IIT School of Applied Technology

ILLINOIS INSTITUTE OF TECHNOLOGY

information technology & management

527 Data Analytics

February 16,18 2016

Week 6 Presentation

Week 6 Topic: Agenda

- Wine 3 Cluster Assignment Questions
- SAS Intro Examples
- Introduce Tax Strategy Project
- Wine 3 Cluster in SAS will be on Tuesday
- Pothole Repair Analysis Review

Week 6 Topic: Note of an Event

4th Annual Advanced Marketing Analytics Symposium

Event Date:

Feb 25, 5:00pm to 8:00pm

Location:

Stuart School of Business
Downtown Campus
565 W. Adams Street, Auditorium
Chicago, IL 60661

Register Now

Week 6 Topic: SAS Clustering Overview

- PROC CLUSTER performs hierarchical clustering of observations by using eleven agglomerative methods applied to coordinate data or distance data.
- PROC FASTCLUS finds disjoint clusters of observations by using a k-means method applied to coordinate data. PROC FASTCLUS is especially suitable for large data sets.
- PROC TREE draws tree diagrams, also called dendrograms or phenograms, by using output from the CLUSTER or VARCLUS procedure. PROC TREE can also create a data set indicating cluster membership at any specified level of the cluster tree.

Week 6 Topic: PROC INPUT – Example

- Review the two variable cluster example from Week 5
- We use INPUT data step to recreate the dataset for the cluster example

- Reference for the INPUT data step. Online: http://support.sas.com/documentation/cdl/en/lrdict/64316/HTML/default/viewer.htm#a000146292.htm
- PDF: http://www2.sas.com/proceedings/sugi29/253-29.pdf

```
data t;
                INFILE DATALINES DLM=',';
                input cid x y ;
datalines;
1,              3.966592205,    1.219366961
2,              3.909376283,    1.385870124
3,              3.994288270,    1.984305005
4,              3.468195914,    1.798214461
5,              3.174782531,    1.947220974
6,              3.065049660,    1.10814584
7,              3.996954506,    1.389417143
8,              3.782241748,    1.802353049
9,              2.541419481,    2.987577174
10,             3.529230221,    1.81079776
11,             3.716756736,    1.292000548
12,             3.736530919,    1.121806042
13,             3.626420852,    1.683066644
14,             3.125058130,    1.964899779
15,             3.164602707,    1.778464839
16,             2.337237954,    0.577319181
17,             1.724042677,    3.339108915
18,             1.254598794,    3.459779854
19,             1.993573468,    3.94314637
20,             1.111193912,    3.989153386
21,             1.938169173,    3.294909245
22,             3.182336806,    1.103533956
23,             1.784693268,    3.731690996
24,             1.667249961,    3.495303458;
run;
```

Week 6 Topic:
PROC CLUSTER – general format

General form of the CLUSTER procedure:

```
PROC CLUSTER DATA=SAS-data-set
        METHOD=method <options>;
      VAR variables;
      FREQ variable;
      RMSSTD variable;
RUN;
```

The required METHOD= option specifies the hierarchical technique to be used to cluster the observations.

Week 6 Topic: PROC CLUSTER – example options

Our example PROC CLUSTER considered is:

```
PROC CLUSTER <options>;
ID ;
VAR var1 var2 var3 … varn;
RUN;
```

Here the options control the printing, computational, and output of the procedures:

- ID variable identify observations. If the ID statement is omitted, each observation is denoted by OB*n*, where *n* is the observation number.
- NOPRINT - suppresses any printed output,
- NOEIGEN - suppresses printing of eigenvalues, specifying the NOEIGEN option saves time if the number of variables is large,
- SIMPLE - produces simple descriptive statistics for each variable,
- METHOD = - controls the clustering method used (required option), we will use CENTROID
- STANDARD - Uses the correlation matrix for computation,
- RMSSTD - displays the root-mean-square standard deviation of each cluster.
- RSQUARE - displays the R^2 and semipartial R^2 to evaluate cluster solution.
- NONORM - prevents the distances from being normalized to unit mean or unit root mean square with most methods.
- OUTTREE = -create an output dataset for cluster diagrams.

The VAR statement lists the variables to be considered as responses.

Week 6 Topic: PROC CLUSTER – Example

- We use the simple centroid option for our example and set the output for a tree diagram.
- As we have an ID variable, we set that as well.
- The variables for our example are just two: x and y.

- Reference for the PROC CLUSTER step. Online: http://support.sas.com/documentation/cdl/en/statug/63033/HTML/default/viewer.htm#cluster_toc.htm
- PDF: https://support.sas.com/documentation/cdl/en/statugclustering/61759/PDF/default/statugclustering.pdf

```
proc cluster simple noeigen
method=centroid  standard rmsstd
rsquare nonorm outtree=tree;

id cid;
var x y;

run;
```

Week 6 Topic:
PROC TREE – general format

- General form of the TREE procedure:

```
PROC TREE DATA=<dendrogram> <options>;
RUN;
```

- The TREE procedure either displays the dendrogram (LEVEL= option), or assigns the observations to a specified number of clusters (NCLUSTERS= option).

Week 6 Topic: PROC TREE – Example

- We generate a tree diagram for the 2 clusters observed.

- Reference for the PROC TREE step. Online: http://support.sas.com/documentation/cdl/en/statug/63033/HTML/default/viewer.htm#statug_tree_sect004.htm
- PDF: https://support.sas.com/documentation/cdl/en/statugtree/61843/PDF/default/statugtree.pdf

```
proc tree data=tree out=clus2
nclusters=2;

id cid;
copy x y;

run;
```

Week 6 Topic:
PROC SORT & PRINT – Example

- We sort and print the results by cluster

```
proc sort; by cluster;

proc print; by cluster;

var cid x y;
title2 '2-cluster solution';

run;
```

Week 6 Topic: PROC SGPLOT – Example

- We generate a scatter plot with data sets x and y using the cluster designations observed

- Reference for the PROC SGPLOT step. Online: Online: https://support.sas.com/documentation/cdl/en/grstatproc/62603/HTML/default/viewer.htm#sgplot-ov.htm
- PDF: http://support.sas.com/resources/papers/proceedings10/154-2010.pdf

```
proc sgplot;
scatter y=y x=x / group=cluster;

title 'Single Linkage Cluster Analysis';
title2 'of Data Containing Well-Separated Clusters';

run;
```

Week 6 Topic: PROC FASTCLUS - overview

- By default, the FASTCLUS procedure uses Euclidean distances, so the cluster centers are based on least squares estimation. This kind of clustering method is often called a k-means model, since the cluster centers are the means of the observations assigned to each cluster when the algorithm is run to complete convergence. Each iteration reduces the least squares criterion until convergence is achieved.
- Often there is no need to run the FASTCLUS procedure to convergence. PROC FASTCLUS is designed to find good clusters (but not necessarily the best possible clusters) with only two or three passes through the data set.
- The initialization method of PROC FASTCLUS guarantees that, if there exist clusters such that all distances between observations in the same cluster are less than all distances between observations in different clusters, and if you tell PROC FASTCLUS the correct number of clusters to find, it can always find such a clustering without iterating. Even with clusters that are not as well separated, PROC FASTCLUS usually finds initial seeds that are sufficiently good that few iterations are required. Hence, by default, PROC FASTCLUS performs only one iteration.
- The initialization method used by the FASTCLUS procedure makes it sensitive to outliers. PROC FASTCLUS can be an effective procedure for detecting outliers because outliers often appear as clusters with only one member.

Week 6 Topic:
PROC FASTCLUS – general format

- General form of the FASTCLUS procedure:

```
PROC FASTCLUS DATA=SAS-data-set
                <MAXC=>|<RADIUS=><options>;
    VAR variables;
RUN;
```

- Because PROC FASTCLUS produces relatively little output, it is often a good idea to create an output data set, and then use other procedures such as PROC MEANS, PROC SGPLOT, PROC DISCRIM, or PROC CANDISC to study the clusters.

Week 6 Topic: PROC FASTCLUS – example options

Our example for PROC FASTCLUS considered is:

```
PROC FASTCLUS MAXCLUSTERS=n RADIUS=t <options>;
VAR var1 var2 var3 … varn;
RUN;
```

Here the options control the printing, computational, and output of the procedures:

- ID variable identify observations. If the ID statement is omitted, each observation is denoted by OB*n*, where *n* is the observation number.
- MAXCLUSTERS=n specifies the maximum number of clusters permitted. If you omit the MAXCLUSTERS= option, a value of 100 is assumed
- RADIUS=t establishes the minimum distance criterion for selecting new seeds. No observation is considered as a new seed unless its minimum distance to previous seeds exceeds the value given by the RADIUS= option. The default value is 0.
- REPLACE= specifies seed replacement method. If you specify the REPLACE=RANDOM option, the RADIUS= option is ignored.
- MAXITER= specifies maximum number of iterations
- DISTANCE displays distances between cluster centers
- LIST displays cluster assignments for all observations

The VAR statement lists the variables to be considered as responses.

Week 6 Topic: PROC FASTCLUS – Example (cont.)

- We generate a tree diagram for the 2 clusters observed.
- We choose radius=0 as minimum separation of data points and select the full seed replacement scheme
- Reference for the PROC FASTCLUS step. Online: http://support.sas.com/documentation/cdl/en/statug/63033/HTML/default/viewer.htm#fastclus_toc.htm
- PDF: https://support.sas.com/documentation/cdl/en/statugfastclus/61784/PDF/default/statugfastclus.pdf

```
proc fastclus radius=0 replace=full
maxclusters=2 maxiter=20 list
distance;

id cid;
var x y;

run;
```

Week 6 Topic: SAS Cheat Sheets

Good 2 page cheat sheet:
https://www.ualberta.ca/~ahamann/teaching/renr480/SAS-Cheat.pdf

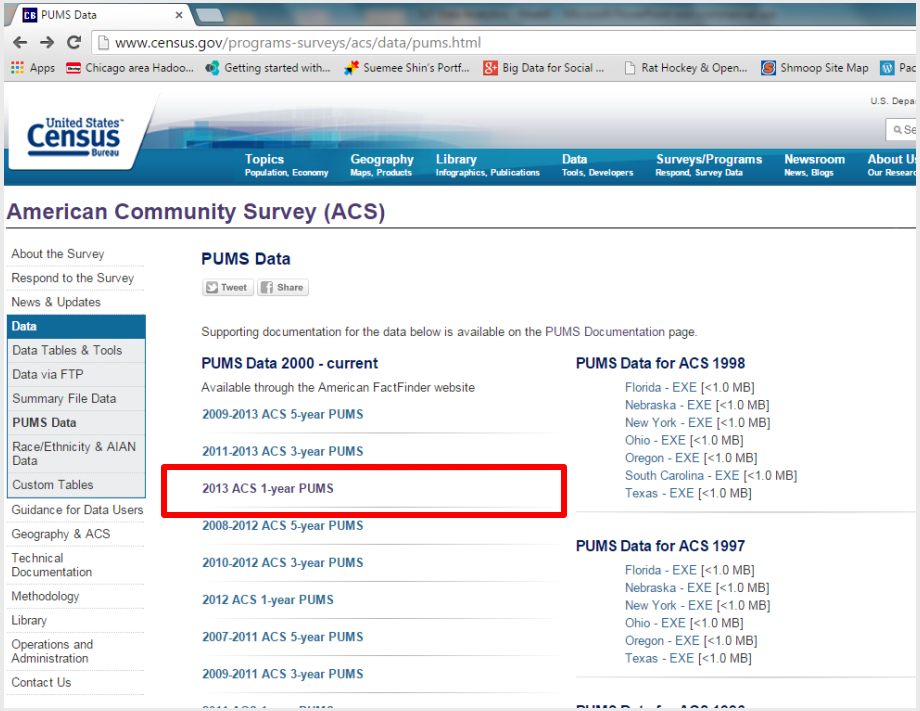
Good guide to date formats:
http://support.sas.com/publishing/bbu/companion_site/update/appendix_59411.pdf

Week 6 Topic:
Tax Strategy Project: Data

AMERICAN COMMUNITY SURVEY
2013 ACS 1-YEAR PUMS FILES

Prepared by:
American Community Survey Office
U.S. Census Bureau
January 15, 2015

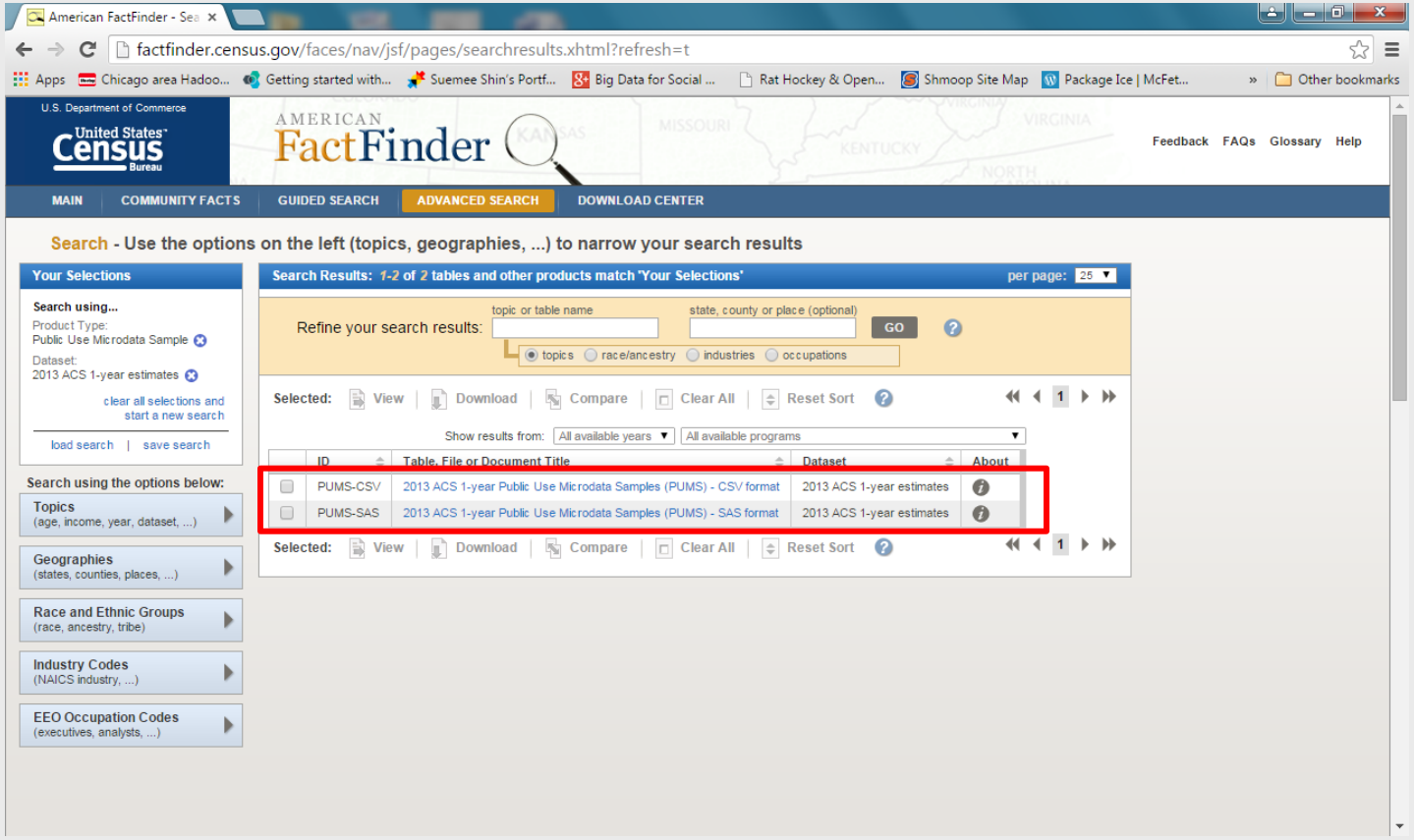
PUMS Data Link:
http://www.census.gov/programs-surveys/acs/data/pums.html



ITM - 527

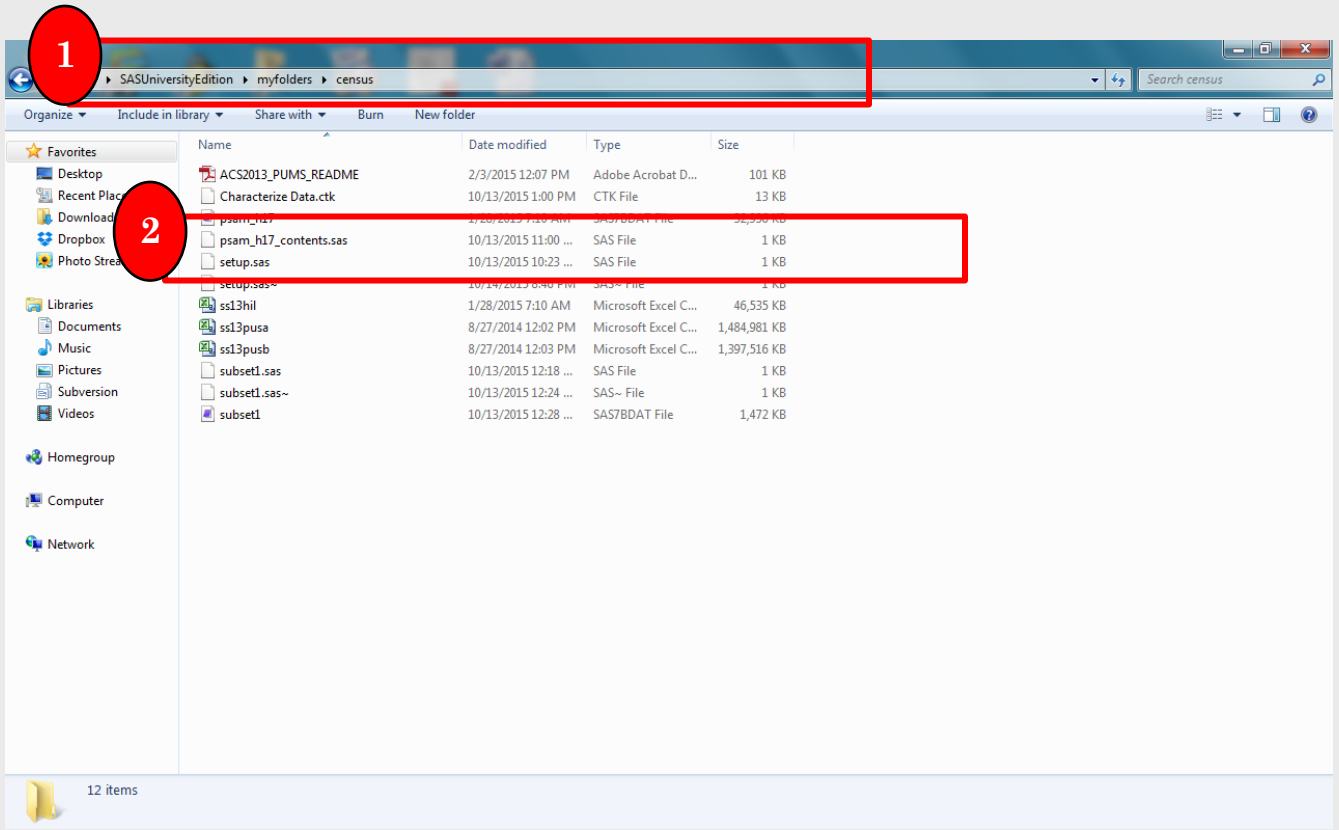
Week 6 Topic:
Tax Strategy Project: File Formats

- **If using SAS, copy the SAS file into your SAS data folder to start**
- **If using other than SAS, upload the CSV file**



Week 6 Topic:
Final Project: Folder set up

- **Create a census folder in your SAS data folder**
- **Copy the SAS census download file into your SAS data folder**



Week 6 Topic:
2013 IL ACS PUMS Metadata

Data Dictionary:
http://www2.census.gov/programs-surveys/acs/tech_docs/pums/data_dict/PUMSDataDict13.pdf

Focus on the following variables to start:

- **MRGP:** First mortgage payment (monthly amount)
- **HINCP:** Household income (past 12 months)
- **TAXP:** Property taxes (yearly amount)
- **VALP:** Property value

Week 6 Topic: Feedback on Pothole Repair Analysis II

- **Objectives:** Can be a paragraph with a single goal or can be a set of bullets with multiple goals, either way these need to be addressed in the analysis and conclusions.
- **Scope:** If you state that you only analyzed a subset of the dataset, you need to state what dataset you kept/are in scope for the analysis.
- **Data Exceptions:** Did everyone catch that only half of the performance metrics data is available for 2011? Did everyone catch that 2014 had different titles e.g., asphalt laborers? When quoting on numbers that are different in scope compared to others, you need to caveat/capture that exception whenever you quote/state said numbers.
- **Phrasing:** Try to avoid using phrases like “very evident”, “it is clear”, “clearly evident”, misuse of “maximum”, etc.
- **Table, Charts, and Graph Labeling:** If you are using figure # notation to reference, be sure to also include a sentence describing it. E.g., Figure 1: Budget comparison over 4 years for Asphalt related entries. Of course, remember to label all axis, data (via legend or otherwise), and add units if not obvious e.g., Miles, Fahrenheit, etc.

Week 6 Topic: Feedback Continued

- **Analysis:** Not just quoting of facts or describing what I am already seeing. E.g., 2014 was the coldest year with one of the highest number of potholes filled but had lowest allocated budget.
- **Metadata:** Only list key data items used in the analysis. No need to copy/paste Excel. There is really no need to copy/paste Excel at any point. E.g., Was street address used?
- **Weather data:** Intention was not to provide a weather report. Should have been correlated/associated to budget or performance.
- **Zip Code Analysis:** Great analysis but should have gone one step further e.g., include a map of the zip codes or state what area it represents to get a sense for why we see more potholes in those areas.
- **Pie vs Histograms:** Try to stay small (less than ~12 observations) for both. If you have a set of data that scales so that the smaller values are extremely small slices of the pie, you may consider using a histogram instead with % breakouts. When dealing with small slices, you may need to add callout text boxes which is labor intensive but necessary.

Attach XLS (zipped) after PPT files!

Week 6 Topic:
Feedback Continued - Conclusions

- **Conclusions:** Should NOT be:
  - Summary of data processing
  - Opinions and hypotheticals
  - List of facts already shown in previous slides
  - List of future items
- **Conclusions:** Should be:
  - Address all stated objectives and goals
  - New finding/derivation based on analysis in previous slides
  - Summary of key analysis with facts and figures, numbers to back up the key findings

Week 6 Topic: Pothole Repair Analysis

We review pothole repair budget and service request metrics data from the past 4 years (2011 ~ 2014) to determine whether enough budget was appropriated to meet the Target Response Days of 7.

- Pothole Repair (PR) data was acquired from data.gov. Category information did not match year to year. Some assumptions were made to gather the budget amounts which are documented in the Excel workbook.
- Weather data (average temperatures and # of snow days) was gathered from wunderground.com for the 4 years.
- Main assumptions for analysis include:
  - Only Asphalt related budget amounts are considered for the analysis
  - Only labor costs are considered as majority of the cost for pothole repairs is assumed to be labor related

Week 6 Topic:
PR Filtering Criteria

Not all file header names were the same for all four years' data files and some of the hierarchies were categorized differently e.g., sub-section description.

- Most differences were with the sub-section description field which houses the "Asphalt" category found in the 2011 data file.
- Based on the titles in the "Asphalt" category for year 2011, we deduce that "Street and Alley Repair Unit", "Street and Alley Resurfacing Unit", and "Street Repair and Resurfacing Unit" categories should be used for years 2012, 2013, and 2014, in addition to "Asphalt". The numbers in the following budget analysis uses these filtering values.
- For the 311 metric data, duplicate entries were removed and only MOST RECENT ACTION= COMPLETED or POTHOLE PATCHED were included in the analysis.

Week 6 Topic:
PR Review of Budget, Metrics & Weather

2014 budget for pothole repair was ill-matched to the extreme weather:

Year	Budget for Pothole Repair (Labor)*	Average days to Complete Request**	# of Completed Requests**	# of Pothole Repair Service Requests to 311
2011	$7,036,432	17***	20,294***	52,952
2012	$7,995,570	8	39,567	31,966
2013	$8,157,000	21	52,208	43,312
2014	**$7,391,010**	**39**	**69,176**	**51,372**

*Pothole repair budget amounts were derived by gathering labor and temporary help costs from Chicago Dept. of Transportation's In-House Construction budget. Assumption is that main costs in repairing potholes is labor related. Also, as potholes are primarily repaired with Asphalt, only those budget amounts related to Asphalt were included.

**A request recorded in the Performance Metrics data is assumed to be the same as a request or call in the 311 Metrics data.

***Only Jun~Dec data available for 2011

Year	Average Temperature for the Months: Nov, Dec, Jan, Feb, Mar	Number of Snow Days for the Months: Nov, Dec, Jan, Feb, Mar
2011	34	51
2012	40	37
2013	31	65
2014	**28**	**72**

Week 6 Topic: PR Budget Analysis

- 2014 budget for pothole repairs were ill-matched to the extreme weather experienced.
- 2013 with similar weather conditions but with just ~$1 million in additional funding reduced request response times by half.
- We derive the cost per request for the past 4 years as follows:

Year	Budget for Pothole Repair (B)	# of Pothole Repair Service Requests to 311 (A)	Cost per Request (B/A)
2011	$7,036,432	52,952	132.88
2012	$7,995,570	31,966	**250.13**
2013	$8,157,000	43,312	188.33
2014	$7,391,010	51,372	143.87

- If the goal is to stay as close to the 7 day response target as possible, 2012 cost per request of ~$250 can be used to derive ideal budget amounts for pothole repairs.
- The following slide uses the 2012 rate to determine the ideal budget amounts.

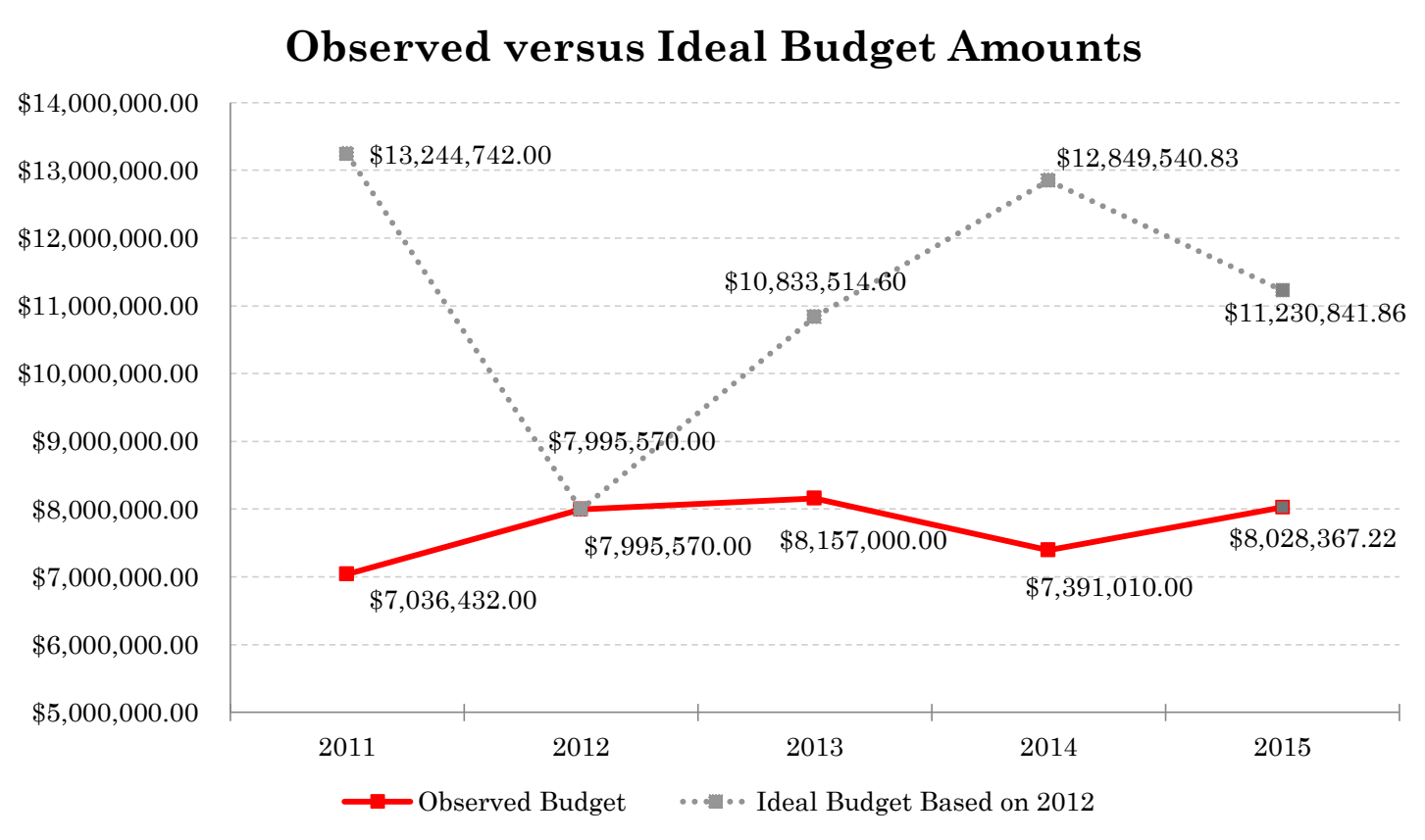
Week 6 Topic:
PR Budget Analysis - Conclusion

If we take an average of the # of calls in the past 4 years as the basis for 2015 request estimates, then the ideal 2015 budget for pothole repairs should be in the order of ***$11,000,000*** (44,901 requests at 2012 cost per request level) to meet the 7 day target.

Year	Budget	Ideal Budget Based on 2012	# of Requests	Average Budget per Request
2011	$7,036,432.00	$13,244,742.00	52,952	$132.88
2012	$7,995,570.00	$7,995,570.00	31,966	$250.13
2013	$8,157,000.00	$10,833,514.60	43,312	$188.33
2014	$7,391,010.00	$12,849,540.83	51,372	$143.87
2015	**$8,028,367.22**	**$11,230,841.86**	**44,901**	**$178.80**

Week 6 Topic:
PR Budget Analysis - Conclusion

If we take an average of the # of calls in the past 4 years as the basis for 2015 request estimates, then the ideal 2015 budget for pothole repairs should be in the order of ***$11,000,000*** (44,901 requests at 2012 cost per request level) to meet the 7 day target.



Week 5 Topic Review: SAS Framework and File Types

SAS framework:

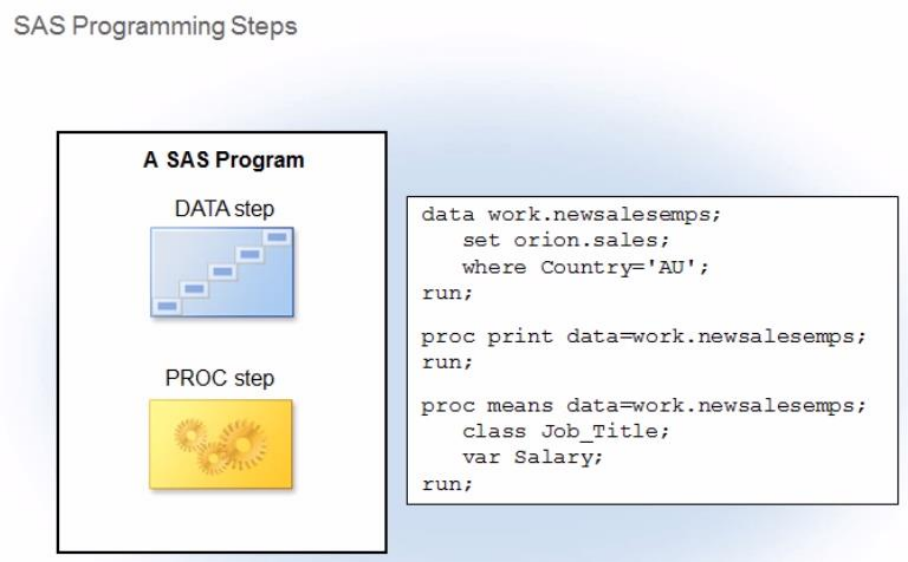
- **Access data**: Using SAS, you can read any kind of data.
- **Manage data**: SAS gives you excellent data management capabilities
- **Analyze data**: For statistical analysis, SAS is the gold standard.
- **Present data**: You can use SAS to present your data meaningfully.

Three major file types:

- **Raw data files** contain data that has not been processed by any other computer program. They are text files that contain one record per line, and the record typically contains multiple fields. Raw data files aren't reports; they are unformatted text.
- **SAS data sets** are specific to SAS. A SAS data set is data in a form that SAS can understand. Like raw data files, SAS data sets contain data. But in SAS data sets, the data is created only by SAS and can be read only by SAS.
- **SAS program files** contain SAS programming code. These instructions tell SAS how to process your data and what output to create. You can save and reuse SAS program files.

Week 5 Topic Review: SAS Steps

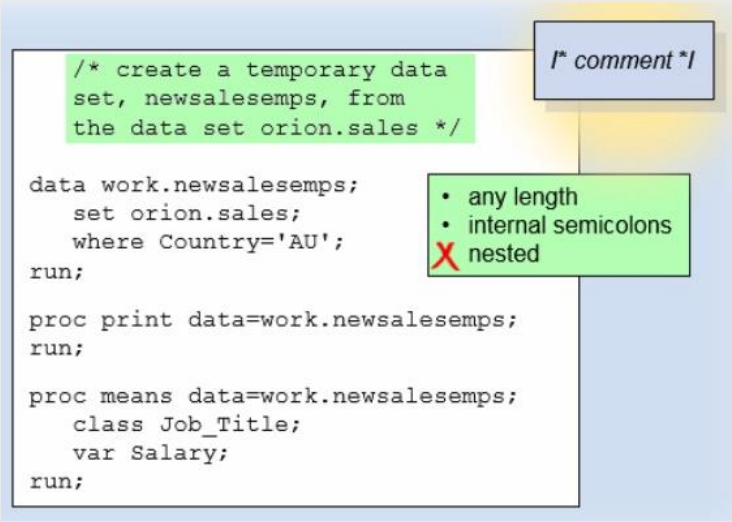
- A SAS program consists of DATA steps and PROC steps. A SAS programming step is comprised of a sequence of statements. Every step has a beginning and ending step boundary. SAS compiles and executes each step independently, based on the step boundaries.
- A SAS program can also contain global statements, which are outside DATA and PROC steps, and typically affect the SAS session. A TITLE statement is a global statement. After it is defined, a title is displayed on every report, unless the title is cleared or canceled.
- SAS statements usually begin with an identifying keyword, and always end with a semicolon. SAS statements are free format and can begin and end in any column. A single statement can span multiple lines, and there can be more than one statement per line. Unquoted values can be lowercase, uppercase, or mixed case. This flexibility can result in programs that are difficult to read.

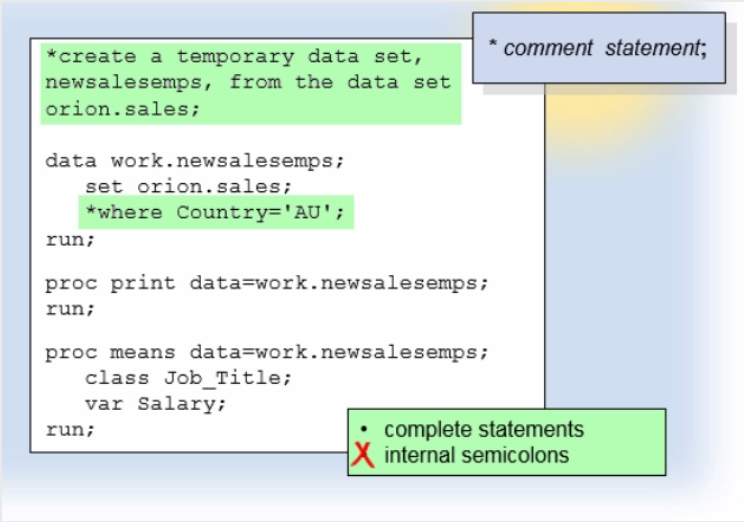


Week 5 Topic Review: SAS Comments

- Comments are used to document a program and to mark SAS code as non-executing text. There are two types of comments: *block comments* and *comment statements*.

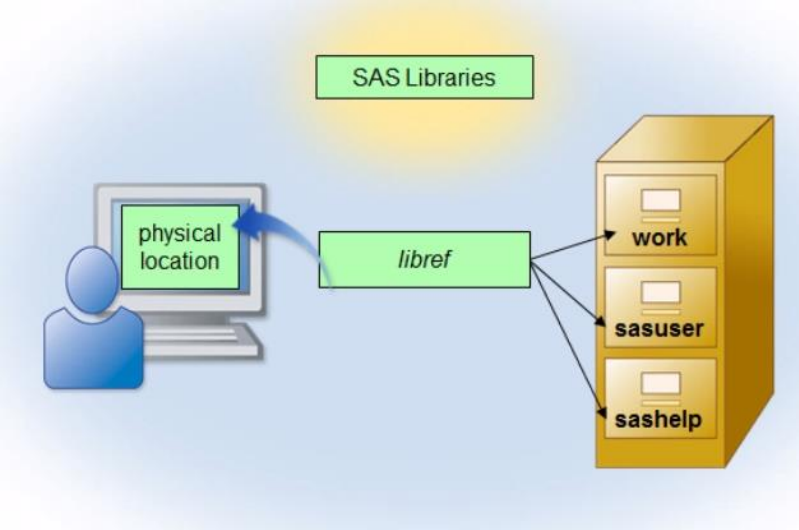
/* comment */
**** comment statement*;**

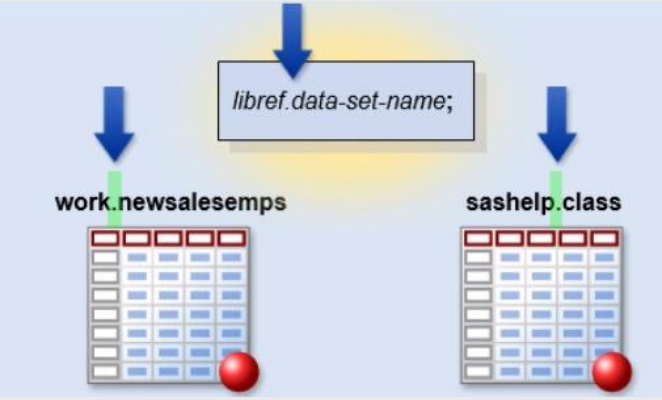




Week 5 Topic Review: SAS Libraries

- SAS data sets are stored in SAS libraries. A SAS library is a collection of one or more SAS files that are recognized by SAS. SAS automatically provides one temporary and at least one permanent SAS library in every SAS session.
- **Work** is a temporary library that is used to store and access SAS data sets for the duration of the session. **Sasuser** and **sashelp** are permanent libraries that are available in every SAS session.
- You refer to a SAS library by a library reference name, or libref. A libref is a shortcut to the physical location of the SAS files.
- All SAS data sets have a two-level name that consists of the libref and the data set name, separated by a period. Data sets in the **work** library can be referenced with a one-level name, consisting of only the data set name, because **work** is the default library. Data sets in permanent libraries must be referenced with a two-level name.





Week 5 Topic Review: SAS Libraries (cont.)

- You can create and access your own SAS libraries. User-defined libraries are permanent but are not automatically available in a SAS session. You must assign a libref to a user-created library to make it available. You use a LIBNAME statement to associate the libref with the physical location of the library, that is, the physical location of your data. You can submit the LIBNAME statement alone at the start of a SAS session, or you can store it in a SAS program so that the SAS library is defined each time the program runs. If your program needs to reference data sets in multiple locations, you can use multiple LIBNAME statements.
- In an interactive SAS session, a libref remains in effect until you cancel it, change it, or end your SAS session. To cancel a libref, you submit a LIBNAME statement with the CLEAR option. This clears or disassociates a libref that was previously assigned. To specify a different physical location, you submit a LIBNAME statement with the same libref name but with a different filepath.
- When a SAS session ends, everything in the **work** library is deleted. The librefs are also deleted. Remember that the contents of permanent libraries still exist in in the operating environment, but each time you start a new SAS session, you must resubmit the LIBNAME statement to redefine a libref for each user-created library that you want to access.

LIBNAME *libref 'SAS-library'* <*options*>**;**

LIBNAME *libref* **CLEAR;**

Week 5 Topic Review: SAS PROC CONTENTS AND PRINT

- Use PROC CONTENTS with *libref*._ALL_ to display the contents of a SAS library. The report will list all the SAS files contained in the library, as well as the descriptor portion of each data set in the library. Use the NODS option in the PROC CONTENTS statement to suppress the descriptor information for each data set.

```
PROC CONTENTS DATA=libref._ALL_ NODS;
RUN;
```

```
PROC CONTENTS DATA=libref.SAS-data-set;
RUN;
```

- After associating a libref with a permanent library, you can write a PROC PRINT step to display a SAS data set within the library.

```
PROC PRINT DATA=libref.SAS-data-set;
RUN;
```

Week 5 Topic Review: SAS Data Sets

- SAS data sets are specially structured data files that SAS creates and that only SAS can read. A SAS data set is displayed as a table composed of variables and observations. A SAS data set contains a descriptor portion and a data portion.
- The descriptor portion contains general information about the data set (such as the data set name and the number of observations) and information about the variable attributes (such as name, type, and length). There are two types of variables: **character and numeric**. A character variable can store any value and can be up to 32,767 characters long. Numeric variables store numeric values in floating point or binary representation in 8 bytes of storage by default. Other attributes include formats, informats, and labels. You can use PROC CONTENTS to browse the descriptor portion of a data set.
- The data portion contains the data values. Data values are either **character or numeric**. A valid value must exist for every variable in every observation in a SAS data set. A missing value is a valid value in SAS. A missing character value is displayed as a ***blank***, and a missing numeric value is displayed as a ***period***. You can specify an alternate character to print for missing numeric values using the MISSING= SAS system option. You can use PROC PRINT to display the data portion of a SAS data set.
- SAS variable and data set names must be 1 to 32 characters in length and start with a **letter or underscore**, followed by letters, underscores, and numbers. Variable names are not case sensitive.

Week 6 Topic: VAR and SUM Statements

- You can use the VAR statement in a PROC PRINT step to subset the variables in a report. You specify the variables to include and list them in the order in which they are to be displayed.
- You can use the SUM statement in a PROC PRINT step to calculate and display report totals for the requested numeric variables.

```
PROC PRINT DATA=SAS-data-set;
        VAR variable(s);
        SUM variable(s);
RUN;
```

Week 6 Topic: WHERE Statement

- The WHERE statement in a PROC PRINT step subsets the observations in a report. When you use a WHERE statement, the output contains only the observations that meet the conditions specified in the WHERE expression. This expression is a sequence of operands and operators that form a set of instructions that define the condition. The operands can be constants or variables. Remember that variable operands must be defined in the input data set. Operators include comparison, arithmetic, logical, and special WHERE operators.

Comparison:

Symbol(s)	Mnemonic	Definition
=	EQ	equal to
^= ¬= ~=	NE	not equal to
>	GT	greater than
<	LT	less than
>=	GE	greater than or equal to
<=	LE	less than or equal to
	IN	equal to one of a list

Examples

```
where Gender='M';
where Gender eq 'M';
where Salary ne .;
where Salary>50000;
where Salary lt 50000;
where Salary<=60000;
where Country in ('AU','US');
```

Week 6 Topic: WHERE Statement (Cont.)

Arithmetic:

Symbol	Definition
**	exponentiation
*	multiplication
/	division
+	addition
-	subtraction

Example

```
where Salary+Bonus<=10000;
```

Logical:

WHERE *where-expression-1* AND | OR *where-expression-n*;

Symbol(s)	Mnemonic	Definition
&	AND	logical *and*
|	OR	logical *or*
^ ¬ ~	NOT	logical *not*

Examples

```
where Country ne 'AU' and Salary>=50000;
where Gender eq 'M' or Salary ge 50000;
where Country='AU' | Country='US';
where Country in ('AU' 'US');
where Country not in ('AU','US');
```

Week 6 Topic: WHERE Statement (Cont.)

Contains:

WHERE *where-expression*;

Symbol	Mnemonic	Definition
?	CONTAINS	includes a substring

Examples

```
where Country='AU' and
      Job_Title contains 'Rep';

where Country='AU' and
      Job_Title ? 'Rep';
```

case sensitive

Mnemonic:

Mnemonic	Definition
BETWEEN-AND	an inclusive range
WHERE SAME AND	augment a where expression
IS NULL	a missing value
IS MISSING	a missing value
LIKE	matches a pattern

Symbol	Replaces
%	any number of characters
_	one character

1) *Use of ~not~ to exclude*
2) *Use of ~ same and~ to augment*
3) *IS NULL and IS MISSING can be used for both numeric and character variables*

Week 6 Topic: Sorting and Grouping

- The SORT procedure sorts the observations in a data set. You can sort on one variable or multiple variables, sort on character or numeric variables, and sort in ascending or descending order. By default, SAS replaces the original SAS data set unless you use the OUT= option to specify an output data set. PROC SORT does not generate printed output.
- Every PROC SORT step must include a BY statement to specify one or more BY variables. These are variables in the input data set whose values are used to sort the data. By default, SAS sorts in ascending order, but you can use the keyword DESCENDING to specify that the values of a variable are to be sorted in descending order. When your SORT step has multiple BY variables, some variables can be in ascending and others in descending order.
- You can also use a BY statement in PROC PRINT to display observations grouped by a particular variable or variables. The groups are referred to as BY groups. Remember that the input data set must be sorted on the variables specified in the BY statement.

```
PROC SORT DATA=input-SAS-data-set
                <OUT=ouput-SAS-data-set>;
     BY <DESCENDING> by-variable(s);
RUN;
```

Week 6 Topic:
ID, TITLE, FOOTNOTE Statement

- You can use the ID statement in a PROC PRINT step to specify a variable to print at the beginning of the row instead of an observation number. The variable that you specify replaces the Obs column.

ID *variable(s)*;

- You can enhance a report by adding titles, footnotes, and column labels. Use the global TITLE statement to define up to 10 lines of titles to be displayed at the top of the output from each procedure. Use the global FOOTNOTE statement to define up to 10 lines of footnotes to be displayed at the bottom of the output from each procedure.

TITLE*n* '*text*';
FOOTNOTE*n* '*text*';

- Titles and footnotes remain in effect until you change or cancel them, or until you end your SAS session. Use a null TITLE statement to cancel all titles, and a null FOOTNOTE statement to cancel all footnotes.

Week 6 Topic: LABEL Statement

- Use the LABEL statement in a PROC PRINT step to define temporary labels to display in the report instead of variable names. Labels can be up to 256 characters in length. Most procedures use labels automatically, but PROC PRINT does not. Use the LABEL option in the PROC PRINT statement to tell SAS to display the labels. Alternatively, the SPLIT= option tells PROC PRINT to use the labels and also specifies a split character to control line breaks in column headings.

```
PROC PRINT DATA=SAS-data-set LABEL;
      LABEL variable='label'
            variable='label'
           ... ;
RUN;
```

```
SPLIT='split-character';
```

Week 6 Topic: FORMAT Statement

- A format is an instruction that tells SAS how to display data values in output reports. You can add a FORMAT statement to a PROC PRINT step to specify temporary SAS formats that control how values appear in the report. There are many existing SAS formats that you can use. Character formats begin with a dollar sign, but numeric formats do not.

FORMAT *variable(s) format*;

- SAS stores date values as the number of days between January 1, 1960, and a specific date. To make the dates in your report recognizable and meaningful, you must apply a SAS date format to the SAS date values.

Week 6 Topic: FORMAT Statement Examples

Format	Stored Value	Displayed Value
MMDDYY6.	0	010160
MMDDYY8.	0	01/01/60
MMDDYY10.	0	01/01/1960
DDMMYY6.	365	311260
DDMMYY8.	365	31/12/60
DDMMYY10.	365	31/12/1960

Format	Stored Value	Displayed Value
$4.	Programming	Prog
12.	27134.5864	27135
12.2	27134.5864	27134.59
COMMA12.2	27134.5864	27,134.59
DOLLAR12.2	27134.5864	$27,134.59
COMMAX12.2	27134.5864	27.134,59
EUROX12.2	27134.5864	€27.134,59

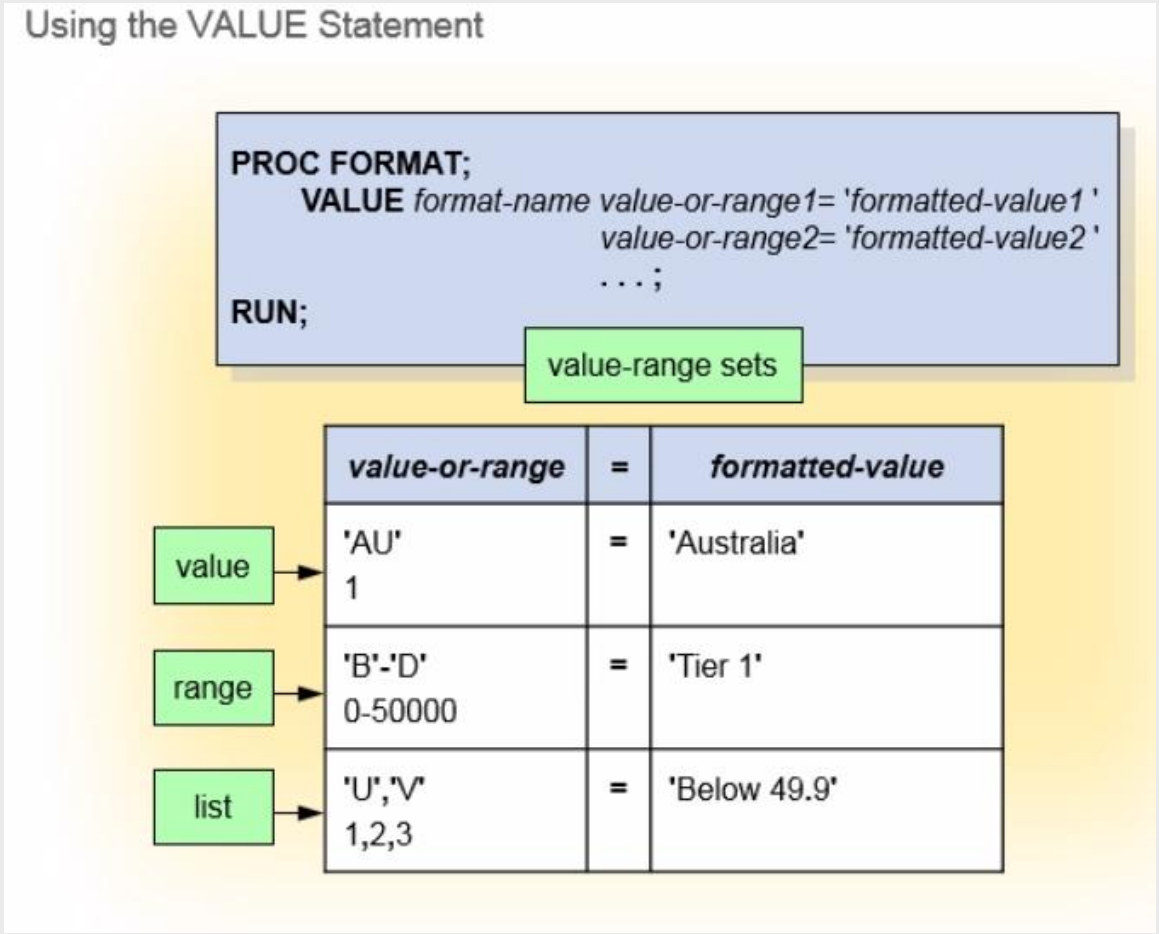
Format	Definition
$*w*.	writes standard character data.
w.d	writes standard numeric data.
COMMA*w.d*	writes numeric values with a comma that separates every three digits and a period that separates the decimal fraction.
DOLLAR*w.d*	writes numeric values with a leading dollar sign, a comma that separates every three digits, and a period that separates the decimal fraction.
COMMAX*w.d*	writes numeric values with a period that separates every three digits and a comma that separates the decimal fraction.
EUROX*w.d*	writes numeric values with a leading euro symbol (€), a period that separates every three digits, and a comma that separates the decimal fraction.

Week 6 Topic: User Defined FORMAT Statement

- You can create your own user-defined formats. When you create a user-defined format, you don't associate it with a particular variable or data set. Instead, you create it based on values that you want to display differently. The formats will be available for the remainder of your SAS session. You can apply user-defined formats to a specific variable in a PROC PRINT step.
- You use the FORMAT procedure to create a format. You assign a format name that can have up to 32 characters. The name of a character format must begin with a dollar sign, followed by a letter or underscore, followed by letters, numbers, and underscores. Names for numeric formats must begin with a letter or underscore, followed by letters, numbers, and underscores. A format name cannot end in a number and cannot be the name of a SAS format.
- You use a VALUE statement in a PROC FORMAT step to specify the way that you want the data values to appear in your output. You define value-range sets to specify the values to be formatted and the formatted values to display instead of the stored value or values. The value portion of a value-range set can include an individual value, a range of values, a list of values, or a keyword. The keyword OTHER is used to define a value to display if the stored data value does not match any of the defined value-ranges.
- When you define a numeric format, it is often convenient to use numeric ranges in the value-range sets. Ranges are inclusive by default. To exclude the endpoints, use a less-than symbol after the low end of the range or before the high end.
- The LOW and HIGH keywords are used to define a continuous range when the lowest and highest values are not known. Remember that for character values, the LOW keyword treats missing values as the lowest possible values. However, for numeric values, LOW does not include missing values.

```
PROC FORMAT;
      VALUE format-name value-or-range1='formatted-value1'
                        value-or-range2='formatted-value2'
                        ...;
RUN;
```

Week 6 Topic: Using VALUE Statement



Week 6 Topic: Creating new datasets

- You use a DATA step to create a new SAS data set from an existing SAS data set. The DATA step begins with a DATA statement, which provides the name of the SAS data set to create. Include a SET statement to name the existing SAS data set to be read in as input.
- You use the WHERE statement to subset the input data set by selecting only the observations that meet a particular condition. To subset based on a SAS date value, you can use a SAS date constant in the WHERE expression. SAS automatically converts a date constant to a SAS date value.

```
DATA output-SAS-data-set;
     SET input-SAS-data-set;
     WHERE where-expression;
RUN;
```

- By default, the SET statement reads all of the observations and variables from the input data set and writes them to the output data set. You can customize the new data set by selecting only the observations and variables that you want to include. You can use a WHERE statement to select the observations, as long as the variables included in the condition come from the input data set. You can use aDROP statement to list the variables to exclude from the new data set, or use a KEEP statement to list the variables to include. If you use a KEEP statement, you must include every variable to be written, including any new variables.

```
DROP variable-list;
KEEP variable-list;
```

Week 6 Topic: Creating new datasets (cont.)

- You can subset the original data set with a WHERE statement for variables that are defined in the input data set, and a subsetting IF statement for new variables that are created in the DATA step. Remember that, although IF expressions are similar to WHERE expressions, you cannot use special WHERE operators in IF expressions.

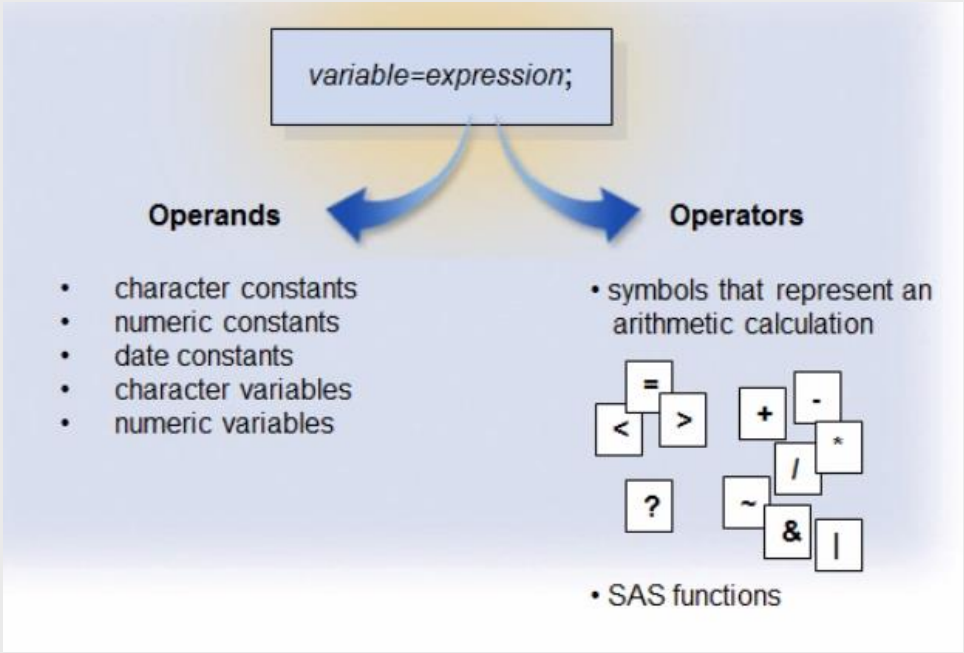
IF *expression*;

- To subset observations in a PROC step, you must use a WHERE statement. You cannot use a subsetting IF statement in a PROC step. To subset observations in a DATA step, you can always use a subsetting IF statement. However, a WHERE statement can make your DATA step more efficient because it subsets on input.
- When you use the LABEL statement in a DATA step, SAS permanently associates the labels to the variables by storing the labels in the descriptor portion of the data set. Using a FORMAT statement in a DATA step permanently associates formats with variables. The format information is also stored in the descriptor portion of the data set. You can use PROC CONTENTS to view the label and format information. PROC PRINT does not display permanent labels unless you use the LABEL or SPLIT= option.

Week 6 Topic: Assignment Statement

- You use an assignment statement to create a new variable. The assignment statement evaluates an expression and assigns the resulting value to a new or existing variable. The expression is a sequence of operands and operators. If the expression includes arithmetic operators, SAS performs the numeric operations based on priority, as in math equations. You can use parentheses to clarify or alter the order of operations.

variable=expression;



Week 6 Topic: Assignment Statement Examples

- Watch for order of execution:

Symbol	Definition	Priority
**	exponentiation	I
*	multiplication	II
/	division	II
+	addition	III
-	subtraction	III

Example	Type
`Salary=26960;`	numeric constant
`Gender='F';`	character constant
`Hire_Date='21JAN1995'd;`	date constant
`Bonus=Salary*.10;`	arithmetic expression
`BonusMonth=month(Hire_Date);`	SAS function