Paper RV-07

SAS® Code to Export and Create Pivot Tables in Excel 2007

By Robert Williams
Amerigroup Corporation, Virginia Beach, VA

ABSTRACT

Management and decision makers are always asking to have their report results in a pivot table in Excel 2007. Why? Because Excel is a widely used office software and pivot tables are popular among the business end users due to ease of drill-down capabilities of the data. Unfortunately, creating pivot tables is a manual process using a mouse. It becomes a chore when a SAS programmer is asked to create pivot tables in Excel 2007 using the data from SAS. In this paper, the step-by-step coding process will show you how SAS eliminates the manual process of creating pivot tables in Excel 2007. Using the Excel pivot table macro, PROC EXPORT and SAS Direct Data Exchange (DDE), SAS will link and communicate with Excel 2007 to automatically create the pivot tables without touching the mouse. You will be amazed, especially for routine weekly or monthly reports, how useful this process is for business reports that requires being in the management's preferred pivot table style in Excel 2007. Let SAS do the work of generating the Excel 2007 pivot tables for you!

INTRODUCTION

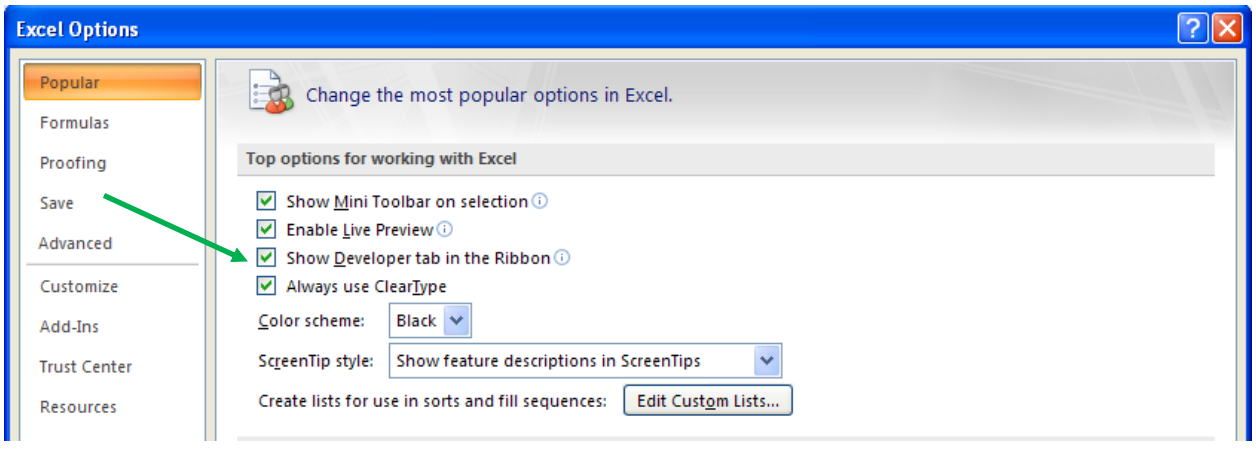
Pivot tables in Excel 2007 are becoming more popular with management and decision makers. They almost always ask for pivot tables as a deliverable for two reasons: one, it has drill down capabilities. Second, Pivot Tables are a user friendly hands-on report. Unfortunately, creating pivot tables is a menu driven process that requires us to use a mouse. As a SAS programmer, it becomes a chore to manually create pivot tables using the data generated from SAS. Not anymore, by using the Excel 2007 macros, PROC EXPORT and SAS Direct Data Exchange (DDE), we can automatically create pivot tables without touching the mouse. This process becomes a valuable tool for the routine business reports especially if you have to generate the same kind of reports on multiple subsets of data. Plus, we can make management happy to have the deliverables in their preferred pivot table style in Excel 2007.

In addition to SAS Base 9.2, this paper requires both Excel 2007 and SAS Access to PC Files to perform the following steps in SAS coding as well as one Excel 2007 step.

- Excel Step: Create Excel macro (VBA) for pivot table
- SAS Steps:
  - Step 1: Export data into Excel
  - Step 2: Open Excel file with the VBA macro
  - Step 3: Open Excel file with the exported SAS data
  - Step 4: Invoke VBA macro to create the pivot table
  - Step 5: Save Excel file with the newly created pivot table

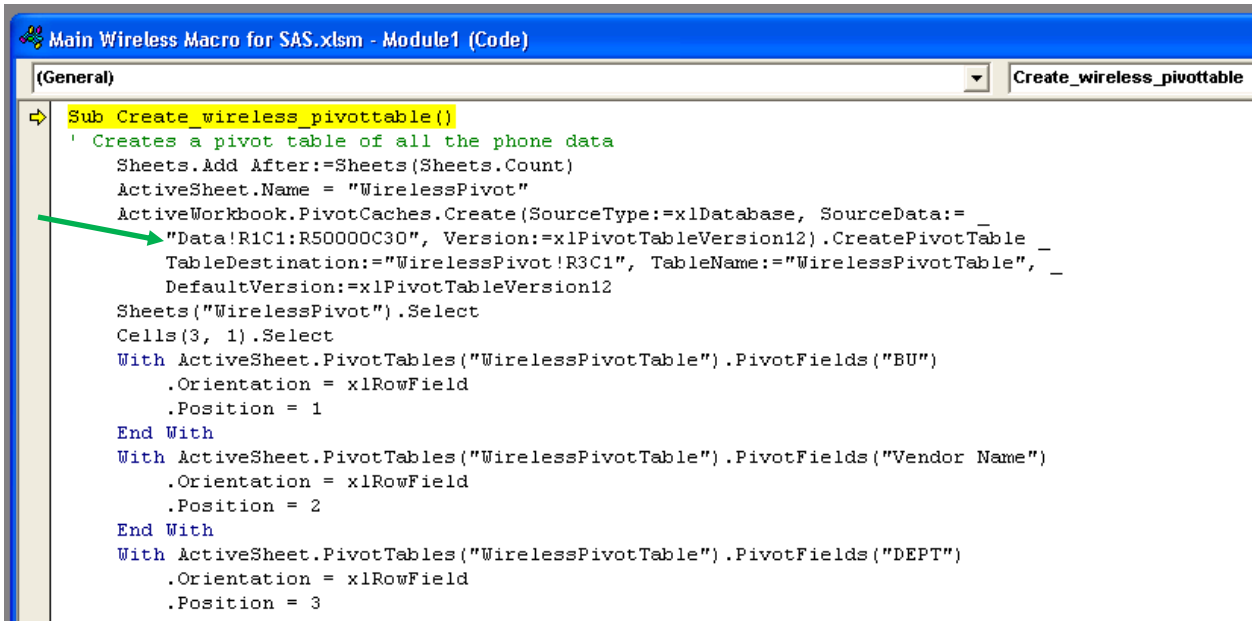
EXCEL 2007 STEP

In this step, we need to create an Excel macro that will perform a pivot table on the data. The easiest way to do this is to use the Excel macro recorder under the Developer menu. If you do not see the Developer menu, you just need to enable the "Show Developer tab in the Ribbon". It is under the Excel Options as shown below:



After you enable the Developer menu in Excel, export a sample data to Excel from SAS using PROC EXPORT. Highlight the sample data and then select Developer → Record Macro to start the process of recording the creation of pivot tables. In the Record Macro dialogue box, assign a name for the macro i.e. "Create_pivottable". Now, Excel is recording everything your mouse is doing. Proceed with the process of inserting and creating a pivot table on the sample data. Once you are finished with creating the pivot table the way you want it, select Developer → Stop Recording to end the process of recording the macro.

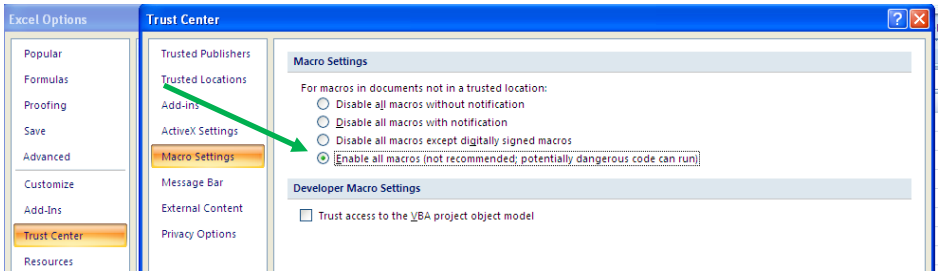
Now, we have recorded an Excel macro that recorded the process of creating a pivot table on the sample data. However, the sample data is typically smaller than the production data. With that in mind, we need to expand the data range to cover the largest production data you might encounter. The way to do this is to modify the VBA code that we just created. The VBA code for pivot tables looks like this. Note: how I manually changed the data rows to 50,000 i.e. modify from `R2100C30` to `R50000C30`. The 'R' stands for rows and 'C' stands for columns.



With the VBA macro recorded and modified, we save the Excel file with the VBA macro as a blank file (no data) to be used later in the SAS coding Step 2. The Excel file must be saved as an Excel macro-enabled workbook (must be .xlsm file extension).

MAKING SURE EXCEL MACRO ARE ENABLED

Since we need for SAS to open the Excel file with the VBA macro in SAS coding Step 2, we must make sure the Excel 2007 macro settings is set to "Enable all macros" in the Trust Center of the Excel Opens. To do that, go to Excel Options > Trust Center > Macro Settings and then click on the radio button to Enable all macros. Be cautious of the files you are opening as all macros are enabled from this point forward. Some macros can be malicious in nature and could harm your computer. Make sure you know the source of the documents that you are opening.



SAS STEPS

Now, we have the VBA code all set to create pivot tables in Excel 2007. We now proceed with the SAS steps. The SAS steps are in 5 steps as illustrated below.

SAS STEP 1: EXPORT DATA INTO EXCEL

Export the actual SAS into Excel 2007 file. There is more than one way to export data into Excel 2007. I usually use a simple PROC EXPORT to do the export like the code below. Be sure to use the same sheet name as the VBA macro has as the source data.

```
/* Export the data into Excel prior to doing the pivot table */
PROC EXPORT DATA=VASUG.sample_wireless
            OUTFILE= H:\SESUG Presentation\VASUG Wireless.xlsx"
            DBMS=EXCEL LABEL REPLACE;
      SHEET="Data";
RUN;
```

SAS STEP 2: OPEN EXCEL FILE WITH THE VBA MACRO

The first Excel file for SAS to open is the one with the VBA macro we created. The Excel file is opened by using the SAS 'x' command with the path and file name of the .xlsm file extension. Below is the example code.

```
/* Opens the Excel macro prior to establishing the DDE to run the Excel VB macro */
%let xlsMacro = H:\SESUG Presentation\Main Wireless Macro for SAS.xlsm;
options noxsync noxwait;
x "'&xlsMacro.'"; /* Opens the Excel macro file first */

data _null_;
      rc = sleep(2);
run;
```

Note, I added a **data** _null_ step to include a sleep() statement. The sleep() statement allows Excel time to fully open the Excel files before proceeding to the next statement, SAS data step or SAS procedure.

SAS STEP 3: OPEN EXCEL FILE WITH THE EXPORTED SAS DATA

It is important to open the Excel file with the actual data after opening the VBA macro file in Step 2. It must be opened in that order so the Excel file with the actual data becomes the active window and the Excel file with the VBA macro is in the background. Before opening the Excel file with the actual data, we have to set up the DDE in SAS. DDE stands for Direct Data Exchange. It allows SAS to control Excel commands. Once the DDE is set up, we can invoke the VBA macro on the exported data. Here is the sample SAS code for the DDE set up and opening the Excel file with the actual data.

```
filename sas2xl dde 'excel|system';
/* This data step opens the file with the data and runs */
/* the Excel macro to create the pivot table of the data */
data _null_;
   file sas2xl;
   put "[open(""H:\SESUG Presentation\VASUG Wireless.xlsx"", 0 , false)]";
   rc = sleep(2);
   put "[run(""'Main Wireless Macro for SAS.xlsm'!Create_wireless_pivottable"")]";
   put '[file.close(true)]';
run;
```

Note, I put another sleep() statement between opening the Excel file and running the VBA macro. Again, the sleep() statement allows Excel time to fully open the Excel files before proceeding to the next statement.

SAS STEP 4: INVOKE VBA MACRO TO CREATE THE PIVOT TABLE

To invoke the VBA macro or run other Excel commands, I use the SAS "put" statement in the above data _null_ step i.e.

```
   put "[run(""'Main Wireless Macro for SAS.xlsm'!Create_wireless_pivottable"")]";
```

The data _null_ is linked to the DDE under the "file" statement. This allows the DDE to run the VBA macro that is available in the background to create the pivot table.

SAS STEP 5: SAVE EXCEL FILE WITH THE NEWLY CREATED PIVOT TABLE

Finally at this point, we just created the pivot table of the exported data. We would want to save the newly created pivot table. Otherwise, we would do all this hard work for nothing, right? To save the work, we save the Excel file while still in the data _null_ step while under the DDE with this SAS statement above i.e.

```
   put '[file.close(true)]';
```

It closes the active Excel file while setting the Excel saved option as 'true'. At this point, Excel is still open with just one file which is the blank file with the VBA macro in it (the one with .xlsm extension). We close Excel with another data _null_ using the same link to the DDE under the "file" statement. Don't forget to put another sleep() statement before closing it. We want to make sure Excel has plenty of time to close and save the active file with the pivot table first.

```
/* Closes the Excel application */
data _null_;
       rc = sleep(5);
run;
data _null_;
       file sas2xl;
       put "[quit()]";
run;
```

CONCLUSION

Now, this paper has shown you how to create Pivot tables in Excel 2007 using SAS in six steps (one Excel VBA and five SAS steps). You will be able to quickly create many more pivot tables that are always popular with management and decision makers. With SAS, these pivot table steps can be repeated for production reports on a daily, weekly or monthly basis. Not only you can use it in a production environment, pivot tables can be performed on multiple subsets of data i.e. by business units, departments and etc. The process will become one of your most valuable tools to add to your SAS bag of coding tricks.

REFERENCES

Linking SAS and Microsoft Products by Nat Wooding, Dominion Virginia Power. Presented at Virginia SAS User's Group on June 17, 2009.

Step-by-Step in Using SAS DDE to Create an Excel Graph Based on N Observations from a SAS Data Set by Choon-Chern Lim, Mayo Clinic, Rochester, MN. Presented at SUGI 31

SAS Online Doc Version 9.2. Cary, NC: SAS Institute Inc.

Excel 2007 Online Help Microsoft Corp.

CONTACT INFORMATION

Your comments and questions are valued and encouraged. Contact the author at:

Robert Williams
Amerigroup Corporation
1300 Amerigroup Way
Virginia Beach, VA 23464
rwilli4@amerigroupcorp.com

SAMPLE SAS CODE

This is the full sample SAS code for this paper.

```
/* This is to export the wireless data and invoke */
/* the VB macro to create the pivot table on the exported data.  */
%let xlsdir = H:\SAS Papers\VASUG Spring 2010 Presentation;
libname VASUG "&xlsdir";

/* Export the data into Excel prior to doing the pivot table */
PROC EXPORT DATA=VASUG.sample_wireless
             OUTFILE= "&xlsdir.\VASUG Wireless.xlsx"
             DBMS=EXCEL LABEL REPLACE;
       SHEET="Data";
run;

/* Opens the Excel macro prior to establishing the DDE to run the Excel VB macro */
%let xlsMacro = &xlsdir\Main Wireless Macro for SAS.xlsm;
options noxsync noxwait;
x "'&xlsMacro.'"; /* Opens the Excel macro file first */
data _null_;
      rc = sleep(2);
run;
filename sas2xl dde 'excel|system';
/* This data step opens the file with the data and runs */
/* the Excel macro to create the pivot table of the data */
data _null_;
   file sas2xl;
   put "[open(""&xlsdir.\VASUG Wireless.xlsx"", 0 , false)]";
   rc = sleep(2);
   put "[run(""'Main Wireless Macro for SAS.xlsm'!Create_wireless_pivottable"")]";
   put '[file.close(true)]';
run;
/* Closes the Excel application */
data _null_;
      rc = sleep(5);
run;
data _null_;
      file sas2xl;
      put "[quit()]";
run;
```

SAMPLE EXCEL 2007 VBA MACRO CODE

This is the full sample Excel2007 VBA macro code for this paper.

```
Sub Create_wireless_pivottable()
' Creates a pivot table of all the phone data
    Sheets.Add After:=Sheets(Sheets.Count)
    ActiveSheet.Name = "WirelessPivot"
    ActiveWorkbook.PivotCaches.Create(SourceType:=xlDatabase, SourceData:= _
        "Data!R1C1:R50000C30", Version:=xlPivotTableVersion12).CreatePivotTable _
        TableDestination:="WirelessPivot!R3C1", TableName:="WirelessPivotTable", _
        DefaultVersion:=xlPivotTableVersion12
    Sheets("WirelessPivot").Select
    Cells(3, 1).Select
    With ActiveSheet.PivotTables("WirelessPivotTable").PivotFields("BU")
        .Orientation = xlRowField
        .Position = 1
    End With
    With ActiveSheet.PivotTables("WirelessPivotTable").PivotFields("Vendor Name")
        .Orientation = xlRowField
        .Position = 2
```

```
    End With
    With ActiveSheet.PivotTables("WirelessPivotTable").PivotFields("DEPT")
        .Orientation = xlRowField
        .Position = 3
    End With
    With ActiveSheet.PivotTables("WirelessPivotTable").PivotFields("Wireless Number")
        .Orientation = xlRowField
        .Position = 4
    End With
    With ActiveSheet.PivotTables("WirelessPivotTable").PivotFields("User Name")
        .Orientation = xlRowField
        .Position = 5
    End With
    With ActiveSheet.PivotTables("WirelessPivotTable").PivotFields("Equipment Make")
        .Orientation = xlRowField
        .Position = 6
    End With
    With ActiveSheet.PivotTables("WirelessPivotTable").PivotFields("Equipment Model")
        .Orientation = xlRowField
        .Position = 7
    End With
    With ActiveSheet.PivotTables("WirelessPivotTable").PivotFields("Type of Service")
        .Orientation = xlRowField
        .Position = 8
    End With

    ActiveSheet.PivotTables("WirelessPivotTable").AddDataField
ActiveSheet.PivotTables( _
        "WirelessPivotTable").PivotFields("Total Current Charges"), _
        "Total Current Charge", xlSum
    ActiveSheet.PivotTables("WirelessPivotTable").AddDataField
ActiveSheet.PivotTables( _
        "WirelessPivotTable").PivotFields("Monthly Voice & Data Charge"), _
        "Monthly Voice/Data Access Charge", xlSum
    ActiveSheet.PivotTables("WirelessPivotTable").AddDataField
ActiveSheet.PivotTables( _
        "WirelessPivotTable").PivotFields("Additional Airtime Charge"), _
        "Additional Airtime Charges", xlSum
    ActiveSheet.PivotTables("WirelessPivotTable").AddDataField
ActiveSheet.PivotTables( _
        "WirelessPivotTable").PivotFields("Total Data Useage Charge (Excluding
Roaming)") _
        , "Data Usage Charges (Excluding Roaming)", xlSum
    ActiveSheet.PivotTables("WirelessPivotTable").AddDataField
ActiveSheet.PivotTables( _
        "WirelessPivotTable").PivotFields("Additional Feature Charge"), _
        "Additional Feature Charges", xlSum
    ActiveSheet.PivotTables("WirelessPivotTable").AddDataField
ActiveSheet.PivotTables( _
        "WirelessPivotTable").PivotFields("Total Equipment Charge"), _
        "Equipment Charges", xlSum
    ActiveSheet.PivotTables("WirelessPivotTable").AddDataField
ActiveSheet.PivotTables( _
        "WirelessPivotTable").PivotFields("Long Distance Charge"), _
        "LD Charges", xlSum
    ActiveSheet.PivotTables("WirelessPivotTable").AddDataField
ActiveSheet.PivotTables( _
        "WirelessPivotTable").PivotFields("Directory Assistance Charge (411 calls)" _
        ), "411 Assistance Charge", xlSum
    ActiveSheet.PivotTables("WirelessPivotTable").AddDataField
ActiveSheet.PivotTables( _
        "WirelessPivotTable").PivotFields("Total Roaming Charge"), _
        "Roaming Charges", xlSum
```

```
    ActiveSheet.PivotTables("WirelessPivotTable").AddDataField
ActiveSheet.PivotTables( _
        "WirelessPivotTable").PivotFields("Service Level Other Charge"), _
        "Other Service Level Charges", xlSum
    ActiveSheet.PivotTables("WirelessPivotTable").AddDataField
ActiveSheet.PivotTables( _
        "WirelessPivotTable").PivotFields("Total Taxes Surcharges and Regulatory
Fees"), _
        "Taxes Surcharges and Regulatory Fees", xlSum
    ActiveSheet.PivotTables("WirelessPivotTable").AddDataField
ActiveSheet.PivotTables( _
        "WirelessPivotTable").PivotFields("Total Miscellaneous Usage Charge"), _
        "Miscellaneous Usage Charge", xlSum
    ActiveSheet.PivotTables("WirelessPivotTable").AddDataField
ActiveSheet.PivotTables( _
        "WirelessPivotTable").PivotFields("Total Non-Communications Charge"), _
        "Non-Communications Charge", xlSum
    ActiveSheet.PivotTables("WirelessPivotTable").AddDataField
ActiveSheet.PivotTables( _
        "WirelessPivotTable").PivotFields("Total Messaging Charge"), _
        "Messaging Charges", xlSum
    ActiveSheet.PivotTables("WirelessPivotTable").AddDataField
ActiveSheet.PivotTables( _
        "WirelessPivotTable").PivotFields("Total Number of Events"), _
        "Total Count of Messages", xlSum
    ActiveSheet.PivotTables("WirelessPivotTable").AddDataField
ActiveSheet.PivotTables( _
        "WirelessPivotTable").PivotFields("Total voice Minutes of Use (MOU)"), _
        "Voice Minutes of Use (MOU)", xlSum
    ActiveSheet.PivotTables("WirelessPivotTable").AddDataField
ActiveSheet.PivotTables( _
        "WirelessPivotTable").PivotFields("Total KB Data Usage"), _
        "Data Usage (Kilobytes)", xlSum
    ActiveSheet.PivotTables("WirelessPivotTable").TableStyle2 = "PivotStyleMedium9"
    ActiveSheet.PivotTables("WirelessPivotTable").RowAxisLayout xlTabularRow
    ' Expand to show all details
    ActiveSheet.PivotTables("WirelessPivotTable").PivotFields("BU"). _
        ShowDetail = True
    ' Removes the subtotals for wireless numbers, User Name,Equipment Make and type of
service
    ActiveSheet.PivotTables("WirelessPivotTable").PivotFields( _
        "Wireless Number").Subtotals = Array(False, False, False, False, False, False,
_
        False, False, False, False, False, False)
    ActiveSheet.PivotTables("WirelessPivotTable").PivotFields("User Name"). _
        Subtotals = Array(False, False, False, False, False, False, False, False,
False, False, _
        False, False)
    ActiveSheet.PivotTables("WirelessPivotTable").PivotFields("Equipment Make"). _
        Subtotals = Array(False, False, False, False, False, False, False, False,
False, False, _
        False, False)
    ActiveSheet.PivotTables("WirelessPivotTable").PivotFields("Type of Service"). _
        Subtotals = Array(False, False, False, False, False, False, False, False,
False, False, _
        False, False)
    ActiveSheet.PivotTables("WirelessPivotTable").PivotFields("Equipment Model"). _
        Subtotals = Array(False, False, False, False, False, False, False, False,
False, False, _
        False, False)
    ActiveSheet.PivotTables("WirelessPivotTable").PivotFields("Dept"). _
        Subtotals = Array(False, False, False, False, False, False, False, False,
False, False, _
```

```
        False, False)
    ActiveSheet.PivotTables("WirelessPivotTable").PivotFields("Vendor Name"). _
        Subtotals = Array(False, False, False, False, False, False, False, False,
False, False, _
        False, False)
    ' Formats the dollars amounts
    Columns("I:V").Select
    Selection.NumberFormat = "$#,##0.00"
    ' Formats the number with thousands separator
    Columns("X:Y").Select
    Selection.NumberFormat = "#,##0"
    ' Adjust the column widths
    Columns("A:A").Select
    Selection.ColumnWidth = 12
    Columns("B:B").Select
    Selection.ColumnWidth = 18
    Columns("C:C").Select
    Selection.ColumnWidth = 10
    Columns("D:D").Select
    Selection.ColumnWidth = 15
    Columns("E:E").Select
    Selection.ColumnWidth = 25
    Columns("F:F").Select
    Selection.ColumnWidth = 14
    Columns("G:G").Select
    Selection.ColumnWidth = 14
    Columns("H:H").Select
    Selection.ColumnWidth = 14
    Columns("I:AA").Select
    Selection.ColumnWidth = 15
    ' Formats the header to wrap text
    Rows("4:4").Select
    With Selection
        .HorizontalAlignment = xlGeneral
        .VerticalAlignment = xlBottom
        .WrapText = True
        .Orientation = 0
        .AddIndent = False
        .IndentLevel = 0
        .ShrinkToFit = False
        .ReadingOrder = xlContext
        .MergeCells = False
    End With

    Range("A1").Select
    ActiveWindow.Zoom = 85
    With ActiveSheet.PageSetup
        .PrintTitleRows = "$4:$4"
        .PrintTitleColumns = ""
    End With
    ActiveSheet.PageSetup.PrintArea = ""
    With ActiveSheet.PageSetup
         .CenterHeader = "&A"
        .LeftMargin = Application.InchesToPoints(0.2)
        .RightMargin = Application.InchesToPoints(0.2)
        .Orientation = xlLandscape
        .PaperSize = xlPaperLetter
        .Zoom = False
        .FitToPagesWide = 1
        .FitToPagesTall = 999
    End With

End Sub
```