203P-2013
Extending the Power of Base SAS® with Microsoft Excel

Shilpa Khambhati, Mathematica Policy Research, Princeton, NJ

Abstract

This paper describes how to use the SAS Macro Language and Microsoft Excel to automatically generate customized reports. The proposed method uses Excel macros to drive SAS macros without having to open SAS programs and manually upgrade parameters for each site's data when the data become available. The process eliminates manually editing SAS programs and improves data quality by reducing programming error and program maintenance time.

Introduction

This poster explains how to develop the Excel base front-end application, which incorporates Excel features such as data validation, look-up tables, and Excel macro with Base SAS to run SAS macros based on parameters selected in the Excel file. The result is a simple, user-friendly interface in Excel for running automated SAS routines.
To illustrate this process, we describe a project that consists of three main tasks:

1. Define and organize the output, input, and SAS program names and location folders.
2. Set up an Excel file with the corresponding SAS program macro variables as look-up values.
3. Develop the SAS Macro programs that inherit the parameters being passed from the Excel file.

1: Common Parameters

a. Identify common settings

First, we identified common settings across all 120 sites:

1. There are four population types for each site: Student, Teacher, Parent, and Principal.
2. For each population, there are two surveys : Baseline and Follow-Up.
3. Generated reports have characteristic formats and functionality that remain consistent across all sites for a particular survey and population.
4. Reports for each site have consistent log and list outputs

b. Derive assumptions

Based on the common settings listed above, we derived the following assumptions for each site:

1. The folder structures will be consistent.
2. Folder paths will be specific for each site.
3. Input programs for each population across sites will be the same for the Baseline and Follow-Up rounds.
4. Programs for each report's output will be the same across each site.
5. Names of the reports themselves and the output SAS data sets will be similar with the specific site IDs and site names embedded in them.
6. The date we receive the data will be different for each instrument type for a site.
7. The date reports are created will be different for each instrument type for a site.

c. Folder structures

For the folder structure, we developed the following hierarchy for each site:

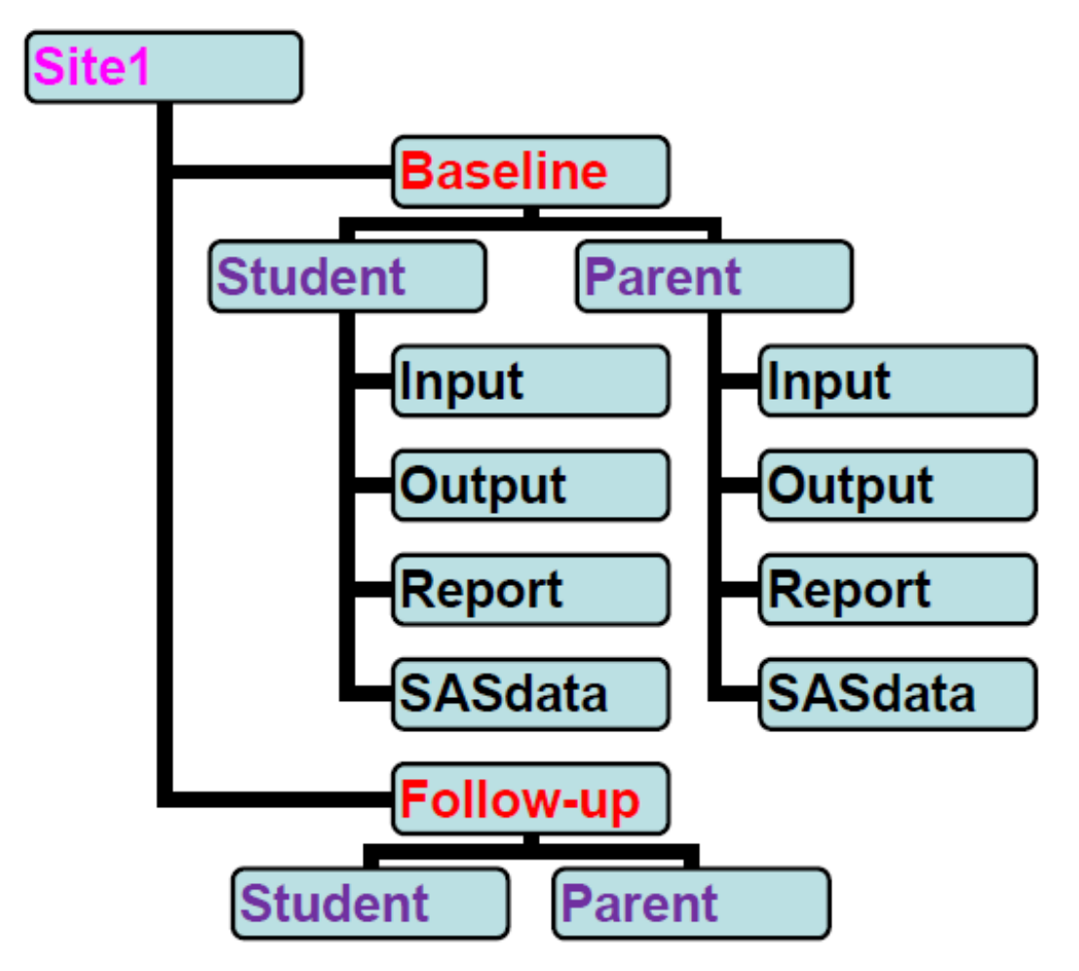


Figure 1. Reports and Output Folder Structure

Because the SAS programs are the same across all the sites for a particular instrument, we placed them in the general repository beneath the "SAS" folder

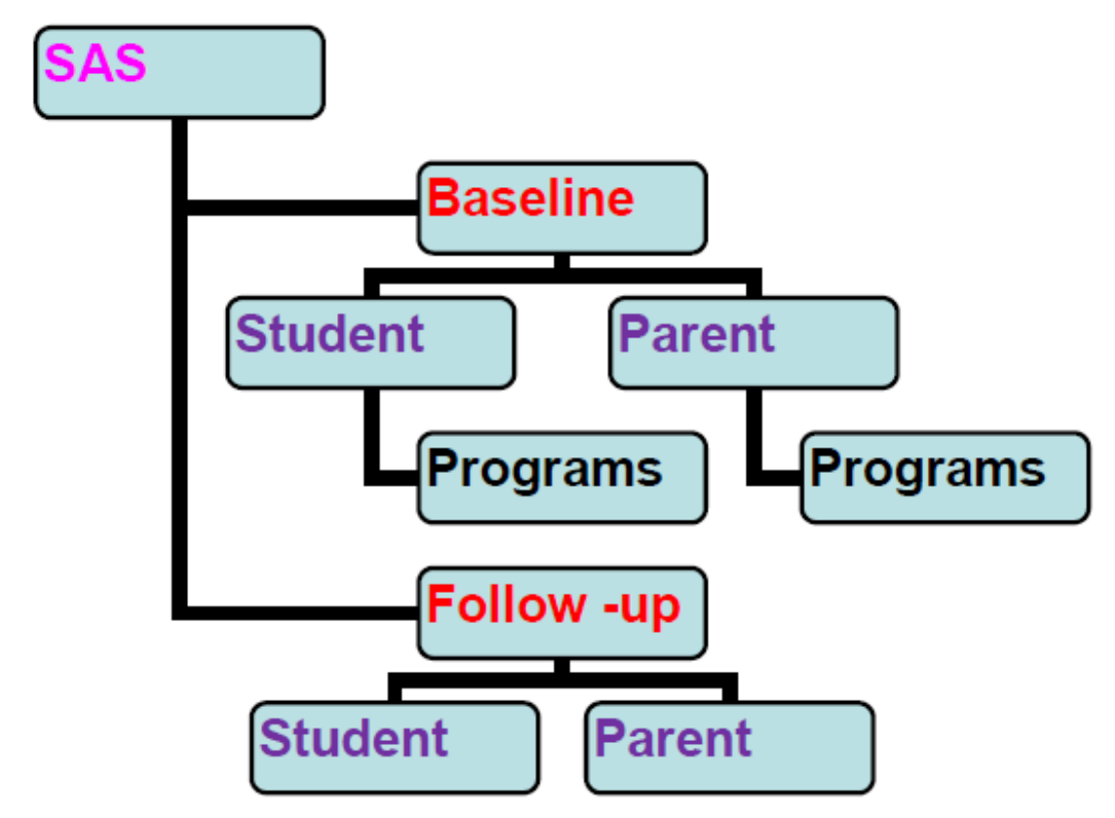
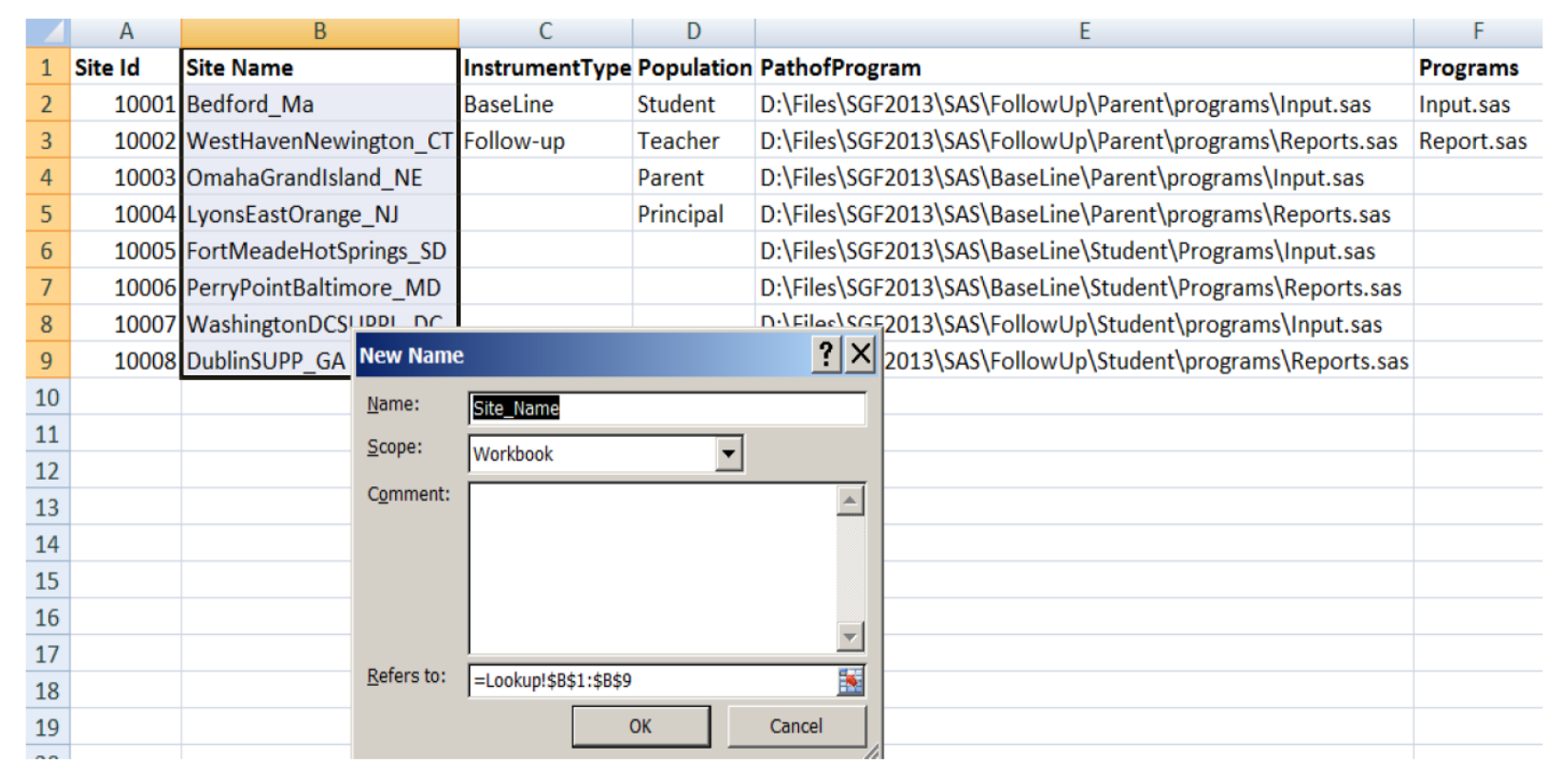


Figure 2. SAS Programs Folders for Each Instrument Type

2: Set up Excel File

a. Look-up tables

1. First create a list containing Site ID, Site Name, and Program names in the look-up sheet.
2. Give a name to each range of these look-up lists.



	A	B	C	D	E	F
1	Site Id	Site Name	InstrumentType	Population	PathofProgram	Programs
2	10001	Bedford_Ma	BaseLine	Student	D:\Files\SGF2013\SAS\FollowUp\Parent\programs\Input.sas	Input.sas
3	10002	WestHavenNewington_CT	Follow-up	Teacher	D:\Files\SGF2013\SAS\FollowUp\Parent\programs\Reports.sas	Report.sas
4	10003	OmahaGrandIsland_NE		Parent	D:\Files\SGF2013\SAS\BaseLine\Parent\programs\Input.sas	
5	10004	LyonsEastOrange_NJ		Principal	D:\Files\SGF2013\SAS\BaseLine\Parent\programs\Reports.sas	
6	10005	FortMeadeHotSprings_SD			D:\Files\SGF2013\SAS\BaseLine\Student\Programs\Input.sas	
7	10006	PerryPointBaltimore_MD			D:\Files\SGF2013\SAS\BaseLine\Student\Programs\Reports.sas	
8	10007	WashingtonDCSUPPL_DC			D:\Files\SGF2013\SAS\FollowUp\Student\programs\Input.sas	
9	10008	DublinSUPP_GA			2013\SAS\FollowUp\Student\programs\Reports.sas	

New Name
Name: Site_Name
Scope: Workbook
Comment:
Refers to: =Lookup!$B$1:$B$9

Figure 3. Look-Up Sheet with Name Range for Site Name Column

b. Main sheet for the users or programmers

In the Main sheet, we positioned drop-down cells for users to select the values and buttons to carry out particular actions using the SAS programs.

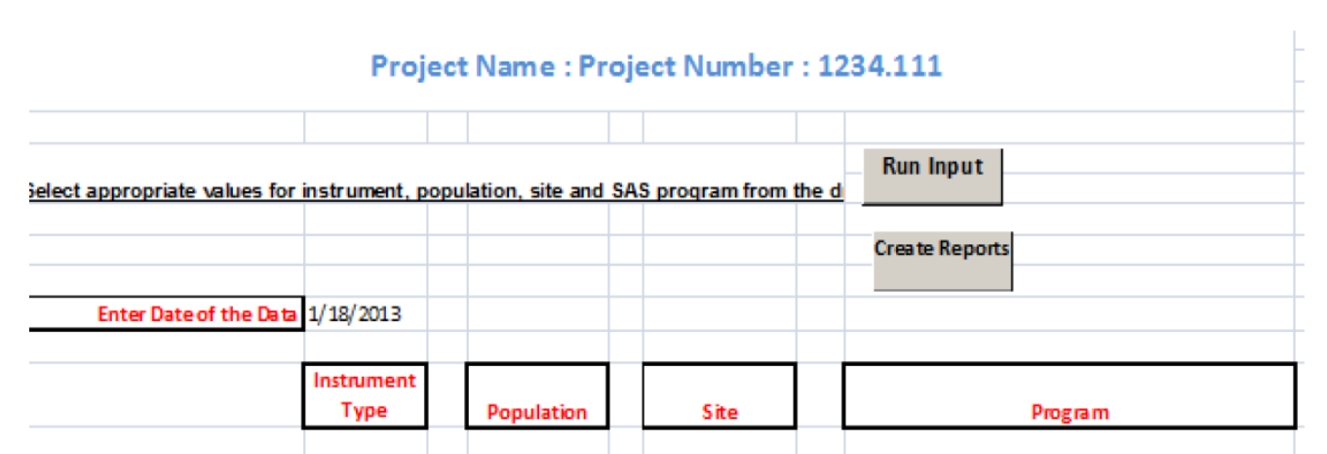
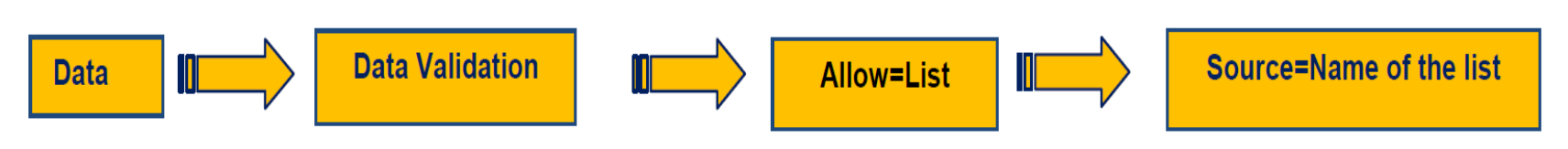
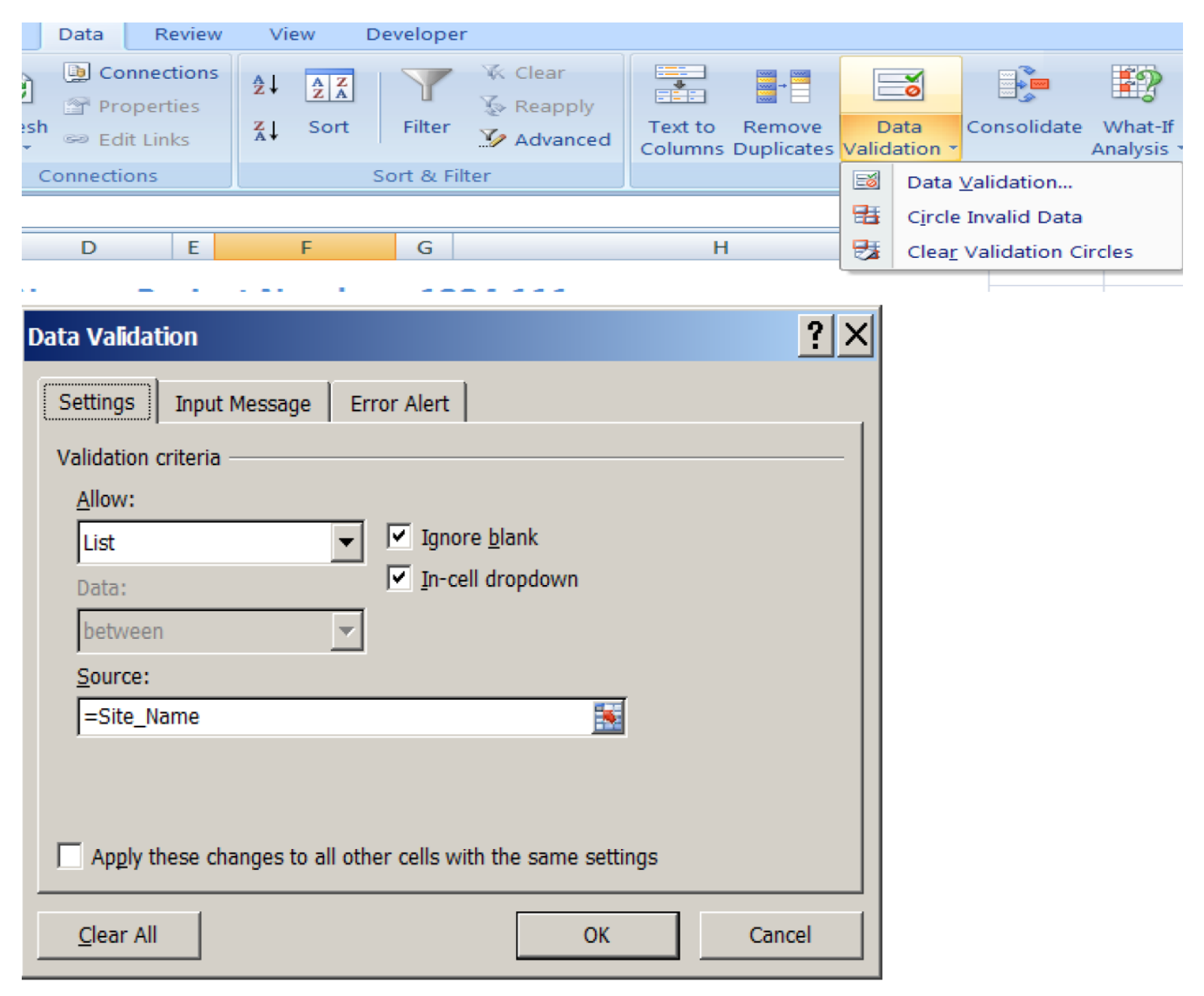

Project Name : Project Number : 1234.111
Select appropriate values for instrument, population, site and SAS program from the d
Run Input
Create Reports
Enter Date of the Data
1/18/2013
Instrument Type
Population
Site
Program

Figure 4. Main Screen Layout Design

c. Populate drop-down cells

To populate Site Name in the drop-down cells, select the Site textbox in the "Main" sheet and then select Data Validation from the Data menu. This will open the Data Validation window; then, select "list" from the Allow box and type "=*site_name" in the source box. "Site_name" is the name range we defined for the site list in the look-up sheet. Repeat these steps for all the other drop-down cells for the Population, Instrument Type, and Program cells.*





Data Validation
Settings | Input Message | Error Alert
Validation criteria
Allow: List
Data: between
[✓] Ignore blank
[✓] In-cell dropdown
Source: =Site_Name
[ ] Apply these changes to all other cells with the same settings
Clear All | OK | Cancel

Figure 5. Data Validation Pop-Up Window

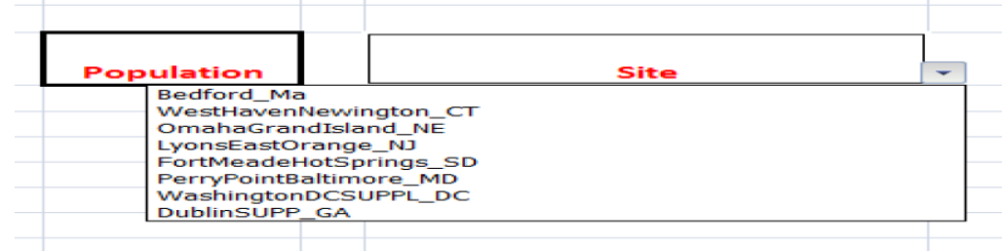
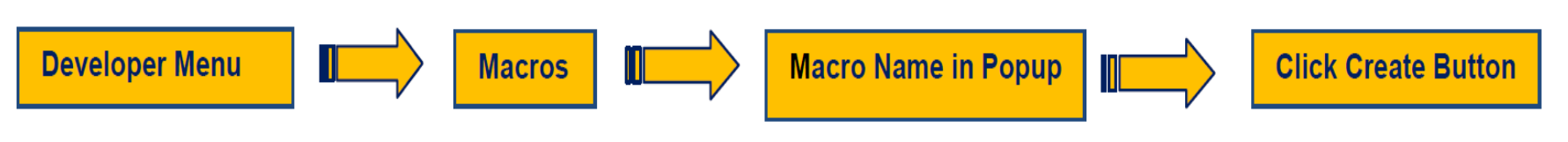
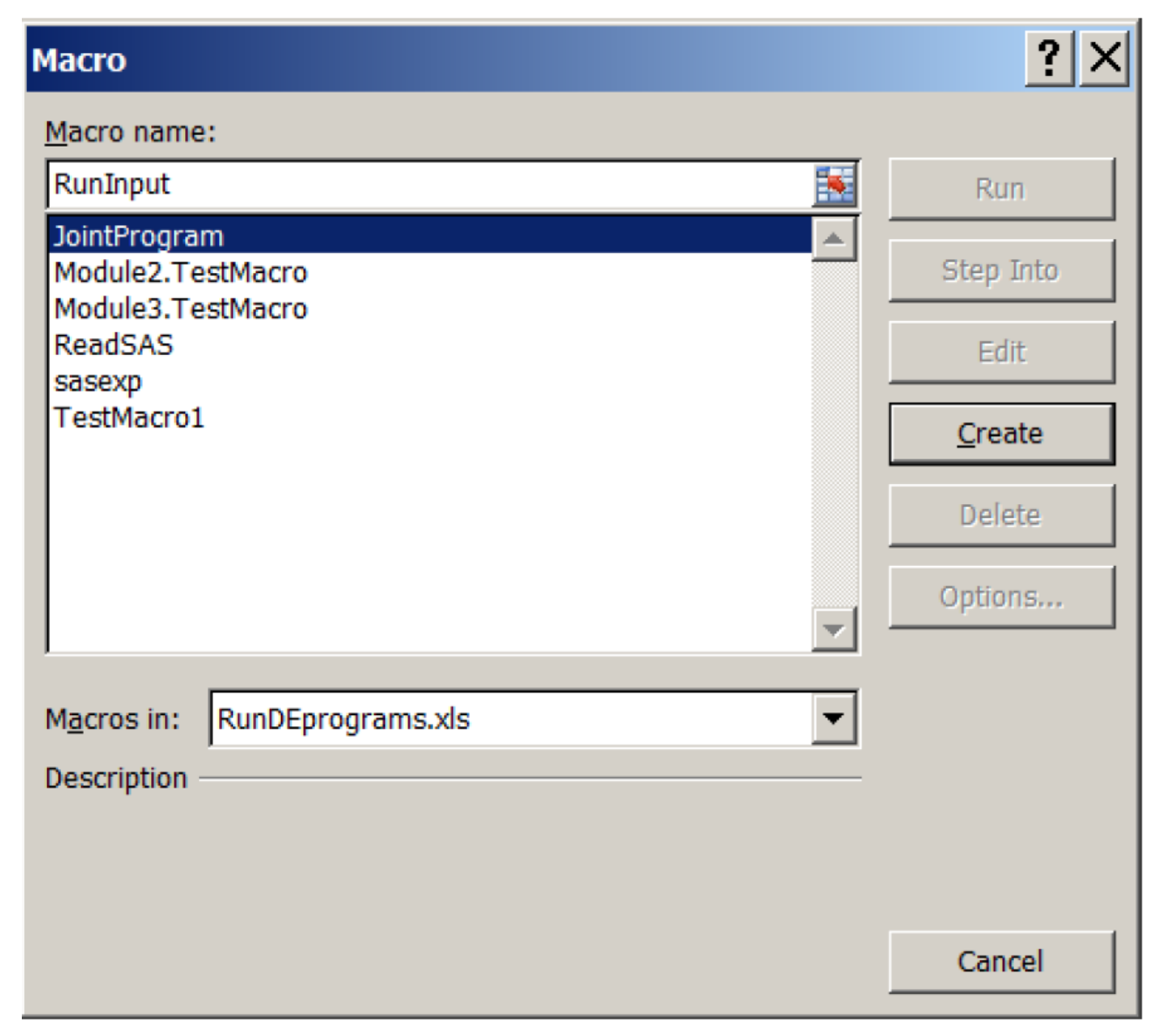


Figure 6. Populated Site Drop-Down

d. Record Excel Macros

The next step is to write the VBA macro that runs the SAS programs. The macro is recorded for the command button click action.





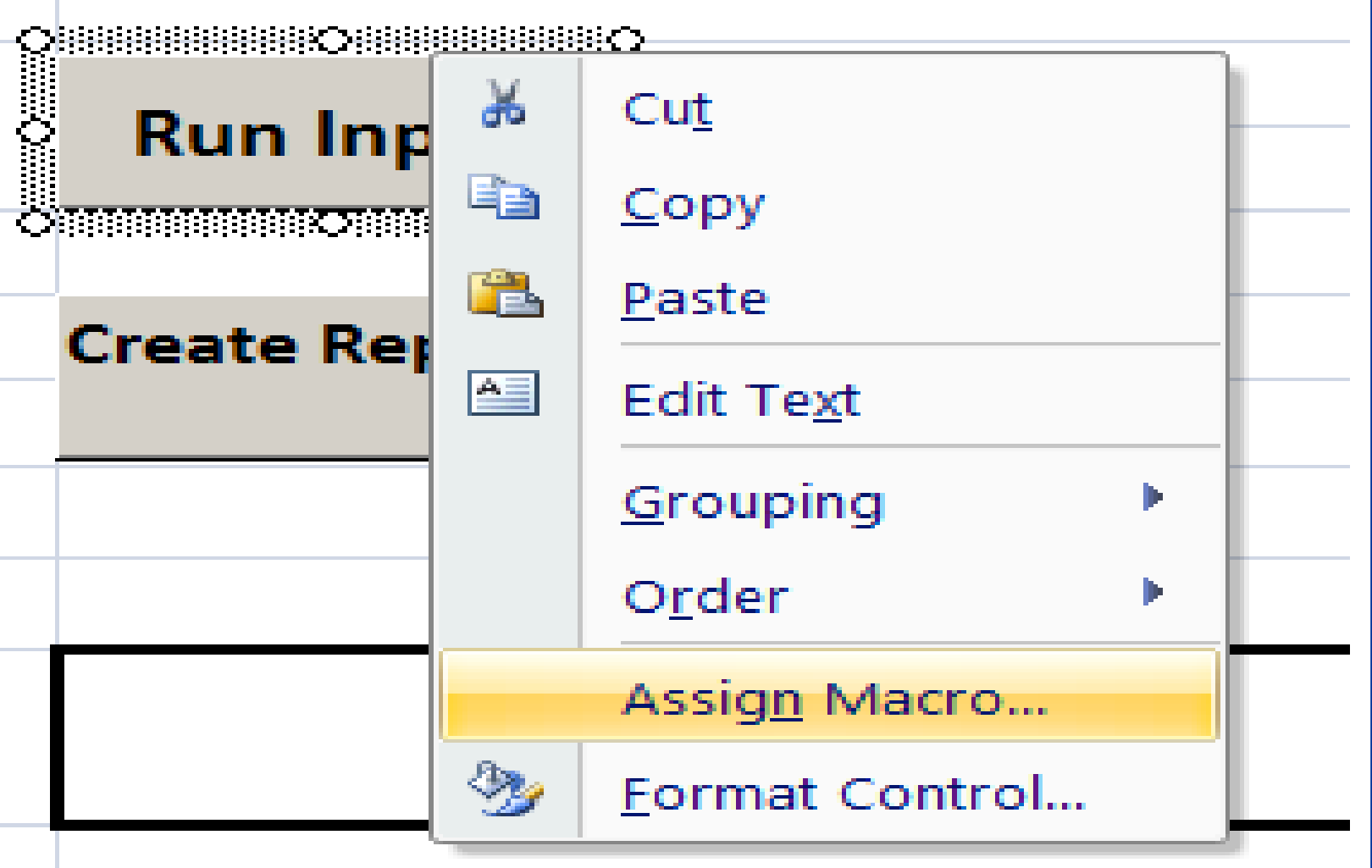
Macro
Macro name: RunInput
JointProgram
Module2.TestMacro
Module3.TestMacro
ReadSAS
sasexp
TestMacro1
Macros in: RunDEprograms.xls
Description
Run | Step Into | Edit | Create | Delete | Options... | Cancel

Figure 7. Create Macro from Excel Developer Menu

Macro RunInput code is written in the Microsoft Visual Basic Code window:

```
Sub RunInput()
Dim OleSAS As Object
'Invoke SAS
      Set OleSAS = CreateObject("SAS.Application")
      OleSAS.Visible = True
'Define %let used in the SAS programs
      OleSAS.Submit ("%Let Site=" & Sheets("Main").Range("F12").Value & ";")
      OleSAS.Submit ("%Let Instrument=" & Sheets("Main").Range("B12").Value & ";")
      OleSAS.Submit ("%let Population=" & Sheets("Main").Range("D12").Value & ";")
      OleSAS.Submit ("%let InProgram=" & Sheets("Main").Range("H12").Value & ";")
      OleSAS.Submit ("%let InDate=" & Sheets("Main").Range("B10").Value & ";")
'Include SAS program
      OleSAS.Submit ("data; %inc '&InProgram.'; run;")
      OleSAS.Quit
End Sub
```

e. Assign Excel macros to the command button



3. SAS Programs

```
ODS Listing;

Options ps=60 ls=200 nodate nocenter;

**Define directory path as discussed in step 1**;

%Let dirpath=D:\Files\SGF2013\&Site.\&Instrument\&Population.;

**SASDATA folder to place final data set **;

Libname dsn "&dirpath.\sasdata";

Proc PrintTo Log   ="&Dirpath.\output\&site.&Instrument.&population..log"  New;  Run;
Proc PrintTo Print  ="&Dirpath.\output\&site.&Instrument.&population..lst"  New;  Run;

%Let outfile1 = &site.&Instrument.&population.
%Let infile1   = &site.&Instrument.&population.&Indate..ds4;
%Let tdate     = %sysfunc(today(), mmddyy6.);

Title "&Site. - &population. &Instrument. : &tdate. ";

 Proc Format;
       Invalue DK (UPCASE JUST)
          'R'=.R    'D'=.D    'M'=.M    'N'=.N   'X'=.X;

 Data dsn.&outfile1.;

   Infile  "&dirpath.\input\&infile1." missover lrecl=705;

   Input

      ***your code***;

 Run;

 Proc Frequency Data=dsn.&outfile1.;
  Table _ALL_ /list missing;
Run;

%Include  "Clean_&Instrument.&population..sas";

%Include  "CrReport_&Instrument.&population..sas";
```

Conclusion

This application is an excellent example of how collaboration between SAS and Excel can result in an automated process for a complex task. We generated reports and SAS files for each site on demand and in turn-key mode by just having Excel and Base SAS on our machine. In order for this application to work, it is important to identify similarities and differences across all desired outputs and data sources. For any iterative data processing that does not require logic modifications to the SAS programs, this type of setup can be cost-effective and efficient.

Contact Information

Your comments and questions are valued and encouraged.
Contact the author at:
Name: Shilpa Khambhati
Company: Mathematica Policy Research
E-mail: SKhambhati@Mathematica-mpr.com

MATHEMATICA
Policy Research, Inc.