | 45  | CHG1 | 69      | 76       | 79       | 223         |
| 5                    | M   | 43  | GTR6 | 54      | 56       | 87       | 197         |
| 6                    | M   | 67  | DFR4 | 79      | 87       | 83       | 249         |
| 7                    | F   | 76  | CHG1 | 57      | 59       | 63       | 179         |
| 8                    | F   | 45  | DFR4 | 68      | 72       | 78       | 218         |
| 9                    | M   | 78  | DFR6 | 67      | 58       | 62       | 167         |
| 10                   | F   | 51  | DFR6 | 65      | 68       | 73       | 206         |
| 11                   | F   | 71  | DFR6 | 68      | 73       | 78       | 219         |
| 12                   | M   | 51  | CHG1 | 73      | 81       | 91       | 245         |
| 13                   | F   | 79  | DFR4 | 55      | 54       | 76       | 185         |
| 14                   | M   | 47  | CHG1 | 85      | 75       | 82       | 222         |
|                      |     |     |      | N       | 14       | 14       | 14          |
|                      |     |     |      | MIN     | 44       | 29       | 62          |
|                      |     |     |      | MAX     | 87       | 87       | 91          |
|                      |     |     |      | MEAN    | 62.42857 | 63.78571 | 77.35714    |
|                      |     |     |      | STD     | 12.7141  | 18.01958 | 8.863123    |

### Invoking an application from SAS

In the previous examples, the invocation of Excel and opening of a spreadsheet file has been done by clicking on the required icon and using the pull down menus within Excel.

The application remains running in the background until the user actively closes it down.

Ideally, you only want an application running for the time that it takes to access the data. Otherwise, the idle application ties up valuable memory that can be

better used. The solution to this problem is to open and close applications from your SAS program.

An application may be called from SAS by using the following statements:

```
OPTIONS NOXWAIT NOXSYNC;
```

```
X 'C:\EXCEL\EXCEL';
```

```
DATA _NULL_;
    T=SLEEP(5);
RUN;
```

**NOXWAIT** causes control to be returned to SAS as soon as the specified 'X' command has been executed. In the above example, as soon as Excel has been loaded, control will return to SAS.

**NOXSYNC** causes the application called from SAS to execute asynchronously so that it does not interfere with the SAS session.

Remember to reset the above options to XWAIT and XSYNC when the application is closed. The X statement contains the path and name of the executable file that calls the application. The **Sleep** statement inside the null data step simply gives the called application time to load. In the above example, 5 seconds has been allowed.

### Issuing Commands from SAS

Having invoked an application like Excel using the above statements, you need to issue commands that carry out tasks such as opening the required spreadsheet file and selecting data. This is done through what is called the **SYSTEM** Topic:

Assign a Fileref that includes the DDE topic **SYSTEM** to enable Excel or Word commands to be issued from SAS.

```
FILENAME CMDS DDE 'Excel|SYSTEM';
```

Commands may now be sent to Excel by using 'Put' statements inside a null data step. Note that the syntax of the commands issued is application dependent.

```
DATA _NULL_;

    FILE CMDS;

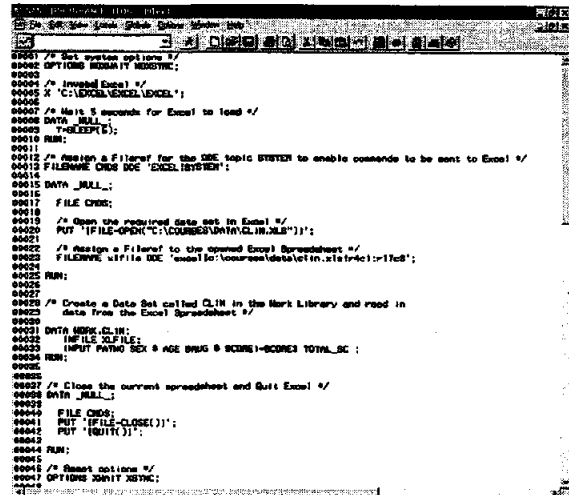
    PUT                                     '[FILE-
OPEN("C:\COURSES\DATA\CLIN.XLS")];
```

RUN;

### Example of invoking and issuing commands to Excel

The example program does the following:

1. Invokes Excel.
2. Opens a Spreadsheet.
3. Reads spreadsheet data into a SAS data set.
4. Closes the Spreadsheet.
5. Closes Excel.

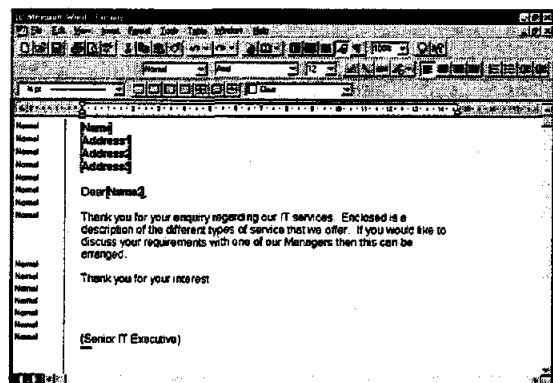


```
0001 /* Set system options */
0002 OPTIONS NOXWAIT NOXSYNC;
0003
0004 /* Invoke Excel */
0005 X 'C:\EXCEL\EXCEL';
0006
0007 /* Wait 5 seconds for Excel to load */
0008 DATA _NULL_;
0009     SLEEP(5);
0010 RUN;
0011
0012 /* Assign a Fileref for the DDE topic SYSTEM to enable commands to be sent to Excel */
0013 FILENAME CMDS DDE 'Excel|SYSTEM';
0014
0015 DATA _NULL_;
0016     FILE CMDS;
0017
0018     /* Open the required data set in Excel */
0019     PUT '[FILE-OPEN("C:\COURSES\DATA\CLIN.XLS")];'
0020
0021 /* Assign a Fileref for the opened Excel Spreadsheet */
0022 FILENAME XFILE DDE 'Excel|workbook\data\cln.xls|1:1';
0023
0024 RUN;
0025
0026 /* Create a Data Set called CLIN in the Work Library and read in
0027    data from the Excel Spreadsheet */
0028 DATA WORK.CLIN;
0029     INFILE XFILE;
0030     INPUT PATNO SEX & AGE WBSG & SCORE1=SCORES TOTAL_SC ;
0031 RUN;
0032
0033 /* Close the current spreadsheet and Quit Excel */
0034 DATA _NULL_;
0035     FILE CMDS;
0036     PUT '[FILE-CLOSE()];'
0037     PUT '[QUIT()];'
0038 RUN;
0039
0040 /* Reset options */
0041 OPTIONS XWAIT XSYNC;
```

### Dynamic Data Exchange to Word

Using DDE to access data stored in a Microsoft Word document is the almost the same procedure as used for Excel. The only difference is that the location of data in Word is specified by referencing Bookmarks rather than a cell range.

A bookmark is simply a named marker placed within the Word document. The following Word Document uses bookmarks to denote the position of Name and Address fields:



Word document content:

Name: \_\_\_\_\_

Address: \_\_\_\_\_

Address: \_\_\_\_\_

Dear [Name],

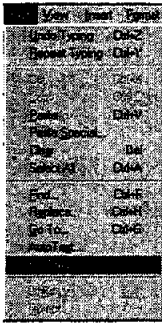
Thank you for your enquiry regarding our IT services. Enclosed is a description of the different types of service that we offer. If you would like to discuss your requirements with one of our Managers then this can be arranged.

Thank you for your interest

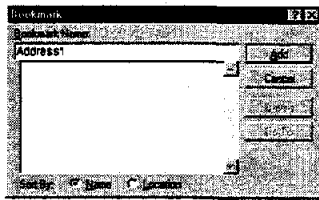
(Senior IT Executive)

To create a bookmark within a Word document, highlight the text which you want to use as a

bookmark, then select **Edit** and **Bookmark** from the pull down menu:



Enter a name for the bookmark and click on **Add**



The components of the DDE Triplet for the above example are as follows:

Name of Executable File: WINWORD

Full Path and Name of Word document:  
C:\COURSES\DATA\ENQUIRY.DOC

Location of data (Name of Bookmark): Address1

### Invoking Word from SAS

The procedure used to call Word from SAS is the same procedure used to call Excel or any other Windows application from SAS. Set the appropriate options, use an 'X' statement to call Word, and allow Word a few seconds to load.

```
OPTIONS NOXWAIT NOXSYNC;
```

```
X 'C:\WINWORD\WINWORD';
```

```
DATA _NULL_;
```

```
T=SLEEP(5);
```

```
RUN;
```

Remember to reset the above Options to XWAIT and XSYNC when the application is closed.

### Issuing Commands to Word from SAS

As with Excel, this is done by assigning a Fileref to the DDE topic SYSTEM:

```
FILENAME CMDS DDE 'WINWORD|SYSTEM';
```

Once the above Fileref is assigned, commands can be sent to Word by using 'Put' statements inside a null data step.

```
DATA _NULL_;
```

```
FILE CMDS;
```

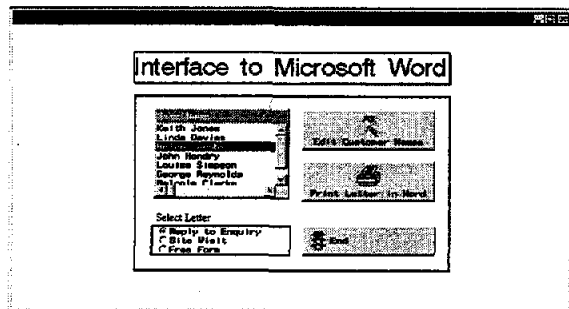
```
PUT
```

```
'[FILEOPEN("C:\COURSES\DATA\ENQUIRY.DOC")]';
```

```
RUN;
```

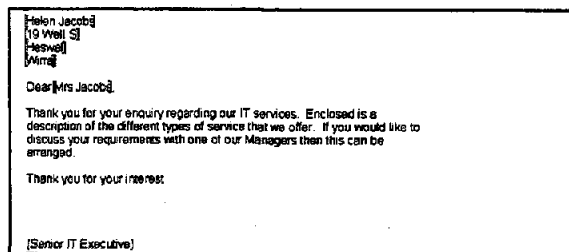
### Example of using DDE to access a Word document

The following SAS/AF Application automatically inserts the name and address of a customer into the selected Word document and prints it:



The name and address information is stored in a SAS data set that may be accessed and updated by clicking on the 'Edit Customer Names' icon.

The name and address information of the selected customer is inserted into the Word document as follows:

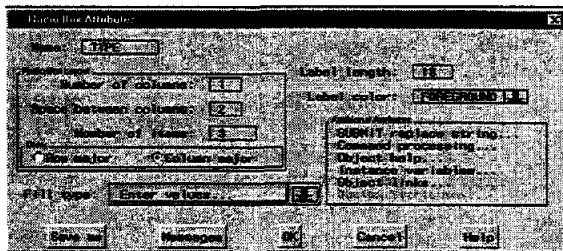


Note how the name and address details have been written to their respective bookmarks. The square brackets, which denote the boundary of a bookmark, do not appear in the printed document.

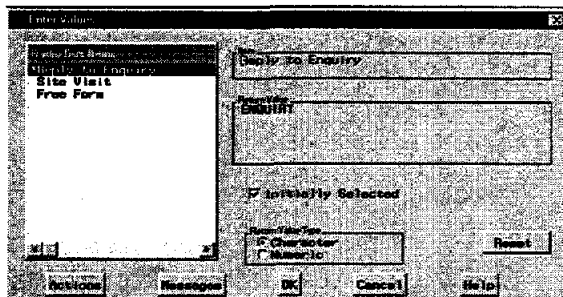
Note the following about the SCL Program that controls the Frame:

1. The physical paths for the Word executable file and Word documents, are assigned to Macro variables in the Autoexec file which runs before the application is called. Call Symget routines are used in the INIT section to assign the values of these Macro variables to SCL variables. This approach avoids having to hard code path names in the application.

- The name of the Word document to open is derived by concatenating the document path, letter type, and the suffix '.DOC'.
- The letter type is obtained by sending the `_GET_TEXT_` method to the Radio Box, which returns the Character value assigned to the selected item. Each Radio Box item has a Character return value specified in the attributes window:



Click on **Enter values...** to access Return values for items on



Clicking on a Radio Box Item displays its Return value.

- A Submit Block contains all of the SAS code used to load Word, access the selected document and print it. Recall that an ampersand within a Submit Block references an SCL variable.
- Five different Filref's are assigned to point at the various name and address bookmarks in the Word document.

The SCL Program is as follows (see next column):

```

0001 /* Declare the length of the Character variables */
0002 length address $20;
0003
0004 INIT:
0005
0006 /* Get the physical path to the Word Documents and to the Word
0007 executable which are stored as Macro variables */
0008 %let wordpath = %sysfunc(pathname('wordpath'));
0009 %let wordexec = %sysfunc(pathname('wordexec'));
0010
0011 return;
0012
0013
0014 MACRO:
0015
0016 /* Get the Name selected from the List Box */
0017 call notify('name', _get_text_._row._name._name);
0018
0019 return;
0020
0021 EDIT:
0022
0023 /* Open the FPDIT window */
0024 call notify('fpdit', 'fpdit');
0025
0026 /* Refresh the contents of the List Box */
0027 call notify('names', _row._name._name);
0028
0029 return;
0030
0031 PRINT:
0032
0033 /* If a name has been selected then generate and print the Word document */
0034 if address ne '' then do;
0035
0036 /* Get the Type of Letter selected from the Radio Box */
0037 call notify('type', _get_text_._letter);
0038
0039 /* Concatenate the Path and Letter Type to generate the name of the
0040 document to be opened in Word */
0041 wordfile = wordpath || _get_text_._letter || '.doc';
0042
0043 SUBMIT CONTINUE:
0044
0045 /* Set system options */
0046 options nocenter nosource;
0047
0048 /* Invoke Word */
0049 %let wordexec = %sysfunc(pathname('wordexec'));
0050
0051 /* Wait 5 seconds for Word to load */
0052 data _null_;
0053 run;
0054
0055 /* Assign a Filref for the SAS topic SYSTEM to enable commands
0056 to be sent to Word */
0057 fileref = 'wordexec';
0058
0059 DATA _null_;
0060
0061 FILE WORD;
0062
0063 /* Open the required Word Document */
0064 PUT 'IFILEOPEN('wordfile')';
0065
0066 /* Assign Filref's to the named bookmarks in the Word Document */
0067 FILENAME wordname 'wordname';
0068 FILENAME wordaddr 'wordaddr';
0069 FILENAME wordcity 'wordcity';
0070 FILENAME wordstate 'wordstate';
0071 FILENAME wordzip 'wordzip';
0072
0073 RUN;
0074
0075 /* Open the Data Set called names and read the observation that
0076 corresponds to the selected row in the List Box. Use File and Put
0077 statements to write out the value of variables to the appropriate
0078 position in the Word Document */
0079
0080 DATA _null_;
0081 SET NAMES (FIRSTOBS=1 LASTOBS=5);
0082
0083 FILE WORD;
0084 PUT wordname;
0085
0086 FILE WORD;
0087 PUT wordaddr;
0088
0089 FILE WORD;
0090 PUT wordcity;
0091
0092 FILE WORD;
0093 PUT wordstate;
0094
0095 FILE WORD;
0096 PUT wordzip;
0097
0098 RUN;
0099
0100 /* Print the Word Document */
0101 DATA _null_;
0102
0103 FILE WORD;
0104 PUT 'IFILEPRINT()';
0105
0106 RUN;
0107
0108 /* Close the current document and Quit Word */
0109 DATA _null_;
0110
0111 FILE WORD;
0112 PUT 'IFILECLOSE()';
0113
0114 RUN;
0115
0116 /* Reset options */
0117 options nocenter nosource;
0118
0119 END;
0120
0121 END;
0122
0123
0124 return;
0125
0126 END;
0127
0128 call execute('end');
0129
0130 return;
0131
0132
0133
0134
0135
0136
0137
0138
0139
0140
0141
0142
0143
0144
0145
0146
0147
0148
0149
0150
0151
0152
0153
0154
0155
0156
0157
0158
0159
0160
0161
0162
0163
0164
0165
0166
0167
0168
0169
0170
0171
0172
0173
0174
0175
0176
0177
0178
0179
0180
0181
0182
0183
0184
0185
0186
0187
0188
0189
0190
0191
0192
0193
0194
0195
0196
0197
0198
0199
0200

```

## Workshop Session

Create an Excel Spreadsheet and enter the text 'Demographic Data' in cell A1. Save the spreadsheet and make a note of the path and file name used.

Write a SAS program to do the following:

- Invoke Excel.
- Open the spreadsheet file that you have created for this exercise.
- Write out values of Demographic Data Set variables to the Excel spreadsheet.
- Save the spreadsheet.
- Close the spreadsheet file.
- Close Excel.

The resulting spreadsheet should appear as follows:

| Demographic Data |                   |       |      |      |   |   |            |
|------------------|-------------------|-------|------|------|---|---|------------|
| 1                | Dem Dem           | 22 M  | 64   | 60 S | 0 | 1 | 13595 low  |
| 2                | Julia Pandelov    | 30 F  | 67   | 74 M | 1 | 1 | 8670 low   |
| 3                | Norman Hertz      | 100 M | 65   | 62 D | 2 | 1 | 12572 low  |
| 4                | Harold Hertz      | 28 M  | 64   | 74 M | 3 | 1 | 23760 high |
| 5                | Mary Moleworth    | 23 F  | 64   | 68 P | 1 | 1 | 10612 low  |
| 6                | Rob Stoenberg     | 30 M  | 67   | 76 S | 0 | 1 | 7320 low   |
| 7                | Paula Bamford     | 34 F  | 76   | 72 M | 2 | 1 | 28612 high |
| 8                | Julia Todd        | 38 F  | 68   | 58 M | 2 | 1 | 14840 low  |
| 9                | Robert Conderford | 26 F  | 66   | 68 M | 3 | 2 | 47520 high |
| 10               | Mary Croft        | 32 F  | 66   | 69 M | 0 | 1 | 23760 high |
| 11               | Jennifer Crofton  | 30 F  | 69 S | 71 M | 3 | 2 | 30556 high |
| 12               | John Jennings     | 36 F  | 61   | 61 M | 0 | 2 | 13780 low  |
| 13               | Shirley Walters   | 46 F  | 63   | 75 W | 2 | 1 | 47620 high |
| 14               | Deane Bailey      | 65 F  | 69   | 81 M | 2 | 1 | 15840 low  |
| 15               | Dean Dore         | 73 F  | 74   | 80 S | 0 | 1 | 12840 low  |

### Hints:

Use a separate Fileref to define the range of cells into which people's names are written and include the NOTAB option i.e.:

```
FILENAME          XLF1          DDE
'EXCEL|C:\COURSES\DATA\DEMOG.XLS!r3c1:
r52c1' NOTAB;
```

The NOTAB option makes SAS ignore Tab Characters between variables. In this example, the space between the first name and surname is taken as a Tab Character, which results in the two parts of the name being put into different columns in Excel unless NOTAB is specified. Use a second Fileref to define the range of cells for the remaining variables but do not include the NOTAB option as non of the variables include space characters:

```
FILENAME          XLF2          DDE
'EXCEL|C:\COURSES\DATA\DEMOG.XLS!r3c1:r52c1';
```

Within a null Data Step, use separate File and Put statements of the following form to output the data:

```
DATA _NULL_;
  FILE XLF1;
  PUT NAME;
  FILE XLF2;
  PUT AGE GENDER etc;
RUN;
```

Remember to reset options NOXWAIT and NOXSYNC to XWAIT and XSYNC respectively.

Prepared by:

Destiny Corporation  
 100 Great Meadow Rd., Suite 601  
 Wethersfield, CT 06109-2355  
 Phone: (860) 721-1684 • 1-800-7TRAINING  
 Fax: (860) 721-9784  
 Email: [info@destinycorp.com](mailto:info@destinycorp.com)  
 Web Site: [www.destinycorp.com](http://www.destinycorp.com)