MWSUG 2019 - Paper RF24

Navigating the Ins and Outs of Mapping in SAS® Enterprise Guide®

Ruth Kurtycz, Spectrum Health, Grand Rapids, MI

ABSTRACT

Mapping is an extremely valuable data visualization tool. Maps are approachable and intuitive, which can be especially useful when trying to display data in such a way that will reach an audience without an analytical background. Although SAS® has the capabilities to easily produce maps at the global, country, and state level, creating maps for smaller geographic areas requires some extra steps. This paper will explain how to create zip code and census tract level maps using SAS Enterprise Guide®. We will go over the basics of obtaining zip code and census tract shapefiles and importing them with the MAPIMPORT procedure. We will explain how to annotate a map using the %ANNOMAC and %MAPLABEL macros, and the process of creating the final visualizations with the GPROJECT, GREMOVE, and GMAP procedures. We will also touch on prepping data for mapping and creating formats for the data categories. The example maps display infant mortality data from 2001 to 2018 for Kent County, Michigan.

INTRODUCTION

Data visualization is a vitally important tool for analysts seeking to engage their audience. Although pie and bar charts are familiar to most people, they are limited to very basic information, such as overall counts and percentages. Bubble charts and stacked bar charts contain more detail, but the visuals can quickly become overwhelming for the audience to understand. This is where the visual of mapped data really shines. Nearly everyone is familiar with maps; we have seen them in our classrooms when learning geography and on our phones when traveling to a new location. In any context, maps are tools of navigation and exploration. Mapping data creates an instantly recognizable and relatable visual that engages all types of audiences while highlighting geographical data insights.

SAS Enterprise Guide contains an internal MAP library with over 300 shapefiles for creating maps at the global, country, and state level (Thompson, 2011), but many times a higher level of granularity is needed to see meaningful geographic trends. This paper will walk through one technique to create zip code and census tract maps in SAS Enterprise Guide. Readers should have at least a basic understanding of SAS code and data manipulation.

THE DATA

The examples in this paper will focus on mapping the number of infant deaths at a zip code and census tract level in Kent County, Michigan between 2001 and 2018. Infant mortality is an important measure of the health and well-being of a population. It is important to track infant deaths so that we may see emerging trends in the communities' health and act on them. The Kent County Fetal Infant Mortality Review (FIMR) program reviews all the deaths of infants one year of age or less whose mothers are residents of Kent County. Kent County is the only FIMR program in the state that reviews every single infant death; this is done so that the data may be analyzed to look at possible causes of disparities in infant mortality among many risk factors, including various racial groups and ethnicities (MacDonald, Kurtycz, Neff, 2018).

There have been 1,090 cases of infant mortality in Kent County between 2001 and 2018. The FIMR Kent County database contains a separate record for each de-identified case and includes, among other information such as family demographics, the zip code and census tract location of the family's home. This is what we will use to create the maps. A sample of this data is shown below.

	year_of_death	zip_code	tract	death_count
1	2005	49507	26081003100	1
2	2001	49505	26081110200	1
3	2001	49508	26081004500	1
4	2001	.	.	1
5	2001	49507	26081003400	1
6	2001	49525	26081000100	1
7	2001	49505	26081000900	1
8	2001	49506	26081003300	1
9	2001	49503	26081002000	1
10	2001	49509	26081014000	1
11	2001	49509	26081013800	1
12	2001	49507	26081003600	1
13	2001	49507	26081003600	1
14	2001	49505	26081013800	1
15	2001	49506	26081003300	1
16	2001	49505	26081110200	1
17	2001	49503	.	1
18	2001	49507	26081003000	1
19	2001	49546	26082260100	1
20	2001	49507	26081003200	1

Display 1. FIMR Infant Mortality Data Sample

To prepare this data for mapping, we use the SQL procedure to calculate the total number of infant deaths from 2001 to 2018 that occurred in each zip code and each census tract, respectively. By using the SUM() and GROUP BY statements, we create data sets with the total number of infant deaths that occurred in each zip code (FIMR_Zip) or census tract (FIMR_Tract) over the period:

```
PROC SQL;
   CREATE TABLE FIMR_ZIP AS SELECT
   ZIP_CODE as ZCTA5
   ,SUM(DEATH_COUNT) AS DEATHS
   FROM FIMR
   GROUP BY ZIP_CODE;
QUIT;

PROC SQL;
   CREATE TABLE FIMR_TRACT AS SELECT
   TRACT
   ,SUM(DEATH_COUNT) AS DEATHS
   FROM FIMR
   GROUP BY TRACT;
QUIT;
```

To finish preparing the data for mapping, we use the FORMAT procedure in coordination with the DATA step to manually set the six categories of infant death counts our maps will display. SAS will automatically calculate categories for the maps if desired but in this case, we need a specific number of categories and cutoff values for the infant death counts. We also create labels for the zip code, census tract, and death count variables within each DATA step:

```
PROC FORMAT;
   VALUE deaths_zip 0-15 = "0-15"
                   16-30 = "16-30"
                   31-45 = "31-45"
                   46-60 = "46-60"
                   61-75 = "61-75"
                  76-1000 = "76+";
```

```
   VALUE deaths_tract  0-3 = "0-3"
                       4-6 = "4-6"
                       7-9 = "7-9"
                     10-12 = "10-12"
                     13-15 = "13-15"
                    16-100 = "16+";
RUN;

DATA fimr_zip2;
   SET fimr_zip;
   FORMAT deaths deaths_zip.;
   LABEL zcta5 = "Zip Code";
   LABEL deaths = "Total Infant Deaths";
RUN;

DATA fimr_tract2;
   SET fimr_tract;
   FORMAT deaths deaths_tract.;
   LABEL tract = "Census Tract";
   LABEL deaths = "Total Infant Deaths";
RUN;
```

EXTERNAL SHAPEFILES

As mentioned above, SAS Enterprise Guide contains a MAP library with shapefiles for creating global, country, and state level maps, but to visualize smaller areas we must utalize external shapefiles. The following section will provide information on where these shapefiles can be found online and how to import them into SAS.

OBTAINING AND DOWNLOADING SHAPEFILES

Zip code and census tract shapefiles are available as a free download through the United States Census Bureau website. This data can be found with a simple internet search, but one access point is below. The files from the census.gov website are ESRI format shapefiles that can be imported into SAS using the MAPIMPORT procedure (Okerson, 2013). To download these files, navigate to the following link and select the desired granularity and location from the dropdown menu:

http://www.census.gov/cgi-bin/geo/shapefiles2010/main

The download contains five files, three of which are required for SAS mapping (Okerson, 2013). Save the .DBF, .SHP, and .SHX files in a folder location that is accessible to SAS for the import, such as the My Documents folder.

It should be noted that the census tract shapefile from the Census Bureau website will only be used to create the county border on the zip code map. The Census Bureau shapefile is not compatible with the method used to create the census tract map because it is missing an order variable to instruct SAS on how to connect the census tract borders. The shapefile with a valid order variable used for the census tract map in this paper was created by Mark Evans as part of a workaround for a similar mapping issue in Tableau (Evans, 2016). The Excel™ file with census tract mapping information for the United States is current as of 2016 and is available as a free download at the following link:

http://bigbytes.mobyus.com/tableau/census_tract_shapefiles_all.csv

IMPORTING AND CLEANING

Once the files are saved in the desired location, we will import the shapefiles using PROC MAPIMPORT. The DATAFILE= statement specifies the location of the shapefiles and the OUT= statement defines the output data set name. The “26” in the shapefile name represents a data set with information specific to Michigan:

```
PROC MAPIMPORT
   DATAFILE="M:\Hcp\EVALUATION DATA\EVALUATION TEAM\Mapping Files\Zip
   Codes\tl_2016_26_zcta510.shp"
   OUT=Michigan_Zip;
RUN;

PROC MAPIMPORT
   DATAFILE="M:\Hcp\EVALUATION DATA\EVALUATION TEAM\Mapping Files\Census
   Tracts\tl_2016_26_tract.shp"
   OUT=Michigan_census;
RUN;
```

Since the ordered census tract information is saved as an Excel file, not a shapefile, we can import it using a simple IMPORT procedure. The DATAFILE= and OUT= statements function as in PROC MAPIMPORT above. The REPLACE command overwrites the previous SAS data set each time the code is run. This is done so the entire mapping program can be run multiple times during a session without incurring any errors when this import code executes:

```
PROC IMPORT
   DATAFILE="M:\Hcp\EVALUATION DATA\EVALUATION TEAM\Mapping Files\ORDERED
   CENSUS TRACTS.xlsx"
   OUT=census_tract_order REPLACE;
RUN;
```

Once the shapefiles have been imported into SAS, the following code restricts the data to information specific to Kent County. There are 28 zip codes that fall partially or completely within the county boundaries. We use an IF IN() statement to limit the data to only those 28 zip codes. In the census tract file, Kent County is coded as county number 81; we use an IF statement with this information to limit the data to only those census tracts within Kent County. In addition to restricting the data, the following code converts the zip code and census tract variables from character type to numeric type to make for easier merging later in the program:

```
DATA KentZips;
   SET Michigan_ZIP;
   ZCTA5=ZCTA5CE10+0;
   IF ZCTA5 IN(49301,49302,49306,49315,49316,49319,49321,49326,49330,49331
   ,49341,49343,49345,49418,49503,49504,49505,49506,49507,49508,49509,
   49512,49519,49525,49534,49544,49546,49548);
RUN;

DATA KentTracts;
   SET Michigan_census(RENAME=(GEOID=GEOID_C));
   GEOID=GEOID_C+0;
   IF COUNTYFP = "081";
RUN;
```

The ordered census tract file has no distinguishing variable to denote state or county. To restrict this data, we PROC SQL with LEFT JOIN to keep only those tracts identified as belonging to Kent County from the U.S. Census Bureau census tract shapefile above:

```
PROC SQL;
   CREATE TABLE ORDERED_TRACTS AS SELECT
   B.*
   FROM KentTracts AS A
   LEFT JOIN CENSUS_TRACT_ORDER AS B
   ON A.GEOID=B.GEOID;
QUIT;
```

SETTING UP THE MAPS

Here we will discuss the use of the %ANNOMAC and %MAPLABEL macros to create and control the font, color, and size of map labels. We will also explain how to use the GREMOVE procedure to add a boundary of interest to the map. Finally, we will show how the GPROJECT procedure is used in conjunction with the ordered census tract shapefile to convert latitude and longitude values to Cartesian coordinates; this will correct for any distortion caused by mapping detailed geographic areas.

MAP MACROS FOR LABELING

Adding zip code labels will enhance the information displayed on the map by identifying areas the readers may be familiar with. We will not create labels for the census tract map because while most people know their zip code of residence, very few of us know our census tract; adding census tract labels would crowd the map and would not add value to the reader in the same way zip codes will.

To label our maps we use the %ANNOMAC and %MAPLABEL macros to create an output data set that can be used to annotate graphics created with the GMAP procedure. The %ANNOMAC macro must be run first because it compiles all the SAS annotate macros and makes them available for use within the session. The %ANNOMAC macro has nine arguments, seven of which are required: map data set, attribute data set, output data set, label variable, ID list, label font, and label color. Size and hsys are optional arguments to specify the label size and default to 2 and 3, respectively (Annotate Macro Dictionary, 2019).

Before calling the macro, we use the SORT procedure to create an unduplicated list of zip codes in a data set called KentZips_Nodup. The NODUPKEY option removes any duplicates in the key variable, zip code:

```
PROC SORT DATA=KentZips out=KentZips_Nodup NODUPKEY;
   BY ZCTA5;
RUN;
```

We use this unduplicated zip code list in both the map and attribute data set arguments of the %MAPLABEL macro. We specify the name of the output data set as Anno_Label. Both the label and ID variable is zip code (ZCTA5). The font the labels will use is Arial Black, the color is black, and the size is slightly increased from the default value of 2 to 2.5:

```
%ANNOMAC;
%MAPLABEL(KENTZIPS_Nodup,KENTZIPS_Nodup,Anno_Label,ZCTA5,ZCTA5,Arial
Black,BLACK,2.5,3);
```

ADDING A COUNTY BOUNDARY

Next, we want to add the Kent County boundary to the zip code map. Although we are interested in the distribution of infant deaths by zip code, the county is our FIMR service area and zip codes do not generally follow county borders. The GREMOVE procedure is used to create a data set with only the outline of the area of interest, which in our case is the border of Kent County (GREMOVE Procedure, 2019). We use the Census Bureau census tract shapefile as the input data set and specify Kent_County_Outline as the output data set. The BY statement identifies CountyFP as the unit area of interest; recall that we used the CountyFP variable earlier to limit the census tracts to only those found within county 81 (i.e. Kent County). The ID statement identifies the current, smallest unit area of the input data set, which is census tract (TractCE):

```
PROC GREMOVE DATA = KentTracts OUT = Kent_County_Outline;
   BY CountyFP;
   ID TractCE;
RUN;
```

The code below builds a data set that mimics the output of the %MAPLABEL macro, including options to specify the color and size of the border. We also included a line variable which establishes a segmented

style line for the boundary. For the first observation in the data set, the function variable is set to “move”; this tells SAS to move the cursor into position to draw the border (Schurr, Wiseman, 2013). All other functions are set to “draw” so SAS will connect the observations marking the outline of the county as identified by the output of PROC GREMOVE:

```
DATA County_Line;
   LENGTH function $ 8 color $ 8 position $ 1 line 8.;
   RETAIN xsys ysys '2' hsys '3'
       when 'a' position '5' line 33 size 0.6 color 'black';
   SET Kent_County_Outline;
   IF _N_ = 1 THEN function = 'move';
   IF _N_ NE 1 THEN function = 'draw'; OUTPUT;
   KEEP function color line size position xsys ysys hsys when x y;
RUN;
```

Lastly, the Anno_Label data set from the %MAPLABEL macro is merged with the County_Line data set above to create the final annotate data set for the zip code map. This data set includes both the zip code labels and the border of Kent County:

```
DATA Anno_Label_wCounty;
   SET Anno_Label County_Line;
RUN;
```

CONVERTING COORDINATES WITH THE GPROJECT PROCEDURE

PROC GPROJECT converts traditional coordinates (i.e. latitude and longitude) to Cartesian coordinates. Coordinates are meant to represent a spherical, three-dimensional area but we are attempting to create a visualization of a map on a two-dimensional plane. Due to this, the results of mapping with latitude and longitude variables can be distorted, particularly when it comes to mapping smaller, more detailed areas such as census tracts. As a workaround, we use the process of projecting to convert the coordinates from three- to two-dimensional (GPROJECT Procedure, 2019).

Before utilizing PROC GPROJECT, we use the SORT procedure to order the data by census tract (geoid) and point order. We also rename the traditional coordinates to X and Y so PROC GPROJECT will automatically recognize the proper variables for use in the conversion:

```
PROC SORT DATA=census_tract_order
   (RENAME=(PointLongitude=X PointLatitude=Y));
   BY geoid PointOrder;
RUN;
```

In PROC GPROJECT, we identify the input data set as the ordered census tracts and name the output data set tract_proj. The DEG option specifies that the units of the X and Y variables are measured in degrees and the EASTLONG option indicates that the latitude (X) variable increases as the map moves east; this ensures the map is not reversed after projection:

```
PROC GPROJECT DEG EASTLONG DATA=census_tract_order OUT=tract_proj;
   ID geoid;
RUN;
```

CREATING THE MAPS

We are nearly ready to create the maps. First, we need to select the colors SAS will use to represent the six categories of infant death. The SAS code below identifies six colors ranging from a light yellow to a dark orange. See the appendix for a full list of suggested color schemes and gradients (Schurr, Kurtycz, 2014):

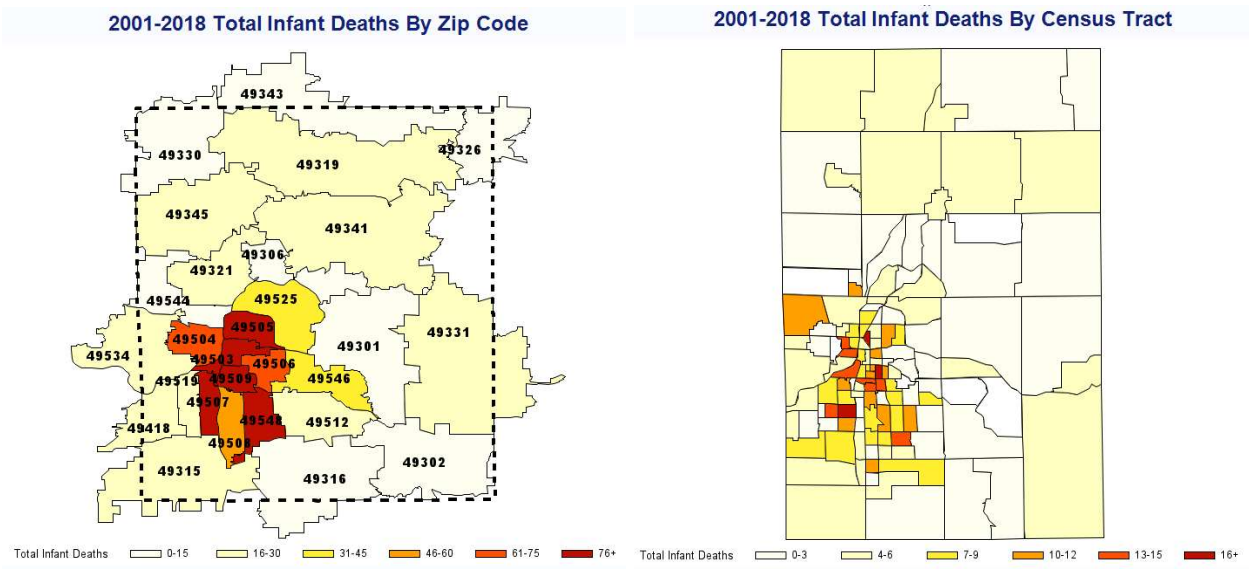
```
pattern1 c=CXFFFFF0;
pattern2 c=CXFFFFC0;
pattern3 c=CXFFEF30;
pattern4 c=CXFF9F00;
pattern5 c=CXFF4F00;
pattern6 c=CXBF0F00;
```

For the final step of the program, we generate the zip code and census tract maps using the GMAP procedure. The DATA= statement identifies the data set with the variable we want to map (i.e. infant deaths) and the MAP= statement locates the shapefile for each map; note that the census tract map uses the output data set from PROC GPROJECT. The ID statement tags the areas to be mapped. The CHORO statement establishes the variable that will be mapped using the previously identified colors and the DISCRETE option creates a new category for each unique value of the CHORO variable (GMAP Procedure, 2019). Since we previously applied our custom format to the death count variable, this option allows us to display the categories exactly as we defined them. Removing this option would permit SAS to calculate the number and cutoffs of the mapped variable categories. In the zip code map, the ANNO= statement connects to the previously created data set with the zip code labels and county border. Finally, both maps use the TITLE statement to add context to the visualization:

```
PROC GMAP DATA=FIMR_Zip2 MAP=KentZips;
   ID ZCTA5;
   CHORO deaths/ANNO=Anno_Label_wCounty DISCRETE;
   TITLE "2001-2018 Infant Deaths by Zip Code";
RUN;
QUIT;

PROC GMAP DATA=FIMR_Tract2 MAP=Tract_Proj;
   ID geoid;
   CHORO deaths/DISCRETE;
   TITLE "2001-2018 Infant Deaths by Zip Code ";
RUN;
QUIT;
```

The final output of the program is below. On the zip code map, we can see that a higher number of infant deaths are clustered in the lower left corner of the county. This is the approximate location of downtown Grand Rapids, which has a higher population density than the surrounding areas, as well as a higher proportion of low-income and disadvantaged families. We would expect to see a higher number of infant deaths occurring in this area. The census tract map provides us more detail as to where exactly in Grand Rapids these deaths have been occurring. By viewing the deaths geographically at such a granular level, we can begin to examine these specific tracts for patterns and underlying causes that may help us prevent future infant deaths in these areas.



Figures 1 and 2. Infant Deaths from 2001-2018 in Kent County, MI by Zip Code (left) and Census Tract (right)

CONCLUSION

Mapping data allows us to visualize and recognize geographical patterns. Although the capabilities exist in SAS to create global, country, and state level maps, a higher level of detail can easily be achieved by utilizing external shapefiles. The addition of custom map annotation for county borders and zip code labels can add an extra dimension of information and context to these visualizations, making them even more effective at communicating any data insights.

Maps give us an intuitive way to view data that is both accessible and understandable to those with and without an analytics background. Whether applied to infant mortality, product sales, or some other variable, readers should consider utilizing SAS with the techniques described in this paper to create zip code and census tract maps for their own visualization needs.

REFERENCES

"Annotate Macro Dictionary". SAS/GRAPH(R) 9.3: Reference, Third Edition. 2019. Available at http://support.sas.com/documentation

Evans, Mark. "Mapping Census Tracts in Tableau". *I Like Big Bytes*. 2016. Available at http://ilikebigbytes.com/index.php/2016/09/07/mapping-census-tracts-in-tableau/

"GMAP Procedure". SAS/GRAPH(R) 9.3: Reference, Third Edition. 2019. Available at http://support.sas.com/documentation

"GPROJECT Procedure". SAS/GRAPH(R) 9.3: Reference, Third Edition. 2019. Available at http://support.sas.com/documentation

"GREMOVE Procedure". SAS/GRAPH(R) 9.3: Reference, Third Edition. 2019. Available at http://support.sas.com/documentation

MacDonald, Sarah; Kurtycz, Ruth; Neff, Ray. 2018. *2001-2017 Infant Deaths in Kent County Michigan: A Statistical Update 2001-2017*. Grand Rapids, MI: Spectrum Health Healthier Communities.

Okerson, Barbra. 2013. "Creating Zip Code-Level Maps with SAS". *Proceedings at 2013 SAS Global Forum*. San Francisco, CA: SASGF. Available at https://support.sas.com/resources/papers/proceedings13/214-2013.pdf

Schurr, Kathryn; Kurtycz, Ruth. 2014. "Categorical and Contiguous – The Best of Both Worlds". *Proceedings at the 2014 Midwest SAS Users Group*. Chicago, IL: MWSUG. Available at https://www.mwsug.org/proceedings/2014/DV/MWSUG-2014-DV01.pdf

Schurr, Kathryn; Wiseman, Jonathan. 2013. “Bordering on Success with PROC GMAP in SAS: Utilizing Annotate Data sets to Enhance Your Maps”. *Proceedings at 2013 Midwest SAS Users Group*. Columbus, OH: MWSUG. Available at https://www.mwsug.org/proceedings/2013/DV/MWSUG-2013-DV06.pdf

Thompson, Stephanie. 2011. “Easier than You Think: Creating Maps with SAS Enterprise Guide”. *Proceedings at 2011 SAS Global Forum*. Las Vegas, NV: SASGF. Available at http://support.sas.com/resources/papers/proceedings11/311-2011.pdf

APPENDIX

The code and visualization below showcase some of the color scheme and gradient options available when creating maps in SAS (Schurr, Kurtycz, 2014).

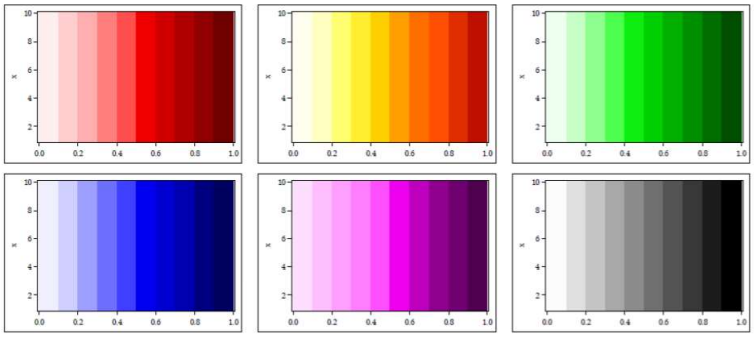


Figure 3. SAS Visualization Color Gradients

```
*COLOR DIRECTORY;
*RED;
   color1='CXFFEFEF'; color2='CXFFCFCF'; color3='CXFFAFAF';
   color4='CXFF7F7F'; color5='CXFF4F4F'; color6='CXEF0000';
   color7='CXCF0000'; color8='CXAF0000'; color9='CX8F0000';
   color10='CX6F0000';

*ORANGE AND YELLOW;
   color11='CXFFFFF0'; color12='CXFFFFC0'; color13='CXFFFF70';
   color14='CXFFEF30'; color15='CXFFCF00'; color16='CXFF9F00';
   color17='CXFF6F00'; color18='CXFF4F00'; color19='CXDF2F00';
   color20='CXBF0F00';

*GREEN;
   color21='CXEFFFEF'; color22='CXC8FFC8'; color23='CX8FFF8F';
   color24='CX4FFF4F'; color25='CX0FEF0F'; color26='CX00CF00';
   color27='CX00AF00'; color28='CX008F00'; color29='CX006F00';
   color30='CX004F00';

*BLUE;
   color31='CXEFEFFF'; color32='CXCFCFFF'; color33='CX9F9FFF';
   color34='CX6F6FFF'; color35='CX3F3FFF'; color36='CX0000EF';
   color37='CX0000CF'; color38='CX0000AF'; color39='CX00007F';
   color40='CX00005F';

*PURPLE;
   color41='CXFFDFFF'; color42='CXFFBFFF'; color43='CXFF9FFF';
   color44='CXFF7FFF'; color45='CXFF4FFF'; color46='CXEF00EF';
   color47='CXBF00BF'; color48='CX8F008F'; color49='CX6F006F';
   color50='CX4F004F';
```

```
*GRAY;
   color51='GRAYFC'; color52='GRAYE0'; color53='GRAYC4'; color54='GRAYA8';
   color55='GRAY8C'; color56='GRAY70'; color57='GRAY54'; color58='GRAY38';
   color59='GRAY1C'; color60='GRAY00';
```

US Census Bureau Shapefile Download Menu:
http://www.census.gov/cgi-bin/geo/shapefiles2010/main

Census Tract Ordered Shapefile Download:
http://bigbytes.mobyus.com/tableau/census_tract_shapefiles_all.csv

ACKNOWLEDGMENTS

I would like to thank the FIMR program for providing the data for this example. I'd also like to thank my team for their continued support and expertise, and Spectrum Health Healthier Communities for giving me the opportunity to further develop my SAS abilities.

CONTACT INFORMATION

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

Ruth Kurtycz
(616) 391-2983
Ruth.Kurtycz@spectrumhealth.org