Paper 1060-2017

Snail Mail to Emails: Generating Auto-Emails from SAS® with Attachments

Crystal Carel, MPH, Baylor Scott & White Health, Dallas, TX

ABSTRACT

Do you get tired of constantly creating new emails each and every time you run a report, frantically searching for the reports, attaching said reports, and writing emails while thinking there has to be a better way? Then have I got some code to share with you! This session will provide you with code to flee your old ways of emailing data and reports and instead have you set up your SAS® code to send an email to your recipients that has attached the most current file(s) each and every time the code is ran without having to manually do anything after you run your SAS® code.

This session will provide a SAS® programmer with instruction on how to create your own email that is in a macro that is based on your current report. We will demonstrate the different options available to tailor the code to add your body (and change the body with macros), add attachments (PDF, Excel), and an additional macro in place to check if a file is present and will put a note in the SAS® log if missing so you won't get a warning message. Using the SAS® code provided will help you become more efficient and effective by automating a tedious process and reducing errors in email attachments, wording, and recipient lists.

INTRODUCTION

You have a report or output that you run on a schedule (daily, weekly, monthly) and then email the results. Having to manually create an email, add attachments, and include recipients is not only time consuming but also causes stress (especially if you accidentally forget to add an attachment or recipient). It is vital to have an automated distribution process for reports that occur on a regular basis. Understanding how to manipulate SAS® code and attached different outputs while keeping the SAS® log free of warnings is imperative to have a seamless process and hands off report.

We will walk through sending an email from SAS®, attaching various file types (.xlsx, .pdf, .sas, .docx, .sas7bdat, .csv) to an email, and adding code to check if an attachment isn't present as to not error, instead displaying a note in the SAS® log. Finally, we will discuss how to include macros in the process. This gives you the ability to parse your reports in multiple ways and allowing different reports to go to different recipients. This paper is using SAS® 9.4 via SAS® EG 7.1, tested with Microsoft Outlook 2013 and Gmail.

To Note: The email environment may need to be set up via your SAS® System Administrator before you are able to send emails. See *References/Recommended Readings* for assistance.

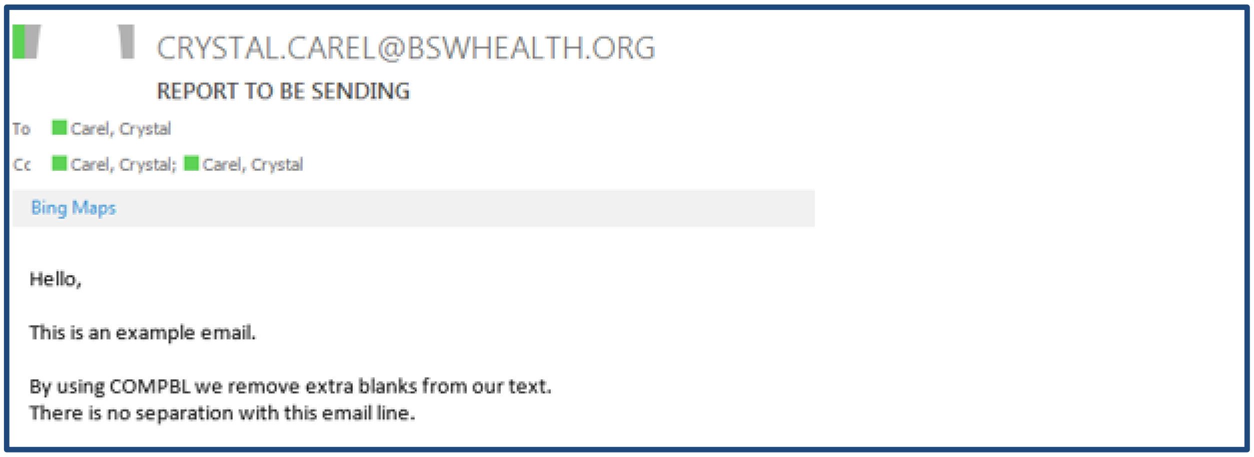
SENDING AN EMAIL FROM SAS® CODE

To get started, use the code below and copy/paste into a SAS® session and see how it works (Figure 1).

Note*: Please change the email recipient to your own email address.*

```
    FILENAME OUTBOX EMAIL
       FROM    = ("CRYSTAL.CAREL@BSWHEALTH.ORG")
       TO      = ("CRYSTAL.CAREL@BSWHEALTH.ORG")
       CC      = ("CRYSTAL.CAREL@BSWHEALTH.ORG","CRYSTAL.CAREL@BSWHEALTH.ORG")
       REPLYTO = ("CRYSTAL.CAREL@BSWHEALTH.ORG")
       SUBJECT = ("REPORT TO BE SENDING");
      DATA _NULL_;
       FILE OUTBOX;
          PUT "Hello,";
          PUT ;
          PUT %SYSFUNC(COMPBL("This is an example email."));
          PUT ;
          PUT %SYSFUNC(COMPBL("By using COMPBL we remove extra blanks from our text."));
          PUT %SYSFUNC(COMPBL("There is no separation with this email line."));
       RUN;
    FILENAME OUTBOX CLEAR;
```

Figure 1: Example of email displayed using SAS® in Microsoft Outlook 2013


CRYSTAL.CAREL@BSWHEALTH.ORG

REPORT TO BE SENDING

To Carel, Crystal

Cc Carel, Crystal; Carel, Crystal

Hello,

This is an example email.

By using COMPBL we remove extra blanks from our text.
There is no separation with this email line.

BREAKDOWN OF THE EMAIL

Congrats! You sent an email, but what do all the pieces mean and do you always need all the pieces?

To better explain what the email is doing, here's the break down step by step:

- ***FILENAME OUTBOX EMAIL –*** Specifies the filename and allows emails by SMTP (Simple Mail Transfer Protocol) email interface.
  - o Syntax:: FILENAME fileref EMAIL <'address' ><email-options>;
- ***FROM/TO/CC/BCC/REPLYTO –*** All are email addresses only.
  - o Fields always have to be in "quotes" (single or double).
  - o Only needed field is **TO**; other fields can be utilized but are not necessary.
  - o **FROM** – If used, should only include 1 email. Can use a different email to mask the sender. Helpful if the stakeholder wants the report to look like it's coming from their own email and not from the programmer.
  - o **TO/CC/REPLYTO** – List of recipient emails separated by comma (,).
  - o **BCC** – (*not shown but can include*) List of recipient emails separated by comma (,). Email addresses will not be seen when distributed.
- ***SUBJECT*** – Can call macros in the SUBJECT line (will show later in paper).
- ***DATA _NULL_*** – Processes the statements to follow without creating a dataset.
- ***FILE OUTBOX*** – Specifics where the output for the **PUT** statements and the device it will go to.
- ***PUT*** – Write lines selected in the **FILE** statement. **PUT;** adds extra blank lines to break up email.
- ***PUT %SYSFUN(COMPBL*** – Same as **PUT** above, but also compresses extra blanks.
- ***FILENAME OUTBOX CLEAR*** – Good practice to wrap up the code and clear fileref assigned.

ADDING ATTACHMENTS TO EMAILS

You need to add attachments? Who doesn't? This is where the code gets handy.

Attachments can be set as **%LET** macros or they can placed in the **ATTACH=.** The code below shows both ways by using **%LET** (Attach1, Attach2, Attach3) and placing the files in the **ATTACH=** statement (Attach4, Attach5, Attach6).

For this paper, the file names are presented with **%LET** statements so we can use later on to check for multiple files – it may be that you want the files that are available emailed, even though some files don't exist.

***Note**: Please change the email recipient to your own email address & change directory to find your files.*

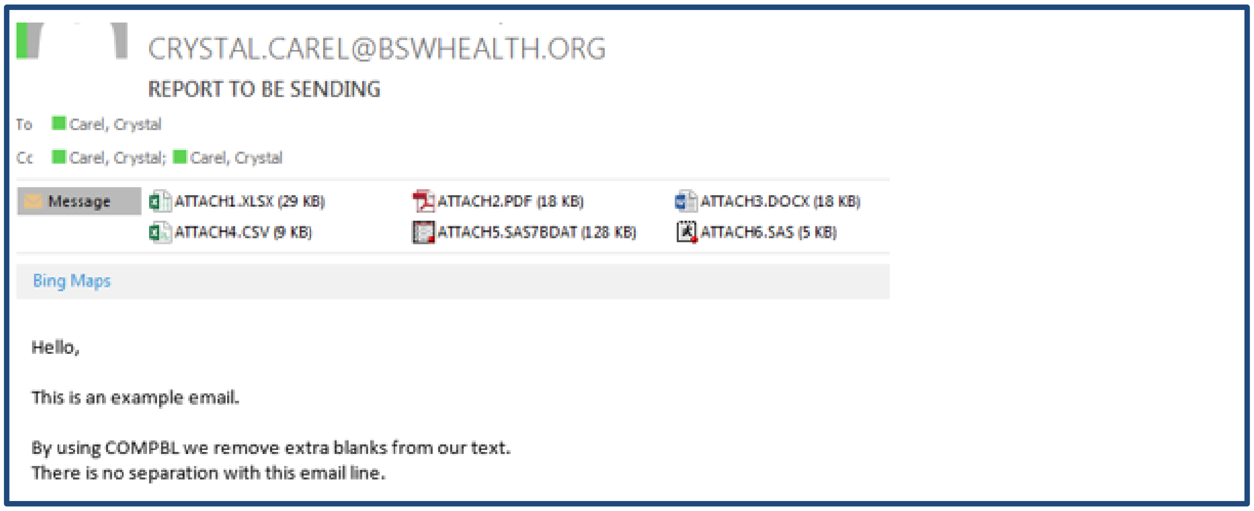
```
%LET ATTACH1=%STR(C:\Users\EXAMPLE\ATTACH1.XLSX);
%LET ATTACH2=%STR(C:\Users\EXAMPLE\ATTACH2.PDF);
%LET ATTACH3=%STR(C:\Users\EXAMPLE\ATTACH3.DOCX);

      FILENAME OUTBOX EMAIL
        FROM    = ("CRYSTAL.CAREL@BSWHEALTH.ORG")
        TO      = ("CRYSTAL.CAREL@BSWHEALTH.ORG")
        CC      = ("CRYSTAL.CAREL@BSWHEALTH.ORG","CRYSTAL.CAREL@BSWHEALTH.ORG")
        REPLYTO = ("CRYSTAL.CAREL@BSWHEALTH.ORG")
        SUBJECT = ("REPORT TO BE SENDING")
        ATTACH  = ("&ATTACH1."                  CT="APPLICATION/MSEXCEL" EXT="XLSX"
                   "&ATTACH2."                  CT="APPLICATION/PDF"     EXT="PDF"
                   "&ATTACH3."                  CT="APPLICATION/DOCX"    EXT="DOCX"
                   "C:\Users\EXAMPLE\ATTACH4." CT="APPLICATION/MSEXCEL" EXT="CSV"
                   "C:\Users\EXAMPLE\ATTACH5." CT="APPLICATION/SAS"     EXT="SAS7BDAT"
                   "C:\Users\EXAMPLE\ATTACH6." CT="APPLICATION/SAS"     EXT="SAS");

      DATA _NULL_;
       FILE OUTBOX;
          PUT "Hello,";
          PUT ;
          PUT %SYSFUNC(COMPBL(
              "This is an example email."));
          PUT ;
          PUT %SYSFUNC(COMPBL(
              "By using COMPBL we remove extra blanks from our text."));
          PUT %SYSFUNC(COMPBL(
              "There is no separation with this email line."));
       RUN;
    FILENAME OUTBOX CLEAR;
```

The important part to remember when adding attachments is to include the *Content Type* (**CT=**) and the *Extension* (**EXT=**). If you forget to add these, your code may spit out an error message and you'll spend more time than it's worth only to find out that it can be avoided by adding the **CT=** and **EXT=.**

Figure 2: Example of email with attachments displayed using SAS® in Microsoft Outlook 2013


CRYSTAL.CAREL@BSWHEALTH.ORG

REPORT TO BE SENDING

To Carel, Crystal

Cc Carel, Crystal; Carel, Crystal

Message | ATTACH1.XLSX (29 KB) | ATTACH2.PDF (18 KB) | ATTACH3.DOCX (18 KB) | ATTACH4.CSV (9 KB) | ATTACH5.SAS7BDAT (128 KB) | ATTACH6.SAS (5 KB)

Bing Maps

Hello,

This is an example email.

By using COMPBL we remove extra blanks from our text.
There is no separation with this email line.

ADDING MACRO TO FIND IF FILE EXISTS

What if we need this process to let us know if a file does not exist without causing a warning message? We can do that by adding in a macro to check and see if the file is available before the email is sent. So if no file, no email. The SAS® log can be used to document this.

In this step, we add a macro called **FINDMYFILE** to check and see if the file is available. If the file does not exist, a NOTE will appear in green in the SAS® log. We do this by having attachments as **%LET** statements as we need these outside of the macro.

We used '*and*' looking for both files – but if you have multiple files to be sent regardless of the other files being present, you can manipulate the code to use for that situation (you can use '*or*' instead). This is where having the **%LET** macros come in handy outside of the bigger **FINDMYFILE** macro.

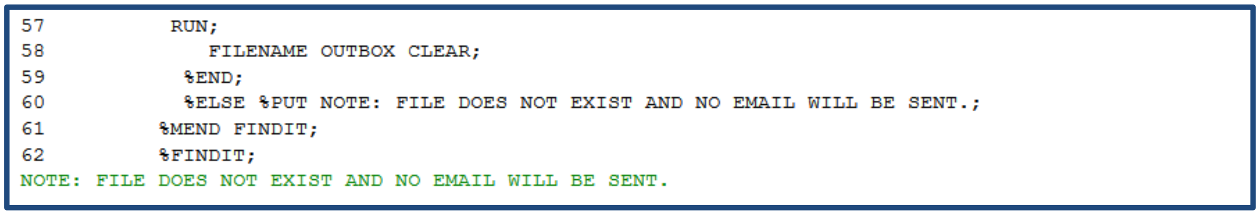
***Note**: Please change the email recipient to your own email address & change directory to find your files.*

```
%LET ATTACH1=%STR(C:\Users\EXAMPLE\ATTACH1.XLSX);
%LET ATTACH2=%STR(C:\Users\EXAMPLE\ATTACH2.PDF);
%MACRO FINDMYFILE;
    %IF %SYSFUNC(FILEEXIST(&ATTACH1))  AND   %SYSFUNC(FILEEXIST(&ATTACH2)) %THEN %DO;
    FILENAME OUTBOX EMAIL
        FROM    = ("CRYSTAL.CAREL@BSWHEALTH.ORG")
        TO      = ("CRYSTAL.CAREL@BSWHEALTH.ORG")
        CC      = ("CRYSTAL.CAREL@BSWHEALTH.ORG","CRYSTAL.CAREL@BSWHEALTH.ORG")
        REPLYTO = ("CRYSTAL.CAREL@BSWHEALTH.ORG")
        SUBJECT = ("REPORT TO BE SENDING")
        ATTACH  = ("&ATTACH1." CT="APPLICATION/MSEXCEL" EXT="XLSX"
                   "&ATTACH2." CT="APPLICATION/PDF"     EXT="PDF");
      DATA _NULL_;
       FILE OUTBOX;
          PUT "Hello,";
          PUT ;
          PUT %SYSFUNC(COMPBL(
               "This is an example email."));
          PUT ;
          PUT %SYSFUNC(COMPBL(
               "By using COMPBL we remove extra blanks from our text."));
          PUT %SYSFUNC(COMPBL(
               "There is no separation with this email line."));
       RUN;
    FILENAME OUTBOX CLEAR;
  %END;
  %ELSE %PUT NOTE: FILE DOES NOT EXIST AND NO EMAIL WILL BE SENT.;
%MEND FINDMYFILE;
%FINDMYFILE;
```

Now go back to the code and change the **%LET** statement for ATTACH1 file name to a file you do not have so we can see what happens when there is a file we know that does not exist (Figure 3).

```
%LET ATTACH1=%STR(C:\Users\EXAMPLE\ATTACHDONOTHAVE.XLSX);
%LET ATTACH2=%STR(C:\Users\EXAMPLE\ATTACH2.PDF);
%MACRO FINDMYFILE;
    %IF %SYSFUNC(FILEEXIST(&ATTACH1)) and %SYSFUNC(FILEEXIST(&ATTACH2)) %THEN %DO;………
```

Figure 3: Example of the SAS® log when file that does not exist and using FINDMYFILE macro



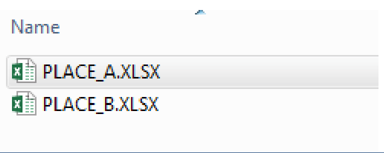
```
            RUN;
               FILENAME OUTBOX CLEAR;
             %END;
             %ELSE %PUT NOTE: FILE DOES NOT EXIST AND NO EMAIL WILL BE SENT.;
           %MEND FINDIT;
           %FINDIT;
NOTE: FILE DOES NOT EXIST AND NO EMAIL WILL BE SENT.
```

MACROTIZING CODE TO SEND MULTIPLE ATTACHMENTS FROM MULTIPLE PLACES

Now that we can send an email, add on one or multiple attachments, and check within the code to see if a file exists – we are going to stretch way out. Next, we'll use what we've learned to see how we could further expand the use of this code for multiple outputs with more personalization in the email.

Before we run the macro, look at the 2 files and the names of the files. These will be important as we walk through the macro.

<u>Figure 4: Example of the 2 files and names: PLACE_A & PLACE_B</u>



You may have multiple various outputs the need to be sent. You can use this macro to stratify by a variable such as location (which is the example shown here). Whatever way you divvied up your reports is fine as long as you have output for each individual file you want to send. We are only sending emails and not manipulating any data or creating any reports.

We started with a macro called **FILEMYFILE**. Then we added another macro wrapping around **FILEMYFILE** called **EMAILS**. By adding the macro **EMAILS**, we can now send recipients certain files – as we don't want Place A to get Place B files and vice versa.

We've added macros for NAME, LOCATION, and TOSEND. These are our changing variables. To call the macros, we call it by the '&' (ampersand), followed by VARIABLE NAME, and then a '.' (period) so (*&VARIABLE.*). Notice the **%LET** statement for LOCATION – there are 2 periods to resolve the macro.

<u>Note</u>*<u>: Please change the email recipient to your own email address & change directory to find your files.</u>*

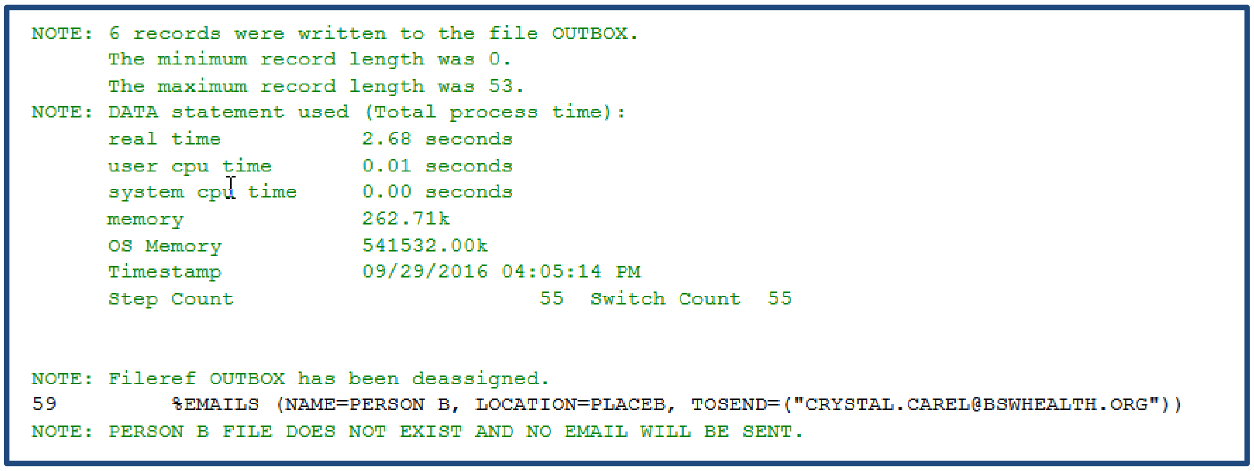
```
%MACRO EMAILS (NAME=, LOCATION=, TOSEND=);
%LET ATTACH1=%STR(C:\Users\EXAMPLE\&LOCATION..XLSX);
%MACRO FINDMYFILE;
    %IF %SYSFUNC(FILEEXIST(&ATTACH1)) %THEN %DO;

    FILENAME OUTBOX EMAIL
        FROM    = ("CRYSTAL.CAREL@BSWHEALTH.ORG")
        TO      = ("CRYSTAL.CAREL@BSWHEALTH.ORG")
        CC      = ("CRYSTAL.CAREL@BSWHEALTH.ORG","CRYSTAL.CAREL@BSWHEALTH.ORG")
        REPLYTO = ("CRYSTAL.CAREL@BSWHEALTH.ORG")
        SUBJECT = ("REPORT TO BE SENDING")
        ATTACH  = ("&ATTACH1." CT="APPLICATION/MSEXCEL" EXT="XLSX");
      DATA _NULL_;
       FILE OUTBOX;
           PUT "Hello &NAME.,";
           PUT ;
           PUT %SYSFUNC(COMPBL(
                "This is an example email."));
           PUT ;
           PUT %SYSFUNC(COMPBL(
                "By using COMPBL we remove extra blanks from our text."));
           PUT %SYSFUNC(COMPBL(
                "There is no separation with this email line."));
       RUN;
    FILENAME OUTBOX CLEAR;
  %END;
  %ELSE %PUT NOTE: FILE DOES NOT EXIST AND NO EMAIL WILL BE SENT.;
%MEND FINDMYFILE;
%FINDMYFILE;

%MEND EMAILS;
%EMAILS (NAME=PERSON A, LOCATION=PLACE_A, TOSEND=("USER_PLACE_A@BSWHEALTH.ORG"))
%EMAILS (NAME=PERSON B, LOCATION=PLACEB, TOSEND=("USER_PLACE_B@BSWHEALTH.ORG"))
```

Remember to go look at the file pic I show you before this macro. You will note in the **EMAILS** macro call that for Person B that the Location was labeled "*PLACEB*". However our file was called "*PLACE_B*"!!!!!! Thus SAS® could not find a file for "*PLACEB"* and inserted the **NOTE** if the file did not exist (Figure 5). Good thing is we did get the file name for "*PLACE_A*" coded out correctly (Figure 6). Whew!

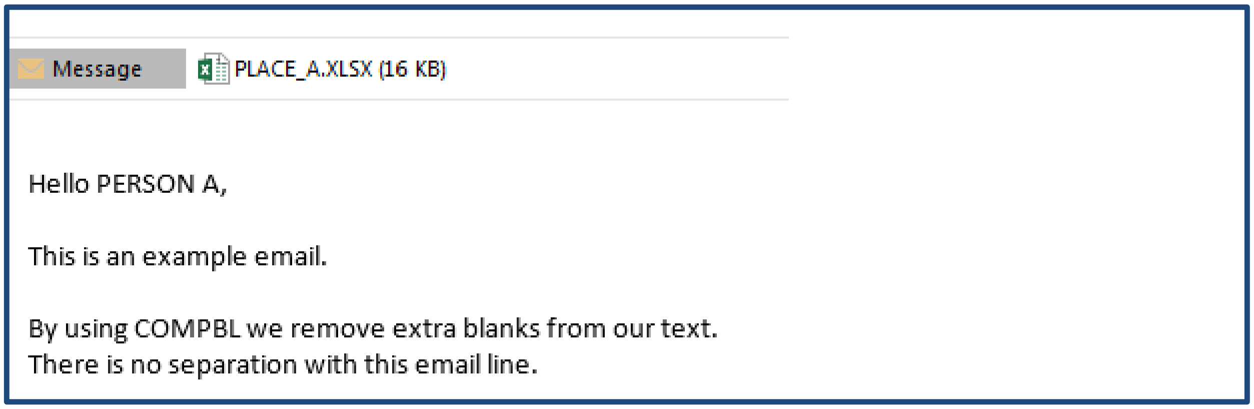
Figure 5: Example of SAS® LOG when entering the WRONG file name for PLACE_B



```
NOTE: 6 records were written to the file OUTBOX.
      The minimum record length was 0.
      The maximum record length was 53.
NOTE: DATA statement used (Total process time):
      real time           2.68 seconds
      user cpu time       0.01 seconds
      system cpu time     0.00 seconds
      memory              262.71k
      OS Memory           541532.00k
      Timestamp           09/29/2016 04:05:14 PM
      Step Count                        55  Switch Count  55


NOTE: Fileref OUTBOX has been deassigned.
59         %EMAILS (NAME=PERSON B, LOCATION=PLACEB, TOSEND=("CRYSTAL.CAREL@BSWHEALTH.ORG"))
NOTE: PERSON B FILE DOES NOT EXIST AND NO EMAIL WILL BE SENT.
```

Figure 6: Example of email sent for PLACE_A



Hello PERSON A,

This is an example email.

By using COMPBL we remove extra blanks from our text.
There is no separation with this email line.

CONCLUSION

By having emails set up in your SAS® code, you will be able to save a lot of time and headaches by having a repetitive process that you won't have to worry each time if you included the right attachments or recipients.

While doing development with the email code, I strongly encourage you to use your own email to test. ☺

Once you are satisfied with the email, wording, attachments, and the way it comes through the email system – only then would I suggest adding the appropriate recipients the email should be addressed. You do not want to be sending out emails while in development.

Using what you have learned, you can expand upon your skills and insert more macros, insert email lists controlled outside of your code, and insert data steps that do *IF/THEN/ELSE* logic based on output or files – the world is the limit within the realm of your outbox!

REFERENCES

Hunley, C. "*SMTP E-Mail Access Method: Hints, Tips, and Tricks.*" Proceedings of the SAS Global Forum Conference – 2010. Seattle, Washington. Available at http://support.sas.com/resources/papers/proceedings10/060-2010.pdf.

SAS Institute Inc., SAS® 9.4 Statements: Reference, Fifth Edition. "*FILENAME Statement, EMAIL (SMTP) Access Method*". Cary, NC. Accessed February 22, 2017. Available at http://support.sas.com/documentation/cdl/en/lestmtsref/68024/HTML/default/viewer.htm#n0ig2krarrz6vtn1aw9zzvtez4qo.htm.

Worden, J. and P. Jones. "*You've Got Mail – E-mailing Messages and Output Using SAS® EMAIL Engine.*" Proceedings of the SAS User Group International Conference – 2004. Montréal, Canada. Available at http://www2.sas.com/proceedings/sugi29/178-29.pdf.

CONTACT INFORMATION

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

Crystal Carel
Baylor Scott & White Health
214-265-3674
Crystal.Carel@BSWHealth.org