Paper SAS3212-2019

Macro-mize Your Time: Using SAS® to Email Hundreds of Individualized Reports in Minutes

Salma Ibrahim, SAS® Institute Inc.

ABSTRACT

Sending out individualized reports can take days if you have hundreds (or thousands) of recipients. SAS® can help you get that time back by generating and sending these emails. In this paper, we use SAS® macros to send hundreds of individualized reports in a matter of minutes. This process can be fully automated with a single input table that maps recipients to their respective reports. We cover options that can be applied to the emails, including personalizing the content of the email, sending from a designated email address, and marking the emails as high priority. In addition, we discuss creating summary reports that provide you with useful information, such as invalid email addresses and a snapshot of the emails sent out in a given run.

INTRODUCTION

Whether you need to send quarterly invoices to customers or annual reports to your employees, SAS® can help you avoid the need to manually draft emails. A combination of SAS® macros and the Email Access Method allows you to send these reports out at the click of a button. You can then schedule the program to run at the interval you need so that you no longer worry about drafting and sending these emails.

This paper will provide instruction and sample code for sending individualized emails with report attachments to recipients using SAS® macros and SAS® DATA step email options. The use case referenced in this paper is sending out annual employee compensation reports using sample HR data. You can replace the input data and modify report content for use cases relevant to your goals. We will start with a simple email and add useful options in subsequent sections. By the end, you will be able to send distinct reports to recipients given that you have a variable that maps a recipient to their respective report. You will also be able to add options like marking email importance and modifying sender or reply to email addresses, as well as create a simple report of invalid email addresses and emails that were sent out successfully.

EMAILING REPORTS AS ATTACHMENTS FROM SAS®

If the reports you have already exist, then you can send these emails out in just two steps. The first is a macro that reads in your end recipient's contact variables and generates the email. This macro will encompass all email options and the email body. The second step is a DATA step that passes the macro input using the data set that holds employee information and executes the macro for every employee (unique employee number) in the data set.

In this case, our inputs are employee name, email address, and department. Our reports are named using the employee's name and division, so we can reuse those input macro variables in the attachment specification. If that is not the case for you, you will need a variable in your data set for report name. Note that year is not a parameter that is read into the macro; this variable is defined earlier in the SAS® program because it is static.

The data set that holds your input variables will need a unique variable for each recipient. Because names can repeat, we use employee number as the BY-variable here. It is also a good idea to check and filter for duplicates so that recipients only receive one email each.

The macro sendreports defines your email options and email body:

```
%macro sendreports(name,email,department);

   filename outbox EMAIL;
         data _null_;
               FILE outbox
               to=("&email.")
               subject="&year. Annual Compensation Report: &name."
               attach=("C:\SGF_Reports\&name._&department..pdf");

   file outbox;
         put "Dear &name.:"; put ;
         put "Thank you for being a valuable part of our company's growth
               and success. Every year, we aim to reward our employees
               with merit increases and bonuses."; put ;
         put "Your annual employee compensation report is attached. Please
         follow up with your manager if you have any questions."; put ;
         put "Thank you,";
         put "Company";
   run;

%mend sendreports;
```

You can use put statements to add text and break lines to the email body. In the above example, we personalize the body with the name macro. You can use additional macros to further personalize the email body. For example, you could reference the employees' division, completed training for the year, years with the company, and so on. The only contingency is that you have that data available to you in the input data set (that we use to read in variables in the next step) and that you include those variables in the macro definition above as well as the call execute command below. You can take the email body customization a step further. There is a reference in the recommended reading at the end of this paper that provides guidelines for this.

The following DATA step reads the input variables into the send reports macro from our contact data set and invokes the macro for each employee:

```
data _null_;
   set employee.contact;
   by employee_number;
   if first.employee_number then do;
   call execute(cats('%sendreports(',name,',',email,',',department,')'));
   end;
run;
```

The DATA step above simply uses each employee's row of data to populate the macro's input variables. We sort by employee_number because it is our unique variable and it is necessary to use the first. qualifier in our if-then statement. The first. if statement qualifier will take care of the possibility of sending multiple emails to the same employee, but it is still a good idea to check for and resolve duplicates before running the code. As mentioned before, if you would like to include additional information (completed training for the year, years with the company, and so on.) in the email body, you will need to add those variables to the call execute statement in the DATA step above.

EMAIL OPTIONS

In the example above, we only define recipient, subject and attachment. There are many options available to you to further personalize to your use case.

Table 1 outlines some useful email options.

Email Options	**Description**
BCC=”bcc@email.com”	Allows you to send blind copy of the email. In our example, we could blind copy the employee’s manager and HR partner.
CC=”cc@email.com”	Allows you to send a copy to other recipients. This is another option to include the employee’s manager and/or HR partner
FROM=”from@email.com”	Allows you to designate a sender address in place of the default address of run user.
SENDER=”from@email.com”	When you use the FROM= option, the email shows both the FROM email address and the email address of the user running the program. If you’d like to just show the FROM= email address, set the SENDER= option to the same email address as FROM=.
REPLYTO=”reply@gmail.com”	Allows you to designate a different email address for recipients to reply to.
IMPORTANCE=”HIGH”	Allows you to set importance level for the email. Options are “LOW”, “NORMAL” (default), or “HIGH”.
SENSITIVITY=”COMPANY-CONFIDENTIAL”	Allows you to set sensitivity level for the email. Options are “NORMAL” (default), ”PRIVATE”, ”PERSONAL”, ”CONFIDENTIAL”, “COMPANY-CONFIDENTIAL”.
EMAILSYS=SMTP	Set before your FILE outbox EMAIL statement. Specifies which e-mail protocol to use for sending email within SAS.
EMAILPORT=25	Set before your FILE outbox EMAIL statement. Specifies the port number that is used by the SMTP server specifies in the EMAILHOST option.
EMAILAUTHPROTOCOL=NONE	Set before your FILE outbox EMAIL statement. Specifies whether the LOGIN authentication protocol should be used.
EMAILHOST=”mailhost.company.com”	Set before your FILE outbox EMAIL statement. Specifies one or more SMTP server domain names for your site.

Table 1. Email Options: Email Options for EMAIL (SMTP) Access Method

Note: If you are receiving one or both error messages below, please set the last four options listed in Table. 1 above. This error can appear when the “bitness” of SAS® and Microsoft Office do not match. Please reach out to your email administrator for assistance if unsure of the option values for your organization.

- “Either there is no default mail client or the current mail client cannot fulfill the messaging request. Please run Microsoft Outlook and set it as the default mail client.”

- ERROR: Undetermined I/O failure

There are additional options that can be viewed in the SAS® 9.4 Global Statements: Reference. The ones listed above are most useful to these use cases.

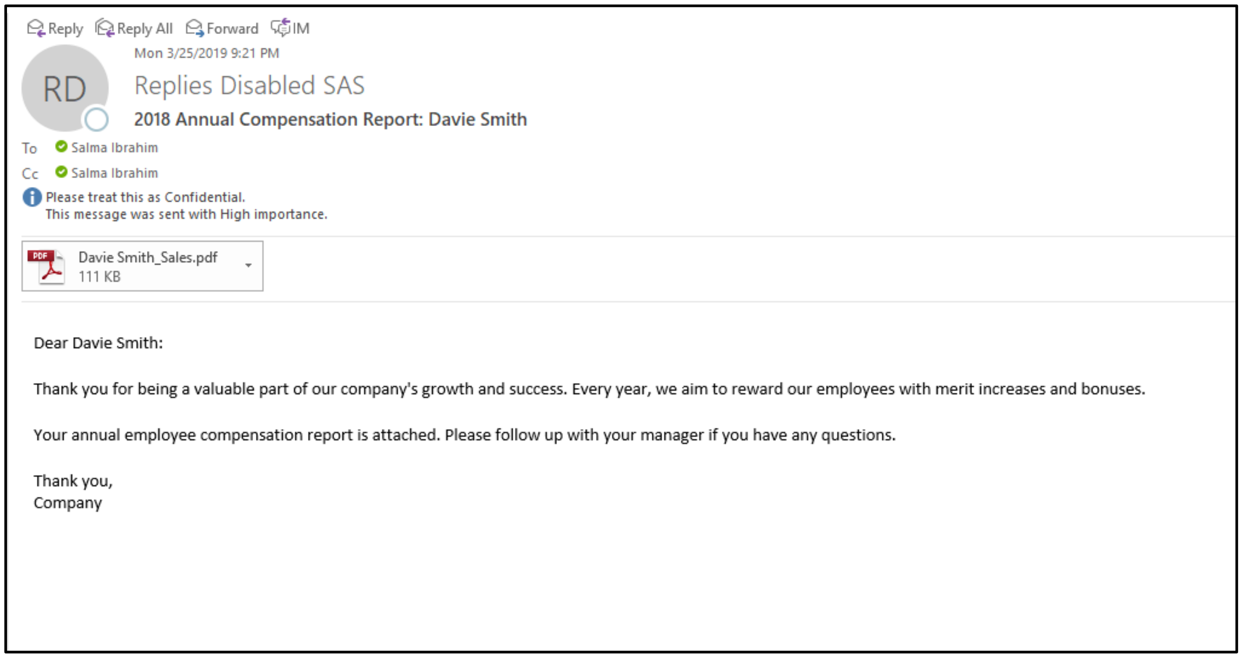
I prefer to use all the above options in my code to ensure that the emails turn out exactly as I would like them too. I recommend adding these in one by one and reviewing how they impact the way a test email appears in your inbox.

Here is the EMAIL options segment of our code with the options above defined:

```
filename outbox EMAIL;
      data _null_;
            FILE outbox
            to=("&email.")
            from=("replies-disabled@company.com")
            sender=("replies-disabled@company.com")
            bcc=("&mgremail." "deptmgremail.")
            cc="hr@company.com"
            replyto="hr@company.com"
            importance="HIGH"
            sensitivity="COMPANY-CONFIDENTIAL"
            subject="&year. Annual Compensation Report: &name."
            attach=("C:\SGF_Reports\&name._&department..pdf");
```

If you have more than one email address or attachment, you will need to separate them with a space. Note that you will need to include any additional input variables (&mgremail and &deptmgremail in this case) in the macro definition and in the macro call.

Display 1. is a sample of the emails generated by the code above. Notice the changes to the sender, recipients, importance, and sensitivity setting at the top of the email. Note: I've mapped sender and recipient email addresses to my email addresses for testing. If I were to select the reply button at the top of the email, the email address I defined in the replyto= statement above would populate the "To:" field in place of the default option (sender).



Mon 3/25/2019 9:21 PM

Replies Disabled SAS

2018 Annual Compensation Report: Davie Smith

To Salma Ibrahim

Cc Salma Ibrahim

Please treat this as Confidential.
This message was sent with High importance.

Davie Smith_Sales.pdf
111 KB

Dear Davie Smith:

Thank you for being a valuable part of our company's growth and success. Every year, we aim to reward our employees with merit increases and bonuses.

Your annual employee compensation report is attached. Please follow up with your manager if you have any questions.

Thank you,
Company

Display 1. Sample Email Generated Using Additional Options

GENERATING REPORTS WITHIN THE EMAIL MACRO

The above code assumes that your reports exist and are ready to attach to your emails. If the reports you're interested in sending out do not already exist (or you're interested in automating their creation), you can include their creation within the send reports macro created above. SAS® offers many options for creating customized reports that allow for creativity and detail. For the purposes of this paper, we will create very simple PDF reports. If you are interested in further customization, please visit the recommended references. You can also review Appendix A code for a more elaborate report.

We can accomplish creating the reports and emailing them out in the same two steps we referenced before. The report creation will be a part of the macro for sending them out. It is important to note that any variables to be used in the report are to be read into the macro with the employee contact information we mentioned before. This means that the input table we use to call the macro needs to include the variables you're reporting on. In this example, the additional variables for the report already exist in the input data set.

```
 %macro sendreports(name,email,department,salary,increase,newsalary,bonus);

     ods pdf file="C:\SGF_Reports\&name._&department..pdf";

           title j=center bold "&name. &year. Employee Compensation Report";
           ods text= "Dear &name.: "; ods text=" ";
           ods text= "Please see below your salary increase and bonus
           information for &year.. Changes and bonus disbursement will take
           effect in your next pay cycle. "; ods text=" ";
           ods text= "Current Salary: &salary. "; ods text=" ";
           ods text= "Salary Increase: &increase. "; ods text=" ";
           ods text= "New Salary: &newsalary. "; ods text=" ";
           ods text= "Bonus: &bonus. "; ods text=" ";

           run; quit;
     ods pdf close;

           data _null_;
                 FILE outbox
                 to=("&email.")
                 from=("replies-disabled@company.com")
                 sender=("replies-disabled@company.com")
                 cc="hr@company.com"
                 replyto="hr@company.com"
                 importance="HIGH"
                 sensitivity="COMPANY-CONFIDENTIAL"
                 subject="&year. Annual Compensation Report: &name."
                 attach=("C:\SGF_Reports\&name._&department..pdf");

     file outbox;
           put "Dear &name.:"; put ;
           put "Thank you for being a valuable part of our company's growth
                 and success. Every year, we aim to reward our employees
                 with merit increases and bonuses."; put ;
           put "Your annual employee compensation report is attached. Please
           follow up with your manager if you have any questions."; put ;
           put "Thank you,";
           put "Company";
     run;

  %mend sendreports;
```

```
data _null_;
   set employee.contact;
   by employee_number;
   if first.employee_number then do;
   call
   execute(cats('%sendreports(',name,',',email,',',department,',’,salary,’
   ,’,increase,’,’,newsalary,’,’,bonus,’)'));
   end;
run;
```

We use ODS PDF file to create the report attachment of employee compensation updates. As discussed before, we have added the relevant variables as input variables in the macro definition, and to the call execute statement that feeds the values into the macro during the DATA step that executes it. Please see the recommended reading references below for additional information about ODS reporting.

CREATING A SEND SUMMARY REPORT

When relying on automation to send important documents to several individuals, it’s important to create a summary report to account for successful sends and failed sends due to missing or invalid syntax email addresses. Of course, you could manually review the log and identify these cases, However, it is more efficient to have a report sent to you that identifies these cases so that you can correct for them. There are two relatively straightforward ways to accomplish this.

The first is using a code snippet to check the email variables in your data set for proper syntax. These checks will not catch everything but are a good start and will be useful paired with the log scrape. Here are some good checks to perform:

- Is the value missing?
- Is the value at least 7 characters long (the bare minimum for an email address is _@_.com, org, edu, and so on.)?
- Does the value include an “@” and a “.”?

The second way to do this is by parsing your log for the success and error statements associated with the EMAIL (SMTP) Access Method. The message “Message sent” indicates a successful send, and the line directly below that provides the email address of that successful send. We can use that line for reporting purposes.

The following are statements that indicate a syntax or send issue with an email address:

- `'WARNING: Bad e-mail address:'`
- `'ERROR: Email:'`

The message ends with the errored email address, so we can pull it for the summary report.

I prefer to use both methods mentioned above so that I am confident that I caught as many scenarios as I could. I also prefer to schedule this as a separate program from the program that generates the emails. The reason for doing this is so that there is a separate complete log to pull information from.

NOTE: Emails that are populated and have proper syntax won’t be identified as invalid, even if they no longer represent an active email account. In these cases, the sender will receive a bounced email in his or her inbox indicating the send has failed and a reason why. When reviewing your summary report, be sure to also look at the sender inbox for bounced emails that had valid syntax but presented another issue.

CREATING A TABLE OF INVALID SYNTAX EMAILS

The following code allows for the first check referenced above by creating a table of email addresses that do not meet the three conditions defined:

```
data employee.invalidsyntax;
   set employee.contact;
   if find(email,'@','i') ge 1 and find(email,'.','i') ge 1 and
   length(email) >= 7 then delete;
run;
```

The FIND function paired with the greater than or equal to operator is useful for determining whether a character exists in the value of a variable. This works similarly to CONTAINS but is compatible with if-then statements.

SCRAPING THE LOG FOR SUCCESSFUL AND FAILED SENDS

The following code allows for the second check referenced above by creating a table of successful sends based on the success message mentioned above, and creating a table of errored sends based on the error messages mentioned above:

```
data error;
   infile  "C:\SGF_Reports\logs\sendemails.log" truncover;
   input email_error $200.;
   if index(email_error, 'WARNING: Bad e-mail address:') > 0 or
   index(email_error, 'ERROR: Email:') > 0
   then output;
run;

data error ;
   set error;
   email=scan(email_error,-1," ");
run;

proc sort data=error; by email; run;

proc sort data=employee.contact out=mergecontact; by email; run;

proc sql;
   create table errorlog as
   select *
   from error left join mergecontact
   on error.email=mergecontact.email;
run;

proc sort data=errorlog nodupkey;
   by employee_number;
run;

data success;
   infile  "C:\SGF_Reports\logs\sendemails.log" truncover;
         input email_sent $100.;
   if index(email_sent, 'Message sent') > 0
   then do;
   count = 2;
   do until (count=0);
```

```
      output;
      count=count-1;
      if count = 1 then input email_sent $100.;
      end;
      end;
   run;

  data success (drop=email_sent count);
     set success;
     if email_sent="Message sent" then delete;
     email_sent=compress(email_sent,'"');
     email=substr(email_sent,14,-1);
   run;

   proc sort data=success; by email; run;

   proc sql;
      create table successlog as
      select *
      from success left join mergecontact
      on success.email=mergecontact.email;
   run;

   proc sort data=successlog nodupkey;
      by employee_number;
   run;
```

The INDEX function allows us to search the log for our success and error messages of interest. In the case of the error messages, the errored email address is in the same line as the error message. For this reason, we can just extract the email address from the error message using the SCAN function and then use that to merge the email with the employee's other information. Because the successful error message appears in the line following the successful send message, we use a do loop to pull the text from the second line and then use the compress and substring functions to pull the email address from that line. We can then merge the successful email addresses with employee data as well.

For your reference, the line containing the email address for successful sends is as follows:

- To: "employee@company.com"

That is why we compressed to remove the quotation marks and then pull the email address starting at position 14. In comparison, we can just scan -1 from the end of the error statement to pull the errored email addresses in those cases.

Note: If you have recipients with valid email addresses copied on the email but your main recipient has an invalid email address, your email will still successfully send to those who were copied. In this case, the recipient will appear in both the successful sends report and the error log report.

EMAILING OUT THE SUMMARY REPORT

Here we combine the two code snippets above and send out the summary report of invalid email syntax, successful sends and errored email addresses:

```
   ods excel file="C:\SGF_Reports\Syntax_Check_&year..xlsx";
      proc print data=employee.invalidsyntax;
      title "Invalid Email Syntax in Employee Dataset";
      run;
   ods excel close;

   ods excel file="C:\SGF_Reports\SENT_TODAY_&year..xlsx";
      proc print data=success;
      title "Reports Successfully Sent Today";
      run;
   ods excel close;

   ods excel file="C:\SGF_Reports\INVALID_EMAIL_&year..xlsx";
      proc print data=errorlog;
      title "Errors Due to Invalid Email";
      run;
   ods excel close;

   filename outbox EMAIL;
      data _null_;
            FILE outbox
            to=("my_email@company.com")
            importance="HIGH"
            subject="Summary Report for Sending Employee Compensation Reports"
            attach=("C:\SGF_Reports\Syntax_Check_&year..xlsx"
            "C:\SGF_Reports\SENT_TODAY_&year..xlsx"
            "C:\SGF_Reports\INVALID_EMAIL_&year..xlsx");
      file outbox;

            put "Please see attached a summary of all reports that were sent
            out today, and a report of errors due to email addresses that are
            invalid.";
            put;
            put "Failed sends will need to be investigated and/or sent
            manually.";
            put;
   run;
```

Here we use ODS again to create Excel files of the error and successful send data sets. You will need to define year as a macro somewhere in your program to use it in the file names. **Note:** You will not be able to open the excel attachment for any data sets that were empty.

CONCLUSION

You can use SAS® as a great time-saving tool for generating and sending emails, whether you need to send a few or few thousand reports. Concerns about customization can be a deterrent from automating these processes but using macro variables can help you overcome that concern and customize as much as you'd like.

There is a lot of opportunity to build on the examples we've walked through today. You can add more employee information in the email body by reading in additional macro inputs, create a more elaborate PDF File report (perhaps with logos, images or text customization) when generating the report within the send reports macro, or you can pull more information from the log for a more robust summary report. I recommend that you brainstorm what features are most useful for you, and reference additional SAS® resources to incorporate them into your code.

APPENDIX A: CODE EXAMPLE FOR AN ELABORATE REPORT

The following code generates a more elaborate report when included within the send reports macro. Note that we add variables for performance rating, years at company, and hours spent in training. The report below includes a company logo, formatted titles, a table of compensation changes, and a table of milestones achieved. Please refer to the associated SAS® programs for more details around input tables, format, and options.

```
ods _all_ close;
options nodate nonumber orientation=portrait;
ods escapechar="^";
ods noresults;

data comp_table;
    set comp_table;
    if Employee_Information="Department" then Value="&department.";
    if Employee_Information="Prior Salary" then Value="$&salary.";
    if Employee_Information="Increase" then Value="$&increase.";
    if Employee_Information="New Salary" then Value="$&newsalary.";
    if Employee_Information="Bonus" then Value="$&bonus.";
run;

data merit;
    set merit;
    if Metric="Performance Rating" then Milestone="&performancerating.";
    if Metric="Years with Company" then Milestone="&yearsatcompany. Years";
    if Metric="Training Hours Completed" then
    Milestone="&trainingtimeslastyear. Hours";
run;

ods pdf file="C:\SGF_Reports\&name._&department..pdf";
    title j=left bold "^S={preimage='C:\SGF_Reports\Logo.png'}";
    title2 j=left bold "&year. Employee Compensation Statement^n";
    title3 j=left italic "Personal and Confidential ^n^n";

    proc report data=table split='00'x style(header)={background=CX3883A8
    foreground=white font_size=7pt} style(column)=[font_size=7pt];
    define Employee_Information / style(header)={cellwidth=3.5in}; define
    Value / style(header)={cellwidth=3.5in};
    label Employee_Information="Metric";
    label Value="Compensation";

ods text= "&name.,^n^n";
ods text= "Company recognizes excellent performance and rewards employees
for helping the company achieve its business goals. When we succeed
together, we share the rewards. Thank you for continuing to help Company
solve our customers’ toughest business problems.^n^n";
ods text= "Congratulations, I am pleased to provide you with the following
pay increase effective the next pay cycle. You have also been awarded a
bonus, paid to you on the next pay cycle.^n^n";
ods text= "Company will provide eligible employees with an annual 401(k)
contribution of 3% to be funded in April. Eligible employees also receive
an additional 401(k) contribution of 3% of their eligible compensation each
pay period, regardless if the employee chooses to contribute or not,
bringing this year’s 401(k) contribution to 6%.
This combined total outranks our competitors in 401(k) employer
contributions. Helping you save for retirement is one of the ways Company
invests in its employees.^n^n";
```

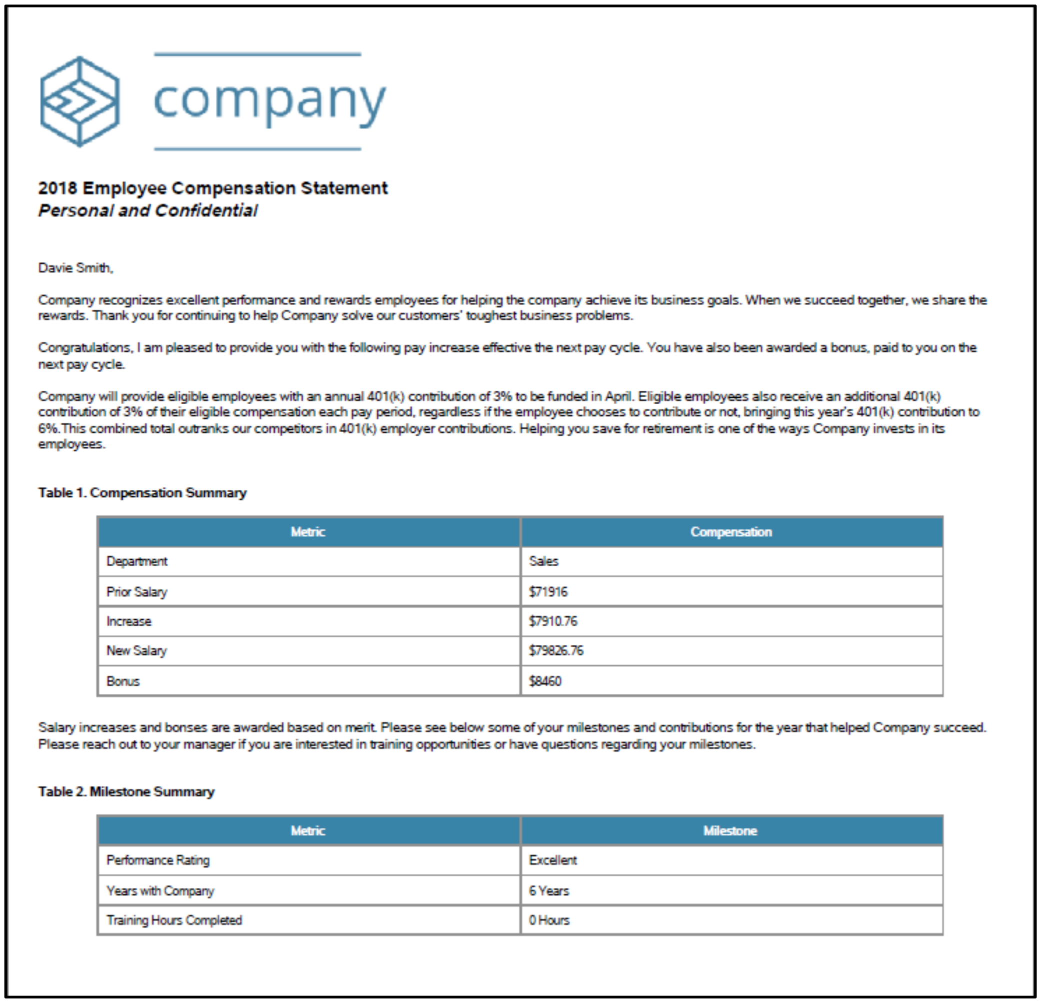
```
ods text="^n^S={font_weight=bold}Table 1. Compensation Summary ^S={}";

ods startpage=no;
    proc report data=merit split='00'x style(header)={background=CX3883A8
    foreground=white font_size=7pt} style(column)=[font_size=7pt];
    define Metric / style(header)={cellwidth=3.5in}; define Milestone /
    style(header)={cellwidth=3.5in};

ods text= "^nSalary increases and bonses are awarded based on merit. Please
see below some of your milestones and contributions for the year that
helped Company succeed. Please reach out to your manager if you are
interested in training opportunities or have questions regarding your
milestones.^n^n";
ods text="^n^S={font_weight=bold}Table 2. Milestone Summary ^S={}";

run; quit;
ods pdf close;
```

Display 2. is a sample of the more elaborate report referenced above. Note that additional variables are needed for this case.



2018 Employee Compensation Statement
Personal and Confidential

Davie Smith,

Company recognizes excellent performance and rewards employees for helping the company achieve its business goals. When we succeed together, we share the rewards. Thank you for continuing to help Company solve our customers' toughest business problems.

Congratulations, I am pleased to provide you with the following pay increase effective the next pay cycle. You have also been awarded a bonus, paid to you on the next pay cycle.

Company will provide eligible employees with an annual 401(k) contribution of 3% to be funded in April. Eligible employees also receive an additional 401(k) contribution of 3% of their eligible compensation each pay period, regardless if the employee chooses to contribute or not, bringing this year's 401(k) contribution to 6%.This combined total outranks our competitors in 401(k) employer contributions. Helping you save for retirement is one of the ways Company invests in its employees.

Table 1. Compensation Summary

Metric	Compensation
Department	Sales
Prior Salary	$71916
Increase	$7910.76
New Salary	$79826.76
Bonus	$8460

Salary increases and bonses are awarded based on merit. Please see below some of your milestones and contributions for the year that helped Company succeed. Please reach out to your manager if you are interested in training opportunities or have questions regarding your milestones.

Table 2. Milestone Summary

Metric	Milestone
Performance Rating	Excellent
Years with Company	6 Years
Training Hours Completed	0 Hours

Display 2. Sample Elaborate Report Generated Within Macro

REFERENCES

SAS® Institute. "FILENAME Statement, EMAIL (SMTP) Access Method." SAS(R) 9.4 Global Statements: Reference. Accessed March 4, 2019. Available at https://go.documentation.sas.com/?docsetId=lestmtsglobal&docsetTarget=n0ig2krarrz6vtn1aw9zzvtez4qo.htm&docsetVersion=9.4&locale=en.

RECOMMENDED READING

- *Make it SASsy: Using SAS® to Generate Personalized, Stylized, and Automated Email Lisa Walter, Cardinal Health, Dublin, OH* available at https://www.mwsug.org/proceedings/2010/appdev/MWSUG-2010-89.pdf
- *Base SAS® ODS (Output Delivery System)* available at http://support.sas.com/rnd/base/ods/

CONTACT INFORMATION

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

Salma Ibrahim
Salma.Ibrahim@sas.com