

Introduction to SAS

For GREAT LAKES : PGP-BABI

By Preeti Pandhu

Founder & Director
DataScienceLab™

DataScienceLab

Know your trainer Preeti Pandhu

- 16+ years of experience
- Masters of Science: Statistics from University of Pune
- Cracked PHD entrance , attended 6 months compulsory course and did 6 months of research
- Research interest area: spatial data mining
- TOP three SAS certifications
- Worked as lecturer, business analyst, research, IP and analytical QA and trainer
- Worked with IDEAS a SAS company,SAS
- International trainer for SAS ,Pan India- R , analytics, SAS trainings
- Founder and director of DataScienceLab
- More than 70 trainings
- More than 750 participants ,338 international (covered since April 2014)
- Technical reviewed two books of US publication

Outline

- ❖ Getting started with SAS
- ❖ Accessing data in SAS
- ❖ Managing and manipulating data
- ❖ Data exploration , validation and analysis
- ❖ Data visualization

Getting started with SAS

DataScienceLab

Outline

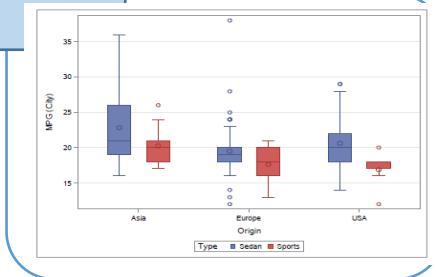
- Introduction to Base SAS
- Programming basics
- Getting help from SAS help facility
- SAS Data structure
- Referencing storage-SAS Library

Introduction to Base SAS

DataScienceLab

Introduction to Base SAS

Report and create graphics



Visualization

Analyze data using various procedures

Analysis



Accessing



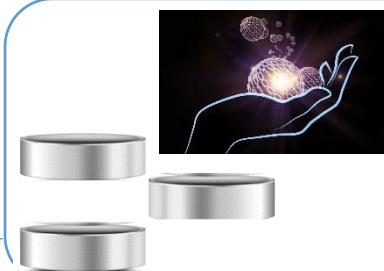
Access to virtually any data source such as DB2, Oracle, SYBASE, Teradata, SAP, and Microsoft Excel



Store data
Support for most widely used character encodings for globalization

Storing

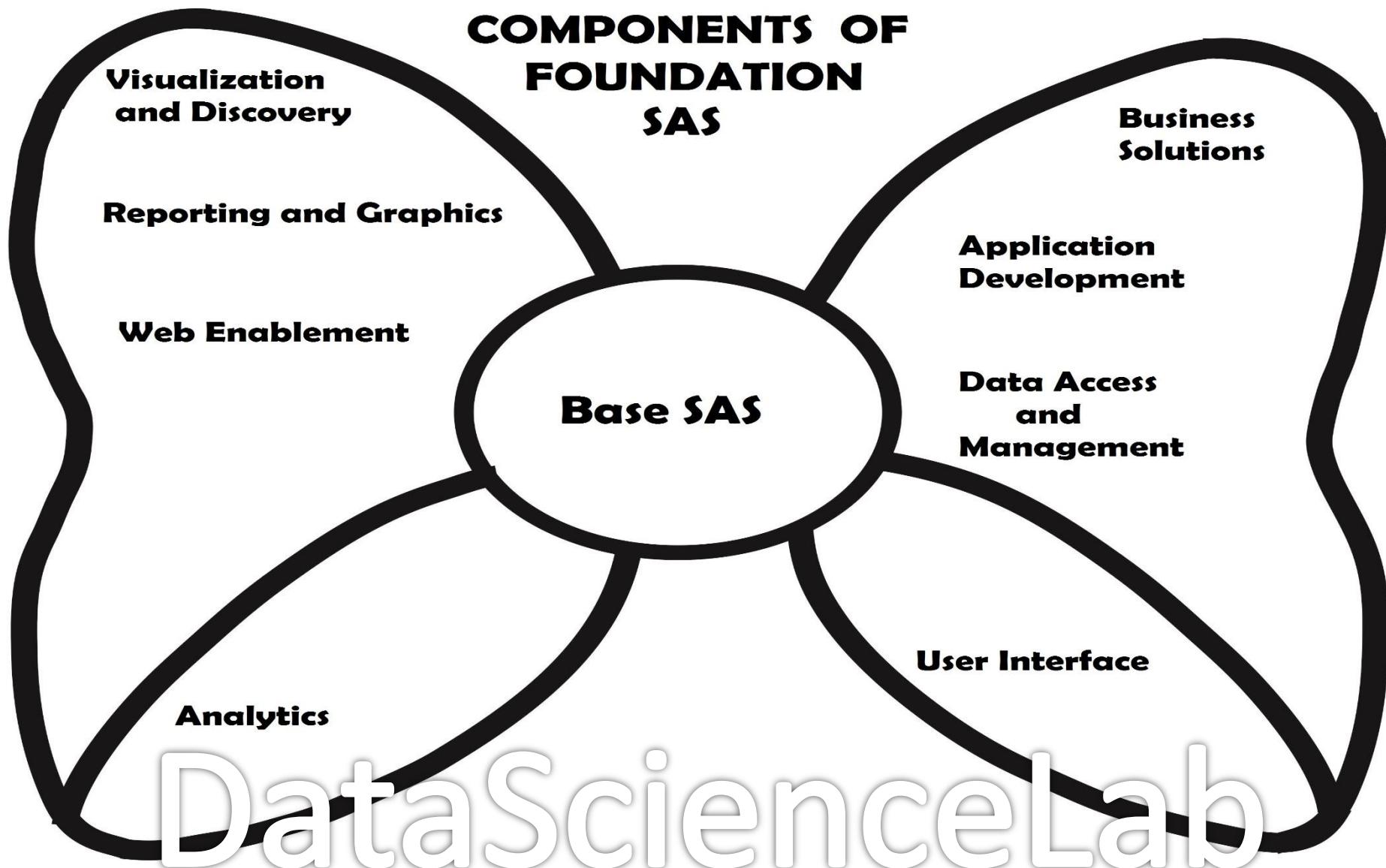
DATA



Transform and manage data

Manipulation & handling

Introduction to Base SAS



The SAS system



ORACLE®
SOLARIS

Portable : Multi Vendor Architecture

The SAS system

INGRES™

SYBASE®
An **SAP** Company

 **MySQL®**



Informix®

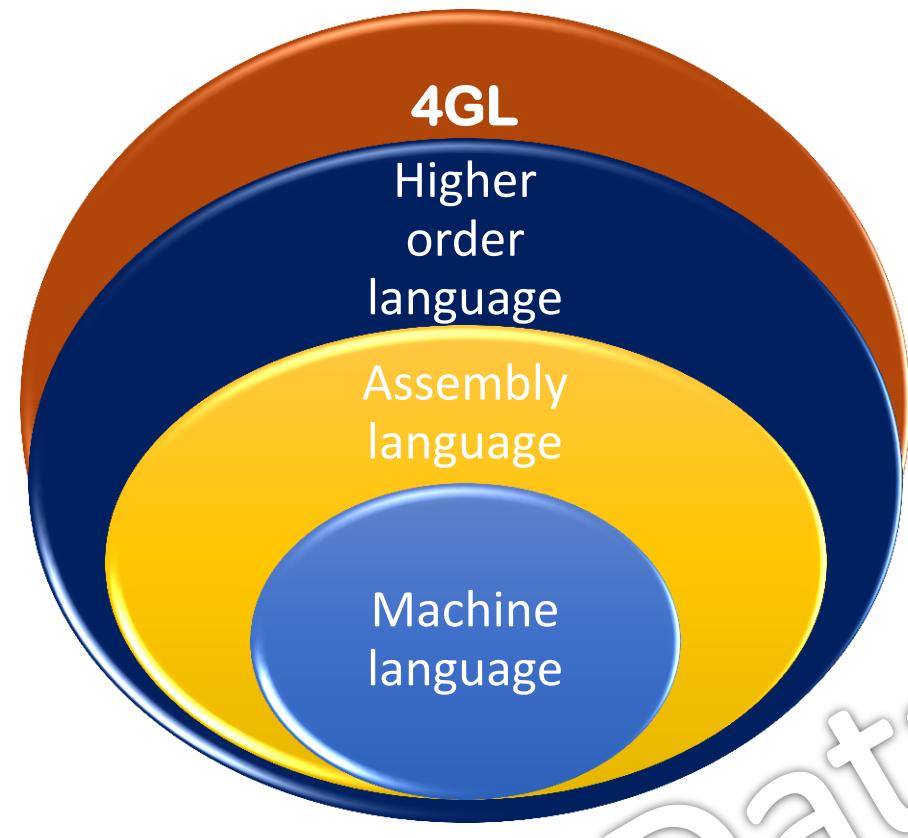


Multi Engine Architecture

TERADATA.

eBase™

Basic Concepts - Base SAS Features to Know



- Fourth Generation Language
 - Data Step
 - Procedures
 - Macro
 - SQL
- Formats – To display data in a style you want , say 20208 vs. April 30, 2015 vs. 04/30/15
- Informs – To read any type of text data
 - Functions – Make data out of “stuff”
 - e.g. Manipulating character values, truncate numeric values, grab month out of given date
- Output Delivery System – Customize your output
 - e.g. pdf, html or output sent directly to printer

Base SAS software consists of more than 70 procedures that provide:

- **Data Access**
- Read file (xls, csv, oracle files etc.)
eg. Proc import, proc sql,
- **Data Management**
- Data manipulation and restructuring.
eg. Proc sort, proc transpose, proc datasets
- **Data Analysis**
- Perform descriptive statistics such as means, frequency, standard deviation, etc.
eg. Proc freq, proc means, proc univariate etc.
- **Data Presentation**
- Produce reports that range from a simple listing of a data set to customized reports of complex relationships and analyses as well as graphics.
eg. Proc print, proc report, proc plot

What's on my sas session ?

Let's run :

```
proc setinit;  
run;
```

Partial log :

```
Operating System: LIN X64 .  
Product expiration dates:  
---Base SAS Software 16JUN2016 (CPU A)  
---SAS/STAT 16JUN2016 (CPU A)  
---SAS/IML 16JUN2016 (CPU A)  
---SAS/Secure 168-bit 16JUN2016 (CPU A)  
---SAS/ACCESS Interface to PC Files 16JUN2016 (CPU A)  
---SAS/ACCESS Interface to ODBC 16JUN2016 (CPU A)  
---SAS/IML Studio 16JUN2016 (CPU A)  
---SAS Workspace Server for Local Access 16JUN2016 (CPU A)  
---SAS Workspace Server for Enterprise Access 16JUN2016 (CPU A)  
---High Performance Suite
```

Programming Basics

DataScienceLab

Outline

- Programming Basics : Structure of SAS Programs
 - SAS steps, statements, step boundaries, comments
- SAS syntax rules
- Type of errors :
 - Syntax Errors
- Handling Syntax Errors

Programming Basics : Structure of SAS Programs

Sequence of SAS statements form a step.

```
data work.test_results; 1  
  set datalib.University_test; 2  
run; 3
```

Sequence of steps build a SAS program.

```
data work.test_results;  
  set datalib.University_test;  
run;
```

```
Proc print data=work.test_results;  
Run;
```

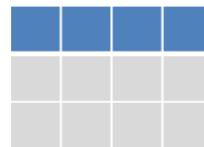
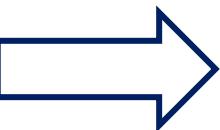
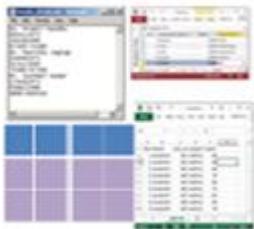
There are two type of steps in SAS :

- ✓ DATA step
- ✓ PROC step

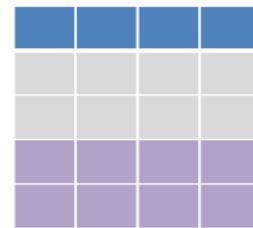
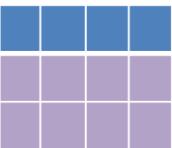
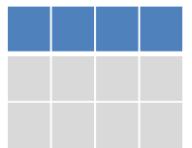
```
data work.test_results;  
  set datalib.University_test;  
run;
```

```
Proc print data=work.test_results;  
run;
```

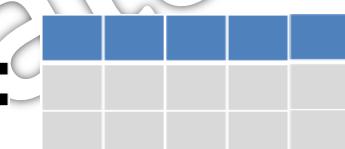
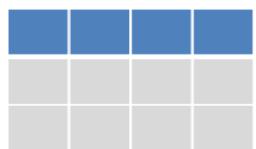
The DATA step is used to:



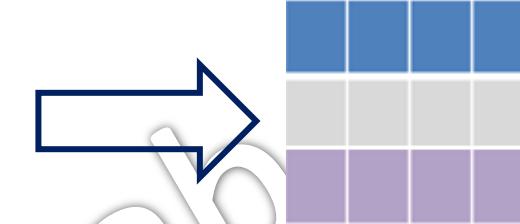
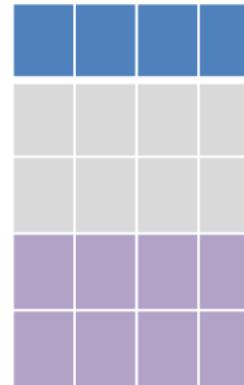
- read data, especially text data



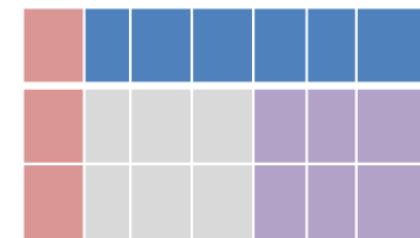
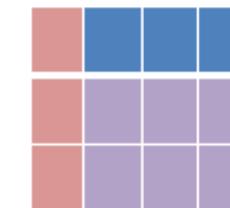
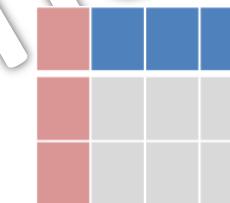
- create new data sets



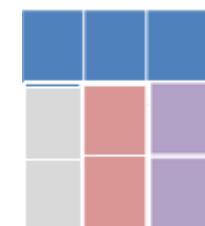
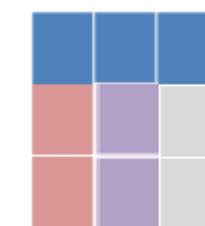
- create new variables



- subset data sets



- concatenate, merge and update data sets



- restructure data

...and a lot!

The PROC step is used to:

Pre-written procedures in Base SAS can be used to perform various tasks such as :

Obs	country_of_export	Date	HS_Code	Qty_	Unit
1	MALAYSIA	22JAN2014	84799020	16	PCS
2	MALAYSIA	22JAN2014	84799020	16	PCS
3	MALAYSIA	22JAN2014	84799020	16	PCS
4	MALAYSIA	22JAN2014	73269099	6	SET
5	MALAYSIA	22JAN2014	73269099	8	PCS

- Detail

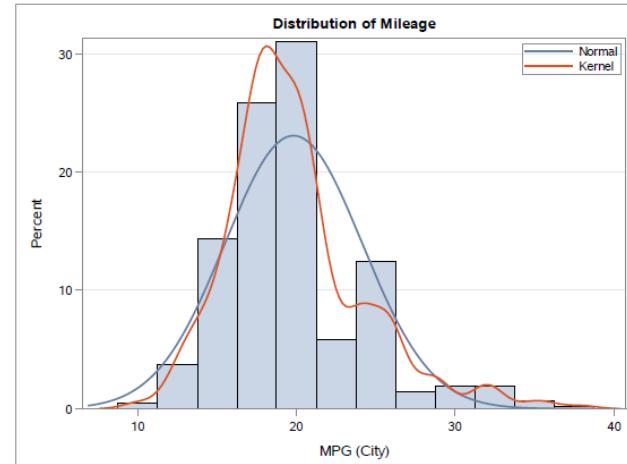
The FREQ Procedure

Country of Export				
country_of_export	Frequency	Percent	Cumulative Frequency	Cumulative Percent
INDONESIA	111	53.11	111	53.11
IRAQ	5	2.39	116	55.50
ITALY	7	3.35	123	58.85
MALAYSIA	39	18.66	162	77.51
NIGERIA	5	2.39	167	79.90
SAUDI ARABIA	14	6.70	181	86.60
SPAIN	2	0.96	183	87.56
UNITED ARAB EMIRATES	12	5.74	195	93.30
UNITED STATES	14	6.70	209	100.00

- Summary

	Qty.
	Sum
Country of Export	
INDONESIA	141418.52
IRAQ	460006.39
ITALY	33260.00
MALAYSIA	223078.20
NIGERIA	341010.79
SAUDI ARABIA	1170681.36
SPAIN	4650.00
UNITED ARAB EMIRATES	563920.76
UNITED STATES	312.00

- Tabular



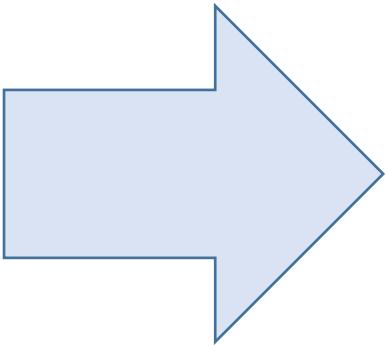
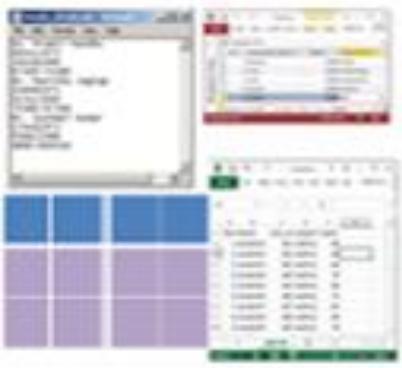
- Graphical

The MEANS Procedure

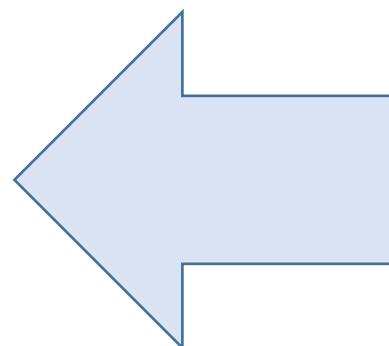
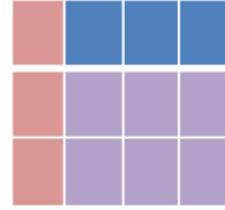
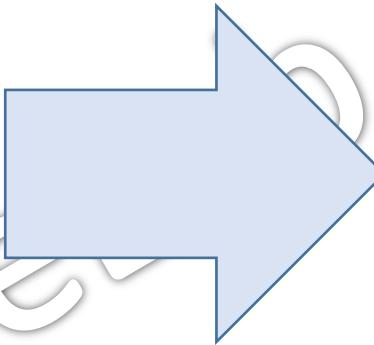
Analysis Variable : Qty_Qty.						
Country of Export	N Obs	N	Mean	Std Dev	Minimum	Maximum
INDONESIA	111	111	1274.04	985.7123599	98.6010000	5000.00
IRAQ	5	5	92001.28	13144.71	68836.00	100047.92
ITALY	7	7	4751.43	5686.94	600.0000000	14400.00
MALAYSIA	39	39	5719.95	14104.75	2.0000000	66009.77
NIGERIA	5	5	68202.16	51561.00	12.0000000	137661.00
SAUDI ARABIA	14	14	83620.10	35239.11	42000.00	146461.00
SPAIN	2	2	2325.00	2227.39	750.0000000	3900.00
UNITED ARAB EMIRATES	12	12	46993.40	24249.45	19265.84	91299.00
UNITED STATES	14	14	22.2857143	32.1616795	1.0000000	120.0000000

- Statistical

Typical Flow of SAS Programs



```
data new;  
set old;  
....  
;  
run;
```



```
...  
Proc xxx  
data=new;  
....  
;  
run;  
...
```

Example of Typical Program

```
data work.test_results;  
length roll_no $ 4 Name $ 12  
subject $6;  
infile 'students.csv' dlm=',';  
input Name $ roll_no $ subject $  
marks;  
run;
```

DATA
STEP

```
proc print data= work.test_results;  
run;
```

PROC
STEP

Step Boundaries

```
data work.test_results;  
length roll_no $ 4 Name $ 12 subject $6;  
infile 'students.csv' dlm='';  
input Name $ roll_no $ subject $ marks;  
run;
```

```
proc sql;  
Create table new select name marks  
from work.test_results;  
quit;
```

RUN statement
acts as a step
boundary

QUIT statement
acts as a step
boundary

- ✓ RUN
 - ✓ QUIT
 - ✓ Start of another step
(DATA or PROC)
- acts as **step boundaries**

Step Boundaries

```
proc print data= work.test_results;
```

Data statement
acts as a step
boundary

 Data percent;
Set new;
Percentage=marks/750*100;

There is **NO**
step boundary
here!!

- ✓ RUN
 - ✓ QUIT
 - ✓ Start of another step
(DATA or PROC)
- acts as **step boundaries**

Various modes for running a SAS Program

Various modes for running SAS program are :

- ✓ interactive mode
 - (e.g.: SAS Studio, SAS windowing environment ,SAS EG)
- ✓ batch mode
- ✓ non interactive mode

Three Primary Windows

You submit and view the results of a SAS program in the SAS windowing environment using three primary windows.

CODE	contains the SAS program to submit.
LOG	contains information about the processing of the SAS program, including any warning and error messages.
OUTPUT	contains reports generated by the SAS program.

DataScienceLab

Code or Editor Window

```
data work.test_results;  
length roll_no $ 4 Name $ 12 subject $ 6;  
infile 'students.csv' dlm='';  
input Name $ roll_no $ subject $ marks;  
run;
```

STEP KEYWORDS

Code or Editor Window

STATEMENT
KEYWORDS

```
data work.test_results;  
length roll_no $ 4 Name $ 12 subject $ 6;  
infile 'students.csv' dlm=',';  
input Name $ roll_no $ subject $ marks;  
run;
```

Code or Editor Window

```
data work.test_results ;  
length roll_no $ 4 Name $ 12 subject $ 6 ;  
infile 'students.csv' dlm=',';  
input Name $ roll_no $ subject $ marks ;  
run;
```

Semicolon ends statement.

Log Window

```
33 data work.test_results;
34 length roll_no $ 4 Name $ 12
35           subject $6 ;
36      infile 'students.csv' dlm=',';
37 input Name $ roll_no $ subject $
38           marks;
39 run;
```

NOTE: The infile 'students.csv' is:
File Name=E:\SAS_Class\students.csv,
RECFM=V,LRECL=256

NOTE: 252 records were read from the infile
'students.csv'.
The minimum record length was 28.
The maximum record length was 47.

NOTE: The data set WORK.test_results has 252
observations and 4 variables.

```
40
41 proc print data=work.test_results;
42 run;
```

NOTE: There were 252 observations read from the data
set WORK.test_results.

Output Window

Obs	Name	roll_no	subject	marks
1	student1	101	maths1	80
2	student2	102	maths1	85
3	student3	103	maths1	84
4	student4	104	maths1	79
5	student5	105	maths1	88
6	student6	106	maths1	73
7	student7	107	maths1	85
8	student8	108	maths1	84
9	student9	109	maths1	79
10	student10	110	maths1	88
11	student11	111	maths1	73
12	student12	112	maths1	92
13	student13	113	maths1	67
14	student14	114	maths1	89
15	student15	115	maths1	73
16	student16	116	maths1	92
17	student17	117	maths1	67
18	student18	118	maths1	89
19	student19	119	maths1	78
20	student20	120	maths1	83
21	student21	121	maths1	82
22	student22	122	maths1	77

SAS Comments

- Comments make program easier to read and edit.
- Comments are used to document notes about the programming flow for a programmer/ reviewer.
- SAS does not execute comments.
- Comments can occur anywhere in the program.
- Two methods of commenting:

```
/* comment */  
* comment ;
```

```
/*This program creates and uses the  
data set called work.test_results.*/
```

1

```
data work.test_results;  
length roll_no $ 4 Name $ 12  
      subject $6;  
infile 'students.csv' dlm=',';  
      input Name $ roll_no $ /*character*/  
      subject $ marks ;  
run;  
/*  
proc print data=work.test_results;  
run;  
*/  
proc means data=work.test_results;  
  *class Name; 4  
  var marks;  
run;
```

2

3

4

SAS Syntax Rules

Structured, consistent spacing makes a SAS program easier to read.

```
data work.test_results;  
    length roll_no $ 4 Name $ 12 subject $ 6;  
    infile 'students.csv' dlm=',';  
    input Name $ roll_no $ subject $ marks;  
run;  
  
proc print data=work.test_results;  
run;  
  
proc means data=work.test_results;  
    class Name;  
    var marks;  
run;
```

**GOOD
Formatting**

SAS Syntax Rules

**BAD
Formatting**

```
data work.test_results;
length roll_no $ 4 Name $ 12 subject $ 6;
infile 'students.csv' dlm=',';input Name $
roll_no $
subject $ marks;run;
proc print data=work.test_results;run;
proc means data=work.test_results;
class Name;
var marks;run;
```

Syntax Errors

- Syntax errors occur when program statements do not follow to the rules of the SAS language.
- When syntax error occurs, SAS prints an error message or a warning to the log.
- SAS stops processing if error message displays to the log.

Examples of syntax errors:

1. Misspelled keywords

```
prog print data = sashelp.air;  
run;
```

Partial log :

WARNING 14-169: Assuming the symbol PROC was misspelled as prog.

REMEMBER!
We are using SAS Studio(SAS
Univ Edition), which auto
corrects certain errors .

Examples of syntax errors:

2. Unmatched quotation marks

```
proc print data=sashelp.cars;  
where Type = 'SUV;  
run;
```

Partial log :

NOTE: No observations were selected from data set SASHELP.CARS.

NOTE: There were 0 observations read from the data set SASHELP.CARS.

WHERE 0 /* an obviously FALSE WHERE clause */ ;

Examples of syntax errors:

3. Missing semicolons

```
proc print data= sashelp.cars  
run;
```

Partial log :

ERROR 22-322: Syntax error, expecting one of the following: ;, (, BLANKLINE, CONTENTS, DATA, DOUBLE, GRANDTOTAL_LABEL, GRANDTOT_LABEL, GRAND_LABEL, GTOTAL_LABEL, GTOT_LABEL, HEADING, LABEL, N, NOOBS, NOSUMLABEL, OBS, ROUND, ROWS, SPLIT, STYLE, SUMLABEL, UNIFORM, WIDTH.

ERROR 202-322: The option or parameter is not recognized and will be ignored.

This syntax error is difficult to recognize.

Examples of syntax errors:

4. Invalid options

```
proc means data=sashelp.cars average;  
run;
```

Partial log :

ERROR 22-322: Syntax error, expecting one of the following: ;, (, ALPHA, CHARTYPE, CLASSDATA, CLM, COMPLETETYPES, CSS, CV, DATA, DESCEND, DESCENDING, DESCENDTYPES, EXCLNPWGT, EXCLNPWGTS, EXCLUSIVE, FW, IDMIN, KURTOSIS, LCLM, MAX, MAXDEC, MEAN, MEDIAN, MIN, MISSING, MODE, N, NDEC, NMISS, NOLABELS, NONOBS, NOPRINT, NOTREADS, NOTRAP, NWAY, ORDER, P1, P10, P20, P25, P30, P40, P5, P50, P60, P70, P75, P80, P90, P95, P99, PCTLDEF, PRINT, PRINTALL, PRINTALLTYPES, PRINTIDS, PRINTIDVARS, PROBT, Q1, Q3, QMARKERS, QMETHOD, QNTLDEF, QRANGE, RANGE, SKEWNESS, STACKODS, STACKODSOUTPUT, STDDEV, STDERR, SUM, SUMSIZE, SUMWGT, T, THREADS, UCLM, USS, VAR, VARDEF. ERROR 202-322: The option or parameter is not recognized and will be ignored.

Try it !

Run Your first program :

1. Open editor and type and submit following program :

Proc setinit;

Run;

2. What did you observe : In output window, results window and log window.

Assignment1

Print first 15 observations

1. Open editor and type and submit following program :

```
proc print data= sashelp.air (obs=15);  
run;
```

2. What did you observe : In output window, results window and log window.

Assignment2

Distribution of Cholesterol for dead and alive males and females

1. Open editor and type and submit following program :

```
proc means data= sashelp.heart maxdec=2;  
class status sex ;  
var Cholesterol;  
run;
```

2. What did you observe : In output window, results window and log window.

SAS Help

Let's learn to take help from the SAS help!

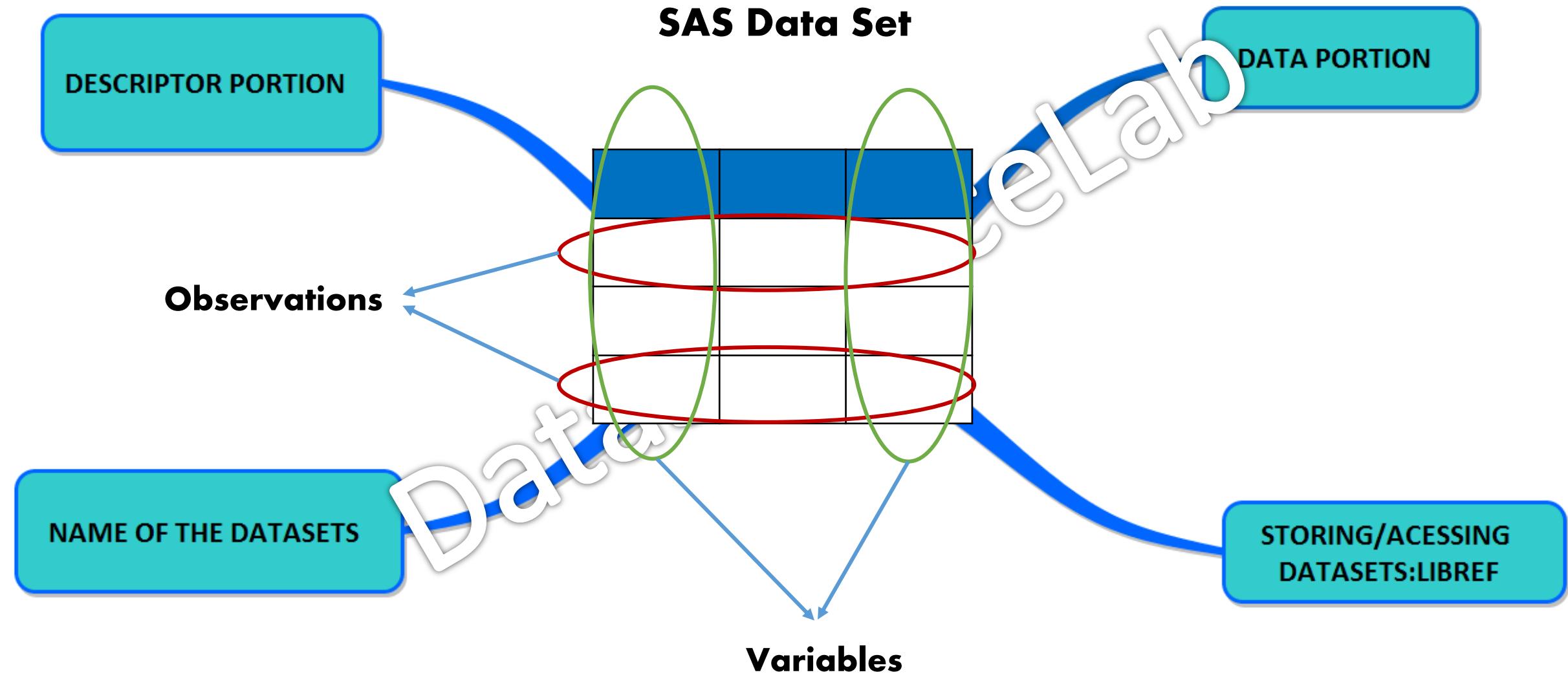
<http://support.sas.com/software/products/sasstudio/index.html?s1=2>

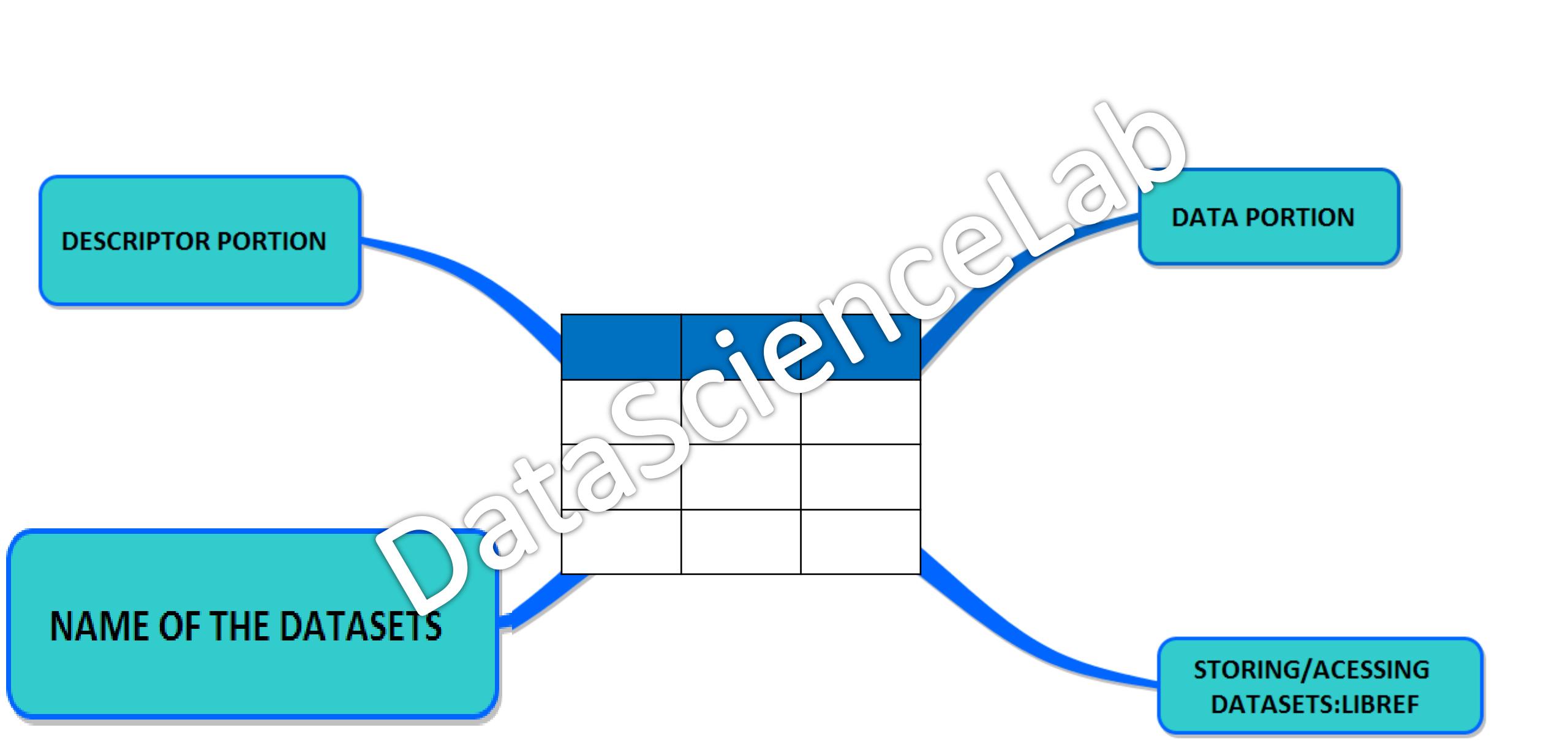
SAS Data Sets

DataScienceLab

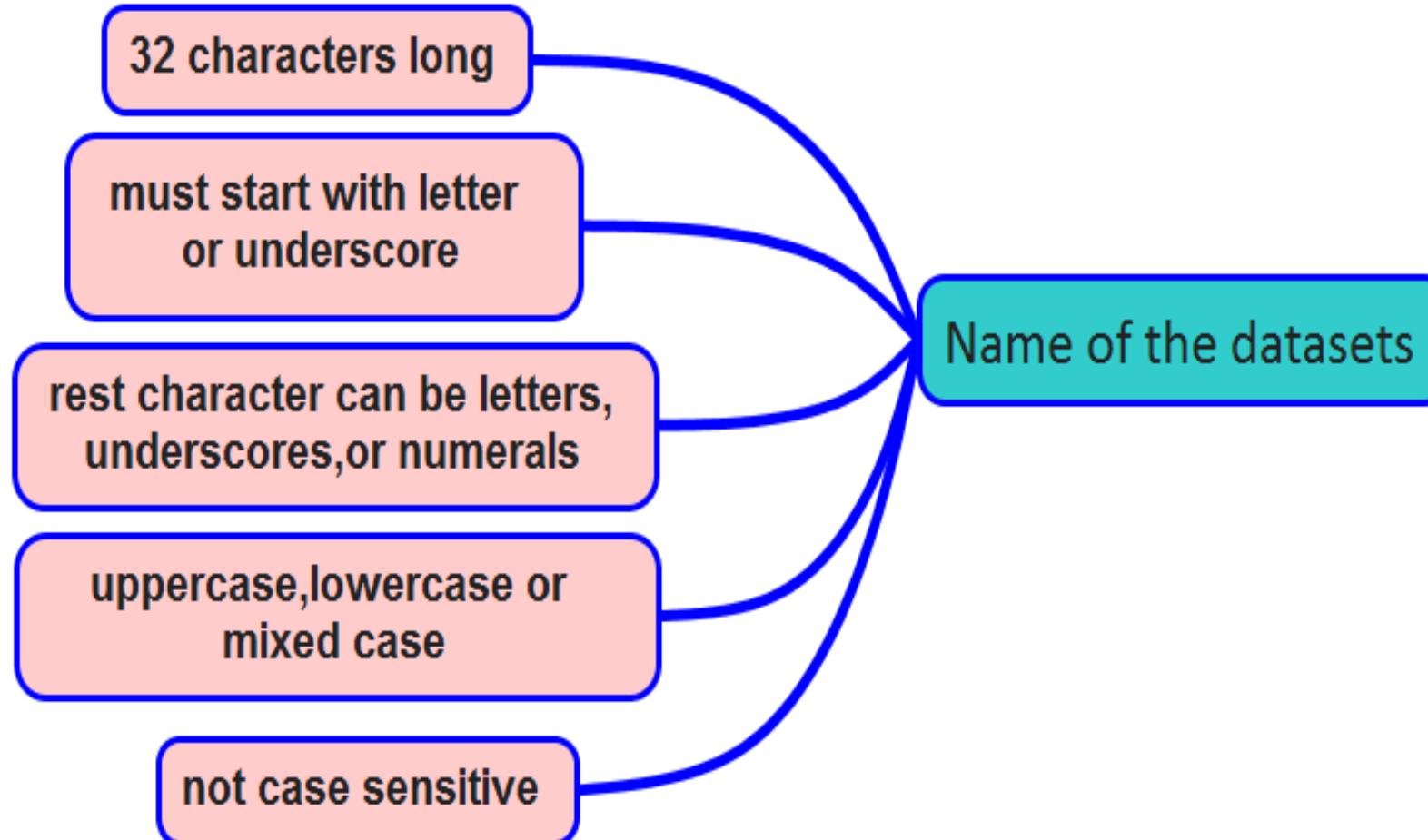
Outline

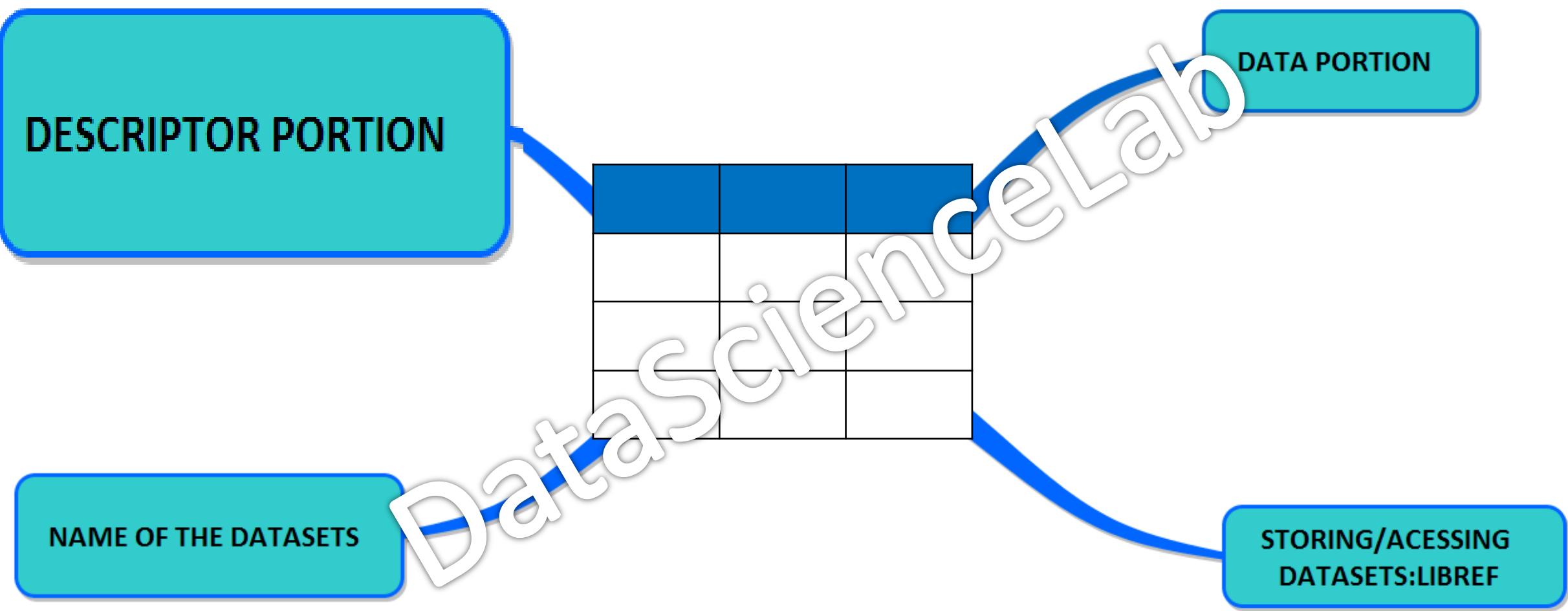
- Rules for SAS dataset
- Parts of SAS data sets
- Storing SAS datasets
- Accessing SAS datasets

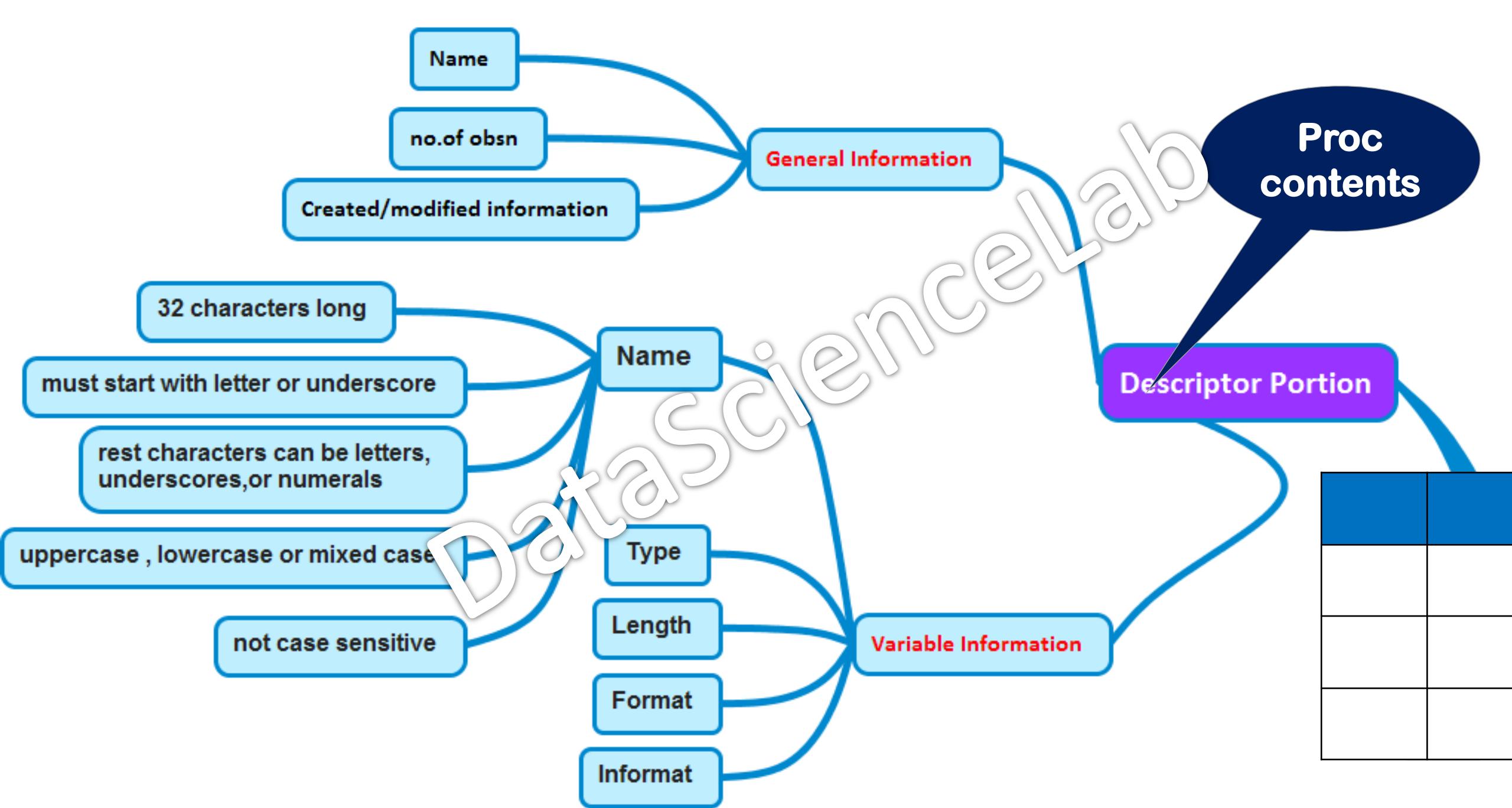




DataScienceLab







SAS Data Set

A SAS data set is a file that SAS creates and processes.

Data Set Name	SASHHELP.CARS	Observations	428
Member Type	DATA	Variables	15
Engine	V9	Indexes	0

Descriptor
Portion

Obs	Make	Model	Type	Origin	DriveTrain	MSRP	Invoice	EngineSize
1	Acura	MDX	SUV	Asia	All	\$36,945	\$33,337	3.5
2	Acura	RSX Type S 2dr	Sedan	Asia	Front	\$23,820	\$21,761	2.0
3	Acura	TSX 4dr	Sedan	Asia	Front	\$26,990	\$24,647	2.4
4	Acura	TL 4dr	Sedan	Asia	Front	\$33,195	\$30,299	3.2
5	Acura	3.5 RL 4dr	Sedan	Asia	Front	\$43,755	\$39,014	3.5

Data
Portion

DataScienceLab

The Descriptor Portion

Use PROC CONTENTS to display the descriptor portion of a data set.

Syntax :

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

Example:

```
proc contents data=sashelp.heart;
run;
```

The Descriptor Portion

Data Set Name	SASHELP.HEART	Observations	5209
Member Type	DATA	Variables	17
Engine	V9	Indexes	0
Created	06/13/2013 06:39:51	Observation Length	168
Last Modified	06/13/2013 06:39:51	Deleted Observations	0
Protection		Compressed	NO
Data Set Type		Sorted	NO
Label	Framingham Heart Study		
Data Representation	SOLARIS_X86_64, LINUX_X86_64, ALPHA_TRU64, LINUX_IA64		
Encoding	us-ascii ASCII (ANSI)		

Alphabetic List of Variables and Attributes				
#	Variable	Type	Len	Label
12	AgeAtDeath	Num	8	Age at Death
5	AgeAtStart	Num	8	Age at Start
3	AgeCHDdiag	Num	8	Age CHD Diagnosed
15	BP_Status	Char	7	Blood Pressure Status
14	Chol_Status	Char	10	Cholesterol Status
13	Cholesterol	Num	8	
2	DeathCause	Char	26	Cause of Death
8	Diastolic	Num	8	
6	Height	Num	8	
10	MRW	Num	8	Metropolitan Relative Weight
4	Sex	Char	6	
11	Smoking	Num	8	
17	Smoking_Status	Char	17	Smoking Status
1	Status	Char	5	
9	Systolic	Num	8	
7	Weight	Num	8	
16	Weight_Status	Char	11	Weight Status

Partial output

DatascienceLab

Assignment1

Find out the option to get the logical listing.

```
proc contents data=sashelp.heart varnum;  
run;
```

Assignment2

Descriptor portion of sashelp.air

DataScienceLab

descriptor portion

DATA PORTION

NAME OF THE DATASETS

STORING/ACESSING
DATASETS:LIBREF

DataScienceLab

DESCRIPTOR PORTION

DATA PORTION

NAME OF THE DATASETS

STORING/ACESSING
DATASETS:LIBREF

Data Science

Proc print

Data Portion

Values

numeric

Display in table form

observation

rows

variables

columns

aA-zZ , numbers,special,blank

stored as length of 1 to 32767 bytes

one byte equal one character

character

missing as blank

case sensitive

stored as floating point numbers in 8 bytes of storage by default

Eight bytes of floating point storage provide space for 16 or 17 significant digits (not restricted to 8 digits)

numbers

missing as periods(.)

date

calculated from 01Jan1960

time

calculated from 00:00:00 midnight

datetime

calculated from midnight 00:00:00 of 01Jan1960

Dates in SAS are numbers

01/01/1960 OR Jan 1,1960

01JAN1960



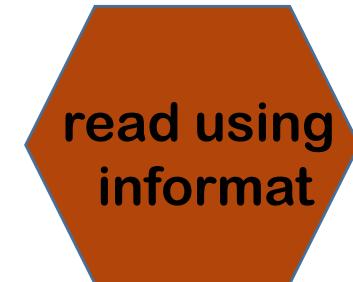
01Jan1959

-365

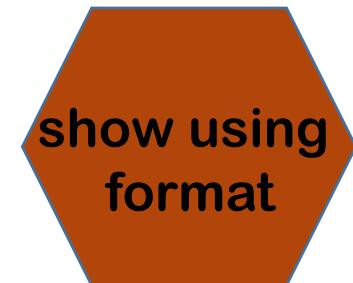
0

+365

31Dec1960



**01JAN1960
or 01/01/60**



Representation of Date, time and date time

```
data test;  
dt='03Jan2016'd;  
tm='11:00:00't;  
dttm='03Jan2016 11:00:00'dt;  
run;
```

```
proc print data=test;  
run;
```

Obs	dt	tm	dttm
1	20456	39600	1767438000

```
proc print data=test;  
format dt date9.  
tm time8.  
dttm datetime17.;  
run;
```

Obs	dt	tm	dttm
1	03JAN2016	11:00:00	03JAN16:11:00:00

The Data Portion

Use the PRINT procedure to see the data portion of a SAS data set.

Syntax :

```
PROC PRINT DATA=SAS-data-set <options>;  
...  
RUN;
```

By default, PROC PRINT displays the following:

- ✓ all observations
- ✓ all variables
- ✓ an obs column on the left side

The Data Portion

Example :

```
proc print data=sashelp.air;  
run;
```

Obs	DATE	AIR
1	JAN49	112
2	FEB49	118
3	MAR49	132
4	APR49	129
5	MAY49	121
6	JUN49	135
7	JUL49	148
8	AUG49	148
9	SEP49	136
10	OCT49	119
11	NOV49	104
12	DEC49	118
13	JAN50	115
14	FEB50	126
15	MAR50	141
16	APR50	135
17	MAY50	125
18	JUN50	149
19	JUL50	170
20	AUG50	170
21	SEP50	158
22	OCT50	133
23	NOV50	114
24	DEC50	140

Partial output

DataScienceLab

Options for browsing the Data Portion

Syntax :

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

Example : PROC PRINT DATA=sashelp.air **(obs=10)** ;
 VAR AIR date ;
 RUN;

- The obs=n option prints only first n observations.
- The VAR statement selects variables and determines their order in the report.

Assignment1

Browse various portion of sashelp.heart

1. See the descriptor portion of sashelp.heart
2. See the data portion of sashelp.heart
3. Select Status, DeathCause , Sex ,AgeAtStart, Height, Weight,Smoking ,AgeAtDeath,Weight_Status variables
4. Print only first 25 obersvations (obs=25)

Assignment2

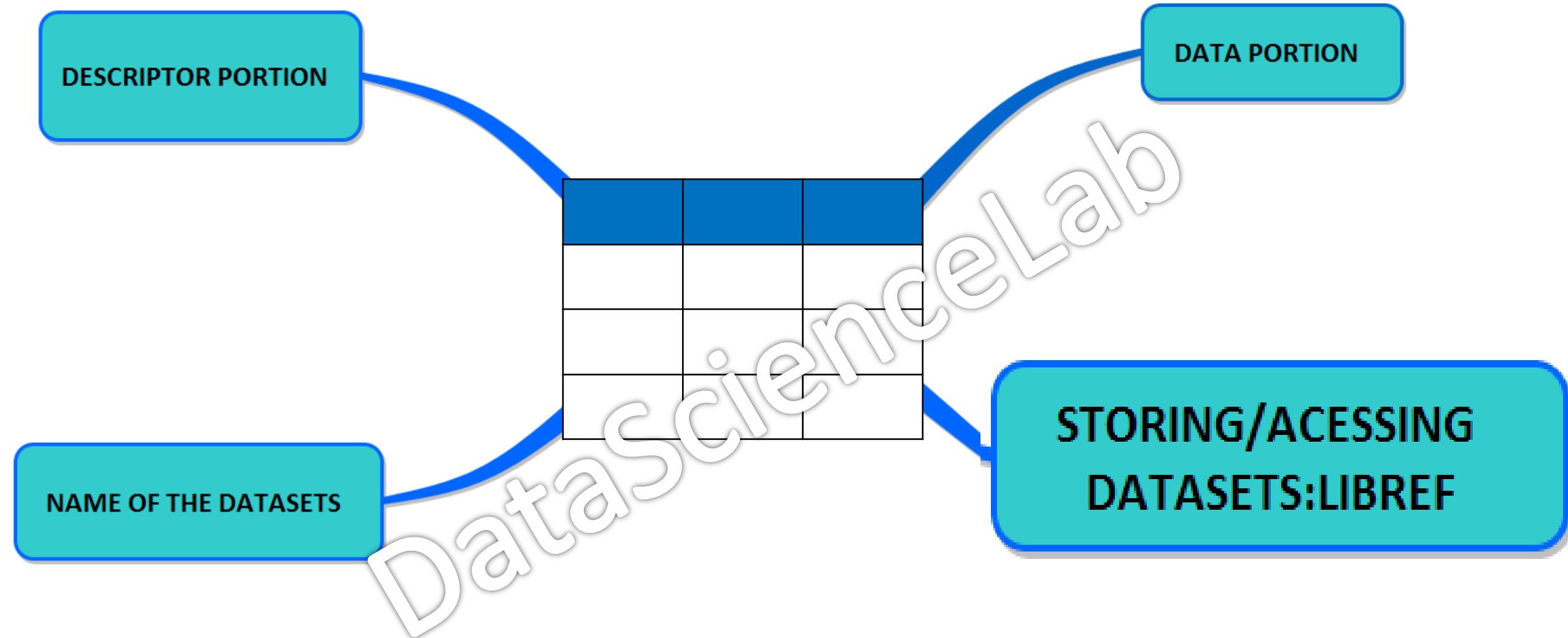
Few options to print sashelp.heart

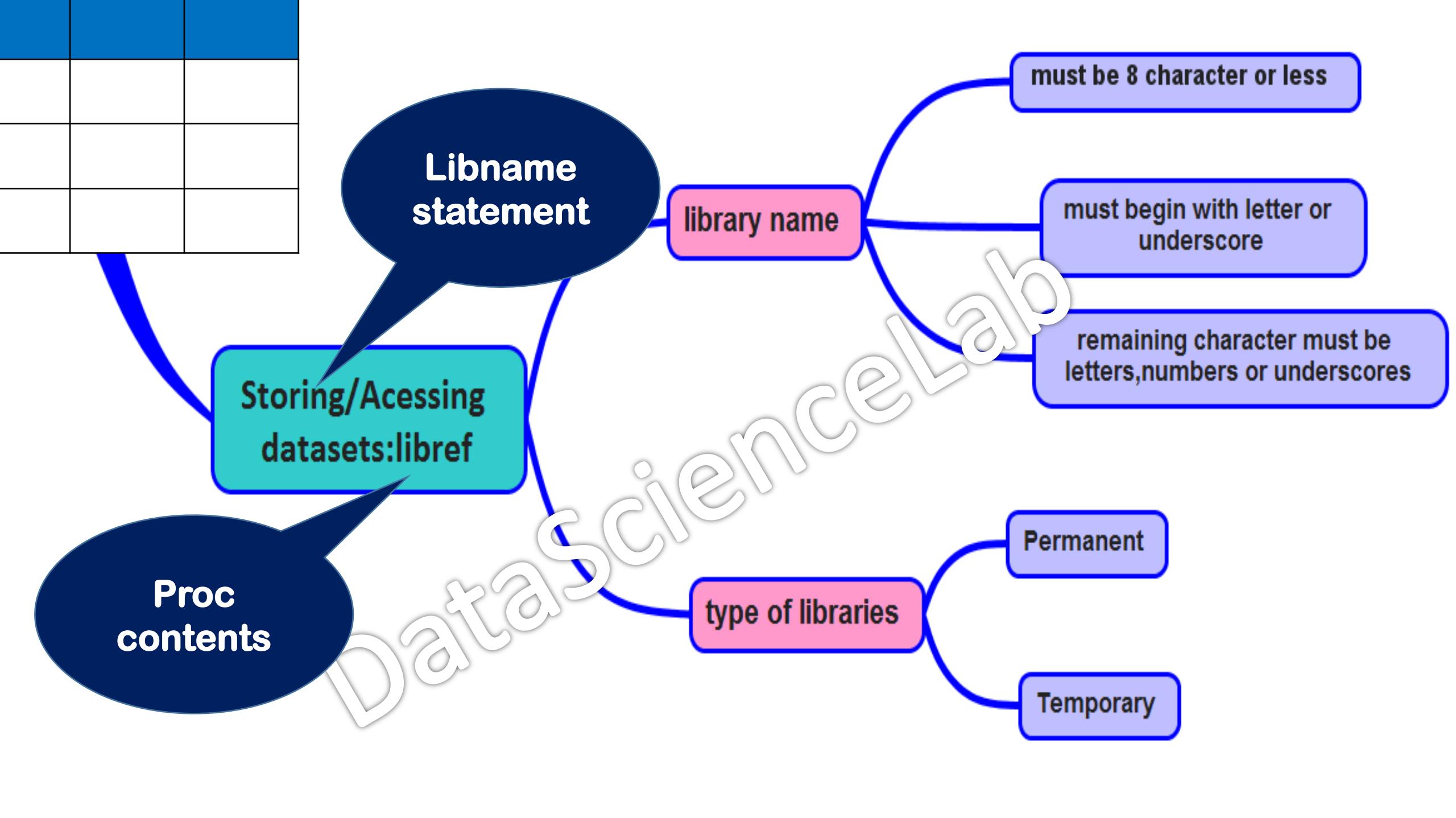
1. Select Status, Sex , Height ,Weight , Cholesterol ,Chol_Status, Weight_Status variables
2. Use noobs option in the proc print statement

Note : noobs option suppresses the observation number column

Accessing SAS Data Libraries

DataScienceLab

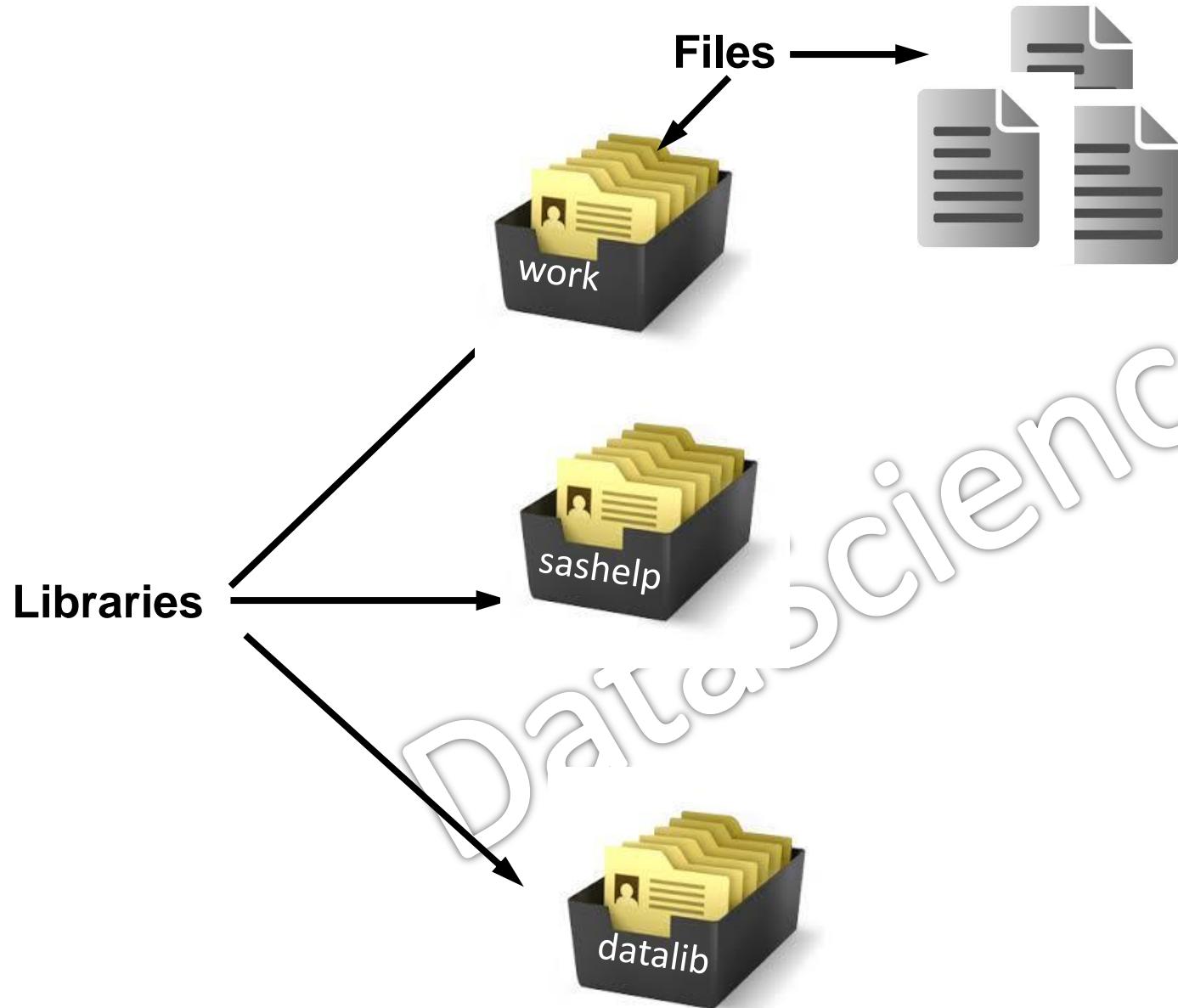




Outline :

- Introduction to SAS library
- Type of libraries
- LIBNAME statement
- Proc contents to see the contents of Library

SAS Libraries



A SAS data library is a collection of SAS files that are recognized as a unit by SAS.

Defining SAS Library

Installation	Pointed to
Windows (BASE/EG etc.)	E:\dataloc
UNIX	/users/ <i>userid</i>
z/OS (OS/390)	<i>userid.dataloc.sasdata</i>
SAS UNIV. EDITION	/folders/MyFolders/dataloc (actual folder : E:\SASUNIVEDITION\myfolders\dataloc)

DataScienceLab

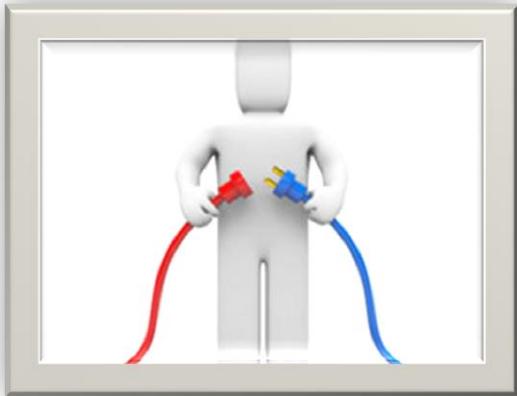
Defining SAS Library

Syntax :

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

Example: For SAS UNIV. EDITION :

libname datalib '/folders/MyFolders/dataloc';



LIBNAME statement makes a connection between a library reference in SAS and the physical location of files on the operating system.

Two type of libraries : Temporary and Permanent

Temporary library : The contents of the library are lost permanently when we end our SAS session.

WORK is a default temporary library.

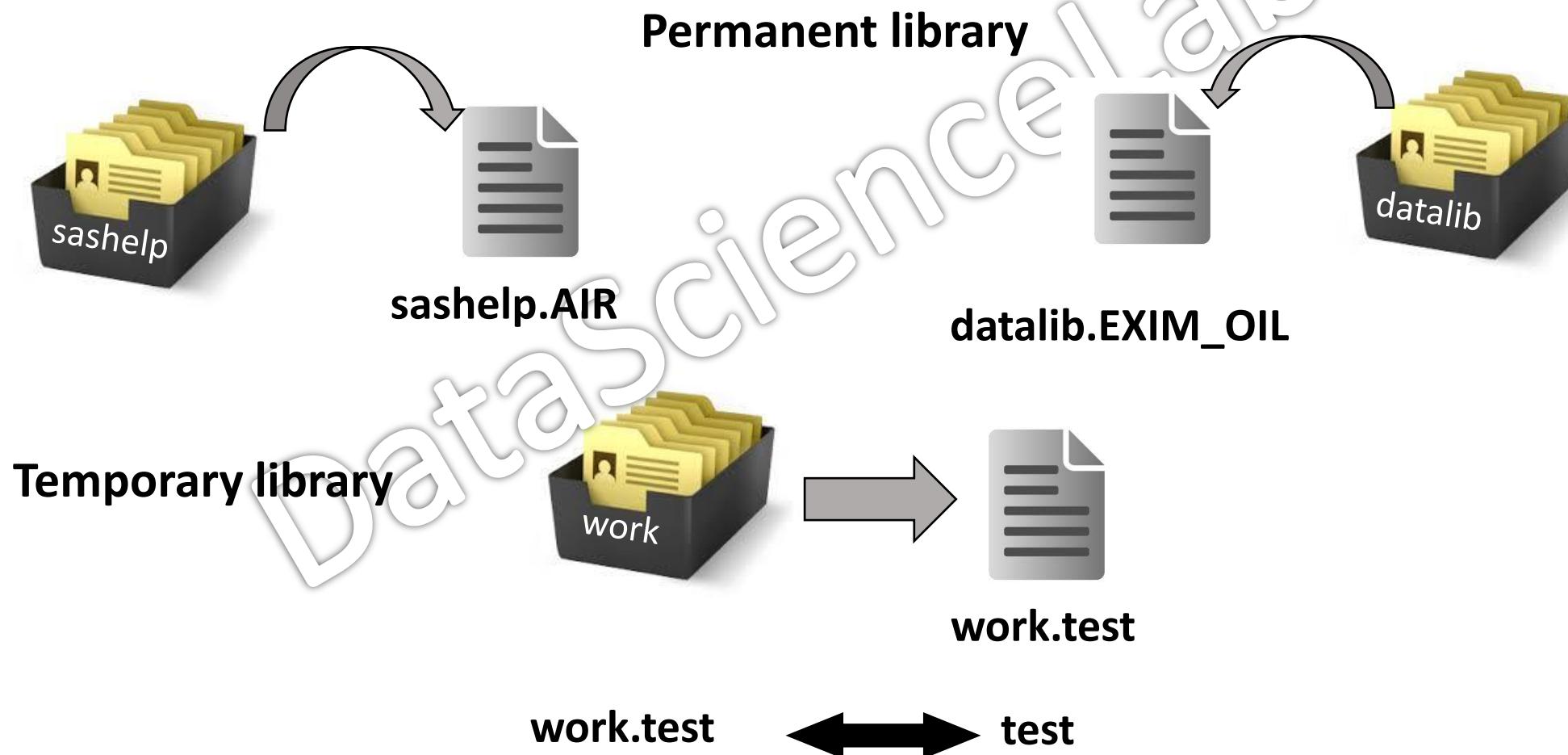
Permanent library : The contents of the library are retained permanently when we end our SAS session. The library reference might be lost from SAS, but the data is still accessible through the actual directory/folder. Also, one can reassign the library reference to get contents of the library in SAS.

SASHELP, SASUSER are default permanent libraries. We can also create and access our own permanent libraries.

Referencing SAS datasets from Temporary and Permanent libraries

Every SAS file has with a two-level name:

libref.filename



Contents of a SAS Library

Use proc contents to see all the SAS files in the librar. To see the list of all the SAS files in the particular library at a time , we use the _ALL_ keyword.

If we use the NODS option, it will hide the descriptor portions of all the data sets and only list of SAS files is shown.

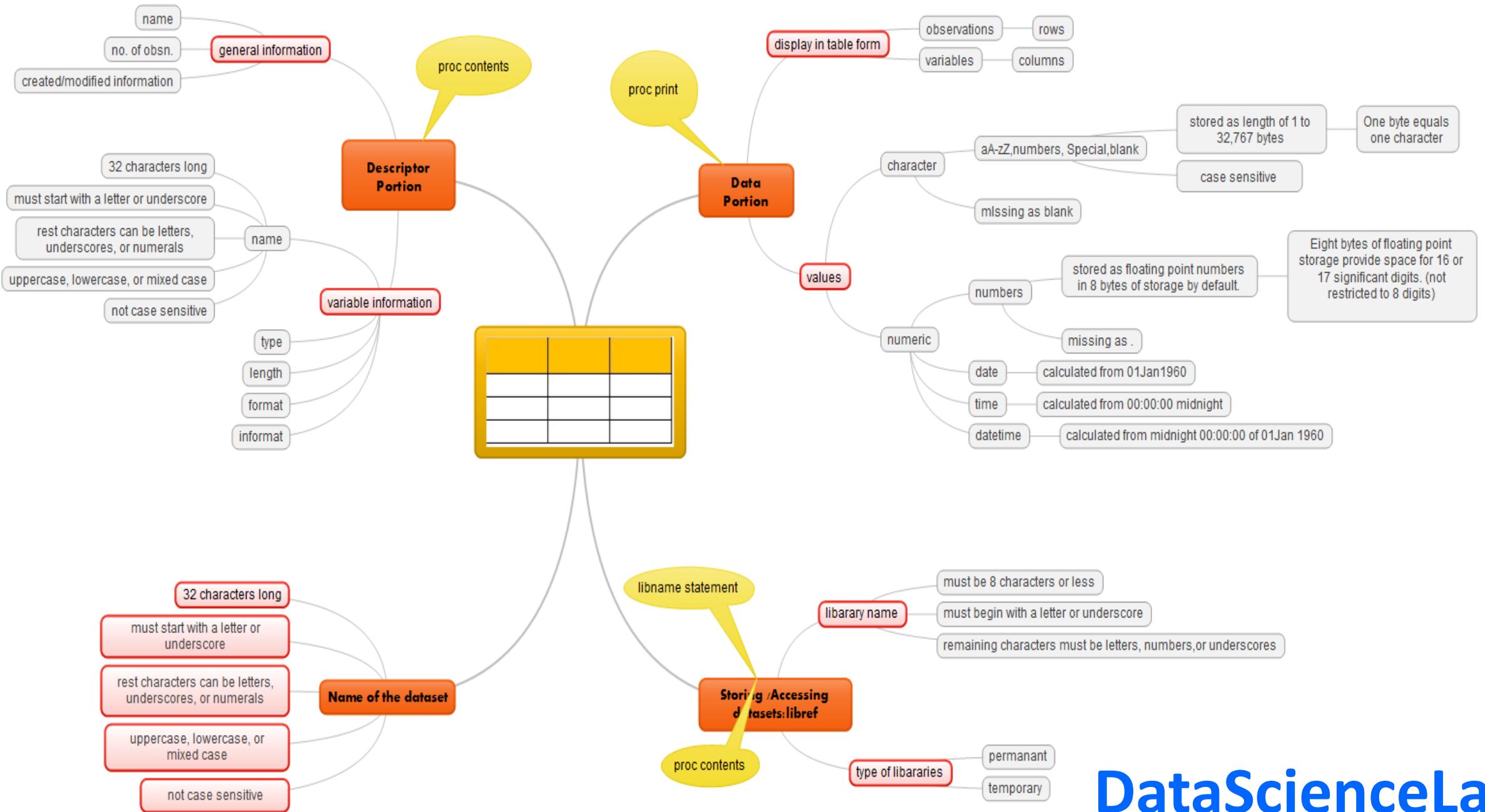
Remember, NODS can be used only along with the _ALL_ keyword.

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

Task

Browsing a SAS Data Library

1. See the contents of datalib library without using NODS option
2. See the contents of datalib library using NODS option
3. What is the difference ?



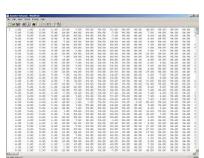
Accessing data in SAS

DataScienceLab

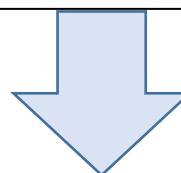
Outline

- Reading in stream data
- Reading from SAS data sets
- Reading from external files

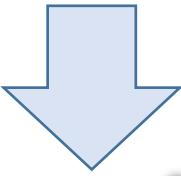
Data sources to create SAS data set



```
data new;  
...;  
...;  
run;
```



Reading in stream data



```
data new;  
datalines;  
Input...;  
....  
run;
```



Reading In stream Data

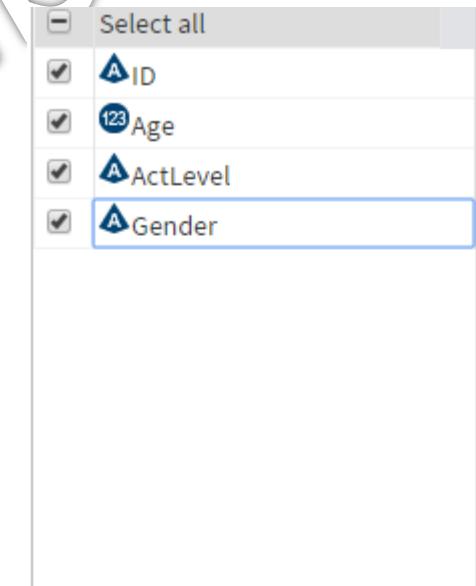
- To read data directly in your SAS program
- To read in stream data:
 - A DATALINES statement is placed as the last statement, immediately preceding data lines.
 - A null statement (single semicolon) to indicate end of input data.
 - Only one DATALINE statement can be used in a DATA step.
 - Do not need a RUN statement after null statement. DATALINE statement works as step boundary.

Reading In stream Data

```
data work.exercise;  
input ID $ Age ActLevel $ Gender $;  
datalines;  
1000 61 MOD F  
1001 38 HIGH M  
1002 42 LOW M  
1003 26 HIGH F  
1004 50 MOD M  
1005 25 HIGH F  
;  
;
```

NOTE: The data set WORK.EXERCISE has 6 observations
and 4 variables.

By default, character variable is also 8 bytes



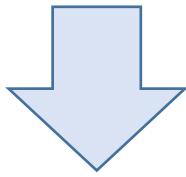
The screenshot shows the Variable Properties panel in SAS Studio. It displays four variables: ID, Age, ActLevel, and Gender. The Gender variable is selected, highlighted with a blue border. The properties table below shows the following details for the Gender variable:

Property	Value
Label	Gender
Name	Gender
Length	8
Type	Char
Format	
Informat	

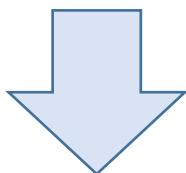
Reading SAS data set to
SAS data set

DataScienceLab

Reading SAS data set to SAS data set



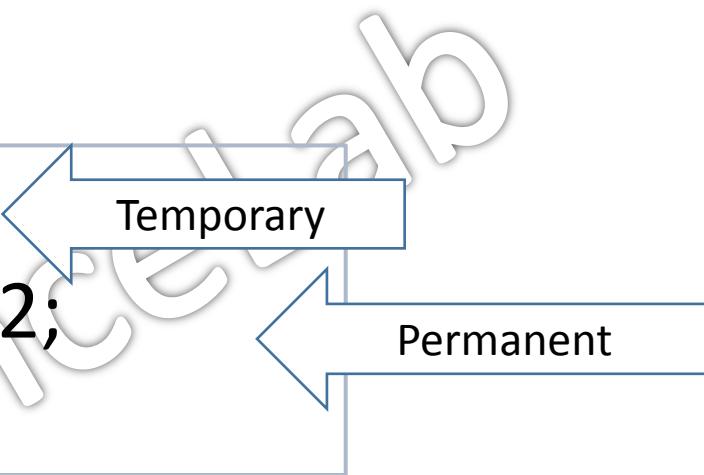
```
data new;  
set old;  
...  
run;
```



Read SAS dataset from SAS dataset

Syntax:

```
data emps2;  
set datalib.emps2;  
run;
```



The diagram consists of two blue arrows pointing to specific words in the SAS code. One arrow points to the word 'Temporary' in the first line, and another points to the word 'Permanent' in the second line.

- ✓ Read from permanent &/or temporary data set into permanent &/or temporary data set
- ✓ It creates exact copy of input dataset unless instructed

Read SAS dataset from SAS dataset

Partial output

EMPID	HIRE_DATE	SERVICEYRS	SALARY
101	13OCT1980	34.5079	146230
102	13OCT1973	41.5086	126960
103	13NOV1973	41.4237	130475
104	13JUN1989	25.8426	533800

Partial log

NOTE: There were **44 observations read** from the data set **datalib.emps2**.

NOTE: The data set **WORK.EMPS2** has **44 observations and 4 variables**.

Read SAS dataset from SAS dataset

There is no restriction on number of datasets to be output

```
data emps2 newemps employees;  
set datalib.emps2;  
run;
```

Partial log

NOTE: There were **44 observations read** from the data set **DATALIB.EMPS2**.

NOTE: The data set **WORK.EMPS2** has **44 observations and 4 variables**.

NOTE: The data set **WORK.NEWEMPS** has **44 observations and 4 variables**.

NOTE: The data set **WORK.EMPLOYEES** has **44 observations and 4 variables**.

Read SAS dataset from SAS dataset

There is no restriction on number of datasets to be read

```
data emps23;  
set datalib.emps2 datalib.emps3;  
run;
```

Partial log

NOTE: There were **44 observations read** from the data set **DATALIB.EMPS2**.

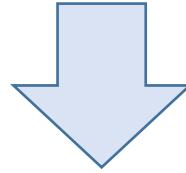
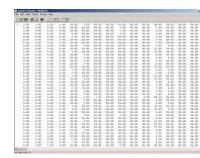
NOTE: There were **44 observations read** from the data set **DATALIB.EMPS3**.

NOTE: The data set **WORK.EMPS23** has **88 observations and 4 variables**.

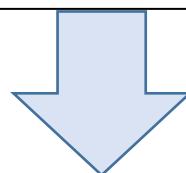
Reading external file
via PROC IMPORT

DataScienceLab

Data sources to create SAS data set



```
proc import  
...;  
...  
run;
```



Proc Import

Syntax :

```
PROC IMPORT
DATAFILE="filename" | TABLE="tablename"
OUT=<libref.>SAS data set <(SAS data set option(s))>
<DBMS=identifier>
<REPLACE>;
RUN;
```

DATAFILE : specifies the complete path and filename or fileref for the input PC file, spreadsheet, or delimited external file.

TABLE : specifies the name of the input DBMS table

OUT : identifies the output SAS data set

DBMS : specifies the type of data to import.

REPLACE : overwrites an existing SAS data set

Proc Import : reading csv file

Read Students.csv file with PROC IMPORT

```
proc import  
datafile="/folders/myshortcuts/MyFolder/GL_SAS/students.csv"  
    out=students  
    dbms=csv  
    replace;  
    getnames=yes;  
run;
```

Generate SAS variable names from the data
values in the first row in the input file

Proc Import

Partial log :

NOTE: The infile '/folders/myshortcuts/MyFolder/GL_SAS/students.csv' is:
Filename=/folders/myshortcuts/MyFolder/GL_SAS/students.csv,
Owner Name=root,Group Name=root,
Access Permission=-rwxrwxrwx,
Last Modified=16Feb2015:19:23:46,
File Size (bytes)=630

NOTE: 22 records were read from the infile
'/folders/myshortcuts/MyFolder/GL_SAS/students.csv'.

The minimum record length was 24.
The maximum record length was 26.

NOTE: The data set WORK.STUDENTS has 22 observations and 5 variables.

Reading csv file having non standard data

Read Students1.csv file with PROC IMPORT

```
proc import  
datafile="/folders/myshortcuts/MyFolder/GL_SAS/students1.csv"  
    out=students  
    dbms=csv  
    replace;  
getnames=yes;  
run;
```

A	B	C	D	E	F	G
Obs	Name	year_of_enrollment	roll_no	subject	marks	DOB
1	student1	2014	101	maths1	80	20/6/1992
2	student2	2014	102	maths1	85	11/4/1992
3	student3	2014	103	maths1	84	3/3/1992
4	student4	2014	104	maths1	79	4/4/1992
5	student5	2014	105	maths1	88	19/2/1992
6	student6	2014	106	maths1	73	13/8/1992
7	student7	2014	107	maths1	85	15/9/1992
8	student8	2014	108	maths1	84	14/7/1992
9	student9	2014	109	maths1	79	2/3/1992
10	student10	2014	110	maths1	88	1/4/1992
11	student11	2014	111	maths1	73	27/2/1992
12	student12	2014	112	maths1	92	4/8/1992
13	student13	2014	113	maths1	67	19/9/1992

Reading delimited file

Read credit_dim.txt : tab delimited file

```
proc import  
datafile="/folders/myshortcuts/MyFolder/GL_SAS/credit_dlm.txt"  
    out=credit  
    dbms=dlm  
    replace;  
delimiter= '09'x;  
getnames=yes;  
run;
```

The screenshot shows the SAS Studio interface during a data import. On the left, the 'Columns' pane displays a list of variables: MALE, _27, _1, _8, _0, and VAR6. The variable '_27' is selected. Below this is a detailed view of the '_27' column properties:

Property	Value
Label	_27
Name	_27
Length	8
Type	Numeric
Format	BEST12.
Informat	BEST32.

On the right, the data preview pane shows 8 rows of data with 6 columns each. The columns are labeled MALE, _27, _1, and _8. The data values are as follows:

	MALE	_27	_1	_8
1	FEMALE	29	3	14
2	FEMALE	40	4	24
3	MALE	.	2	12
4	FEMALE	33	4	16
5	MALE	25	2	9
6	MALE	25	1	0
7	FEMALE	21	1	4
8	MALE	38	3	11

Below the preview, a note states: "NOTE: 8 records were read from the infile '/folders/myshortcuts/MyFolder/GL_SAS/credit_dlm.txt'. The minimum record length was 14. The maximum record length was 19. NOTE: The data set WORK.CREDIT has 8 observations and 6 variables."

On an ASCII platform, the hexadecimal representation of a tab is '09'x.

On an EBCDIC platform, the hexadecimal representation of a tab is a'05'x.

Reading delimited file

Read credit_dim.txt : tab delimited file

```
proc import  
datafile="/folders/myshortcuts/MyFolder/GL_SAS/credit_dlm.txt"  
    out=credit1  
    dbms=dlm  
    replace;  
delimiter= '09'x;  
    getnames=no;  
run;
```

Columns

<input checked="" type="checkbox"/> Select all
<input checked="" type="checkbox"/> VAR1
<input checked="" type="checkbox"/> VAR2
<input checked="" type="checkbox"/> VAR3
<input checked="" type="checkbox"/> VAR4
<input checked="" type="checkbox"/> VAR5
<input checked="" type="checkbox"/> VAR6

Property	Value
Label	VAR3
Name	VAR3
Length	8
Type	Numeric
Format	BEST12.
Informat	BEST32.

Total rows: 9 Total columns: 6

Rows 1-9

	VAR1	VAR2	VAR3	VAR4
1	MALE	27	1	8
2	FEMALE	29	3	14
3	FEMALE	40	4	24
4	MALE	.	2	12
5	FEMALE	33	4	16
6	MALE	25	2	9
7	MALE	25	1	0
8	FEMALE	21	1	4
9	MALE	38	3	11

NOTE: 9 records were read from the infile
'/folders/myshortcuts/MyFolder/GL_SAS/credit_dlm.txt'.
The minimum record length was 14.
The maximum record length was 19.
NOTE: The data set WORK.CREDIT1 has 9 observations and 6 variables.

Reading XLSX

Read students1.xlsx file

```
proc import datafile="/folders/myshortcuts/MyFolder/GL_SAS/students1.xlsx"  
    dbms=XLSX  
    out=students1x  
    replace;  
    datarow = 15;  
run;
```

NOTE: The import data set has 22 observations and 7 variables.

Start reading data from a specific row in the delimited text file

NOTE: The import data set has 9 observations and 7 variables

Proc Import

Advantages :

- Easy to use
- Quick

Disadvantages :

- We can not drop or rearrange some variables.
- PROC IMPORT may not properly assign attributes such as the variables length and format.

Assignments

1. Read Productsales.xlsx file in SAS.
2. Read sashelp.stocks in work library as new dataset stocks_analysis.

Managing and manipulating data

DataScienceLab

Outline

- Add permanent attributes to the data
- Add new variables
- Subset observations and variables

Creating SAS data set

Sub setting Observations
and Variables

LIBNAME *libref* 'SAS-data-library';

DATA *output-SAS-data-set*;

SET *input-SAS-data-set*;

WHERE *where-expression*;

DROP *variable-list*;

LABEL *variable* = 'label'

variable = 'label'

variable = 'label';

FORMAT *variable(s)* *format*;

newVar= expression with oldvars;

RUN;

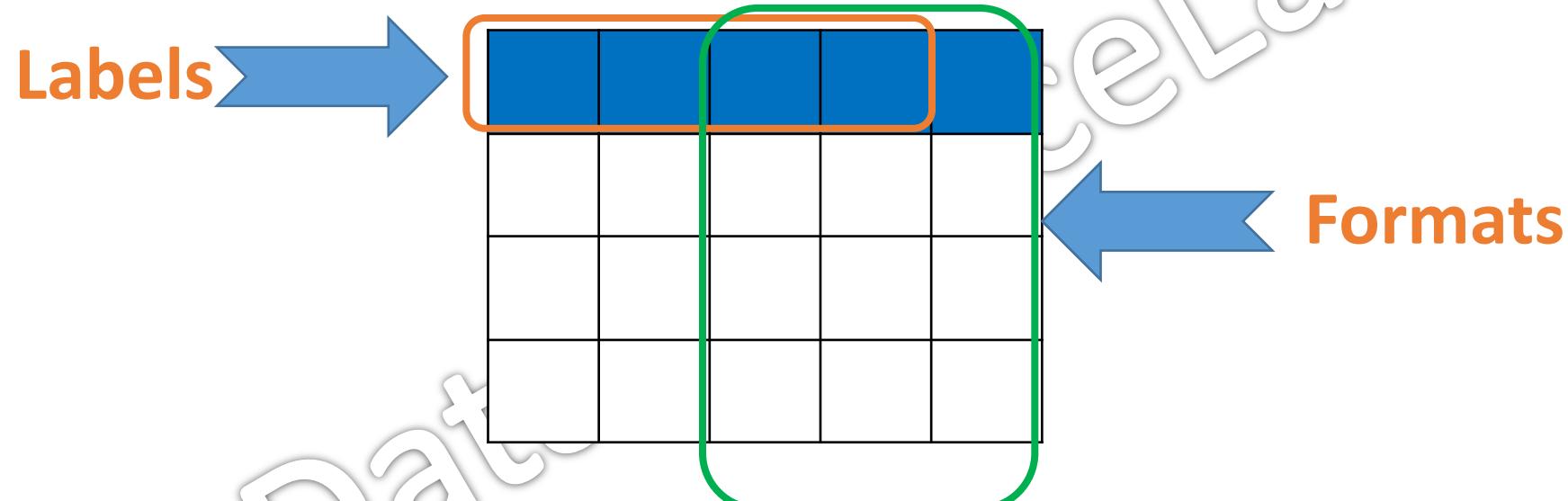
Using SAS Data as Input

Adding Permanent
Attributes

Adding new variables

DataScienceLab

Adding Permanent Attributes



Read SAS dataset from SAS dataset

```
data air;  
set sashelp.air;  
run;
```

DATE	AIR
JAN49	112
FEB49	118
MAR49	132
APR49	129
MAY49	121

Partial data set

Partial log

NOTE: There were 144 observations read from the data set SASHHELP.AIR.

NOTE: The data set WORK.AIR has 144 observations and 2 variables.

Adding Labels

Employee ID	Employee Full Name	Birth Date	Date Of Joining	SALARY

The LABEL Statement

Change the look of variable name by defining label- a description.

Syntax :

```
LABEL variable = 'label'
```

```
    variable = 'label'
```

```
    variable = 'label';
```

- The LABEL statement assigns descriptive labels to variable names.
- A label can be up to 256 characters.
- Any number of variables can be associated with labels in a single LABEL statement.

Permanent labels and temporary labels

Permanent label :

- When assigned in a data step becomes a part of descriptor portion of a data set
- To override: Reassign in data step or assign in proc step

Example:

```
data work.air; /* assign label*/  
set sashelp.air;  
label date='Month Of Flying';  
run;  
  
Proc contents data=air; /* chk descriptor*/  
run;  
  
proc print data=air label; /* use perm label*/  
run;
```

Permanent labels and temporary labels

Temporary label :

- When assigned in a proc step it is used only for that step
- Some Proc steps use labels automatically , while some requires exclusive instructions

Example :

```
/*exclusive instructions*/  
proc print data=air label; /*label option */  
label date='Flying Month' air ='Air miles Travelled'; /*label statement */  
run;  
  
/*automatic use of labels*/  
proc means data=air;  
run;
```

Adding Formats

Employee ID	Employee Full Name	Birth Date	Date Of Joining	SALARY
		09Jul1975	30Apr2004	\$25,000
		4Jul1967	29Jun1999	\$50,000
		07Sep1944	17Aug1973	\$35,000

The FORMAT Statement

Change the look of variable name by defining label- a description.

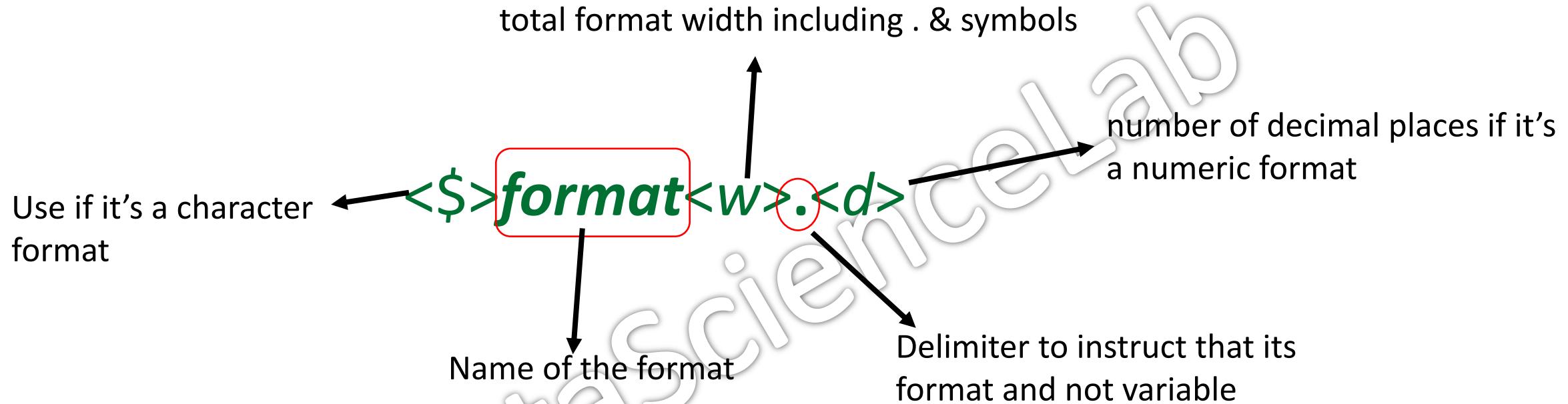
Syntax :

```
FORMAT variable(s) format ;
```

- The *FORMAT statement* assigns formats to variable values.
- A *format* is an instruction that SAS uses to write data values.
- Permanent formats are assigned in data step and temporary formats are defined in the proc steps
- Format statement can assign both, system defined and user defined formats.

Format notation

Notation syntax:



Examples:

SASProgrammingClass → \$4. → SASP

16191 → dollar10.2 → \$16,191.00 OR 5. → 16191 OR eurox9.1 → € 16.191,00

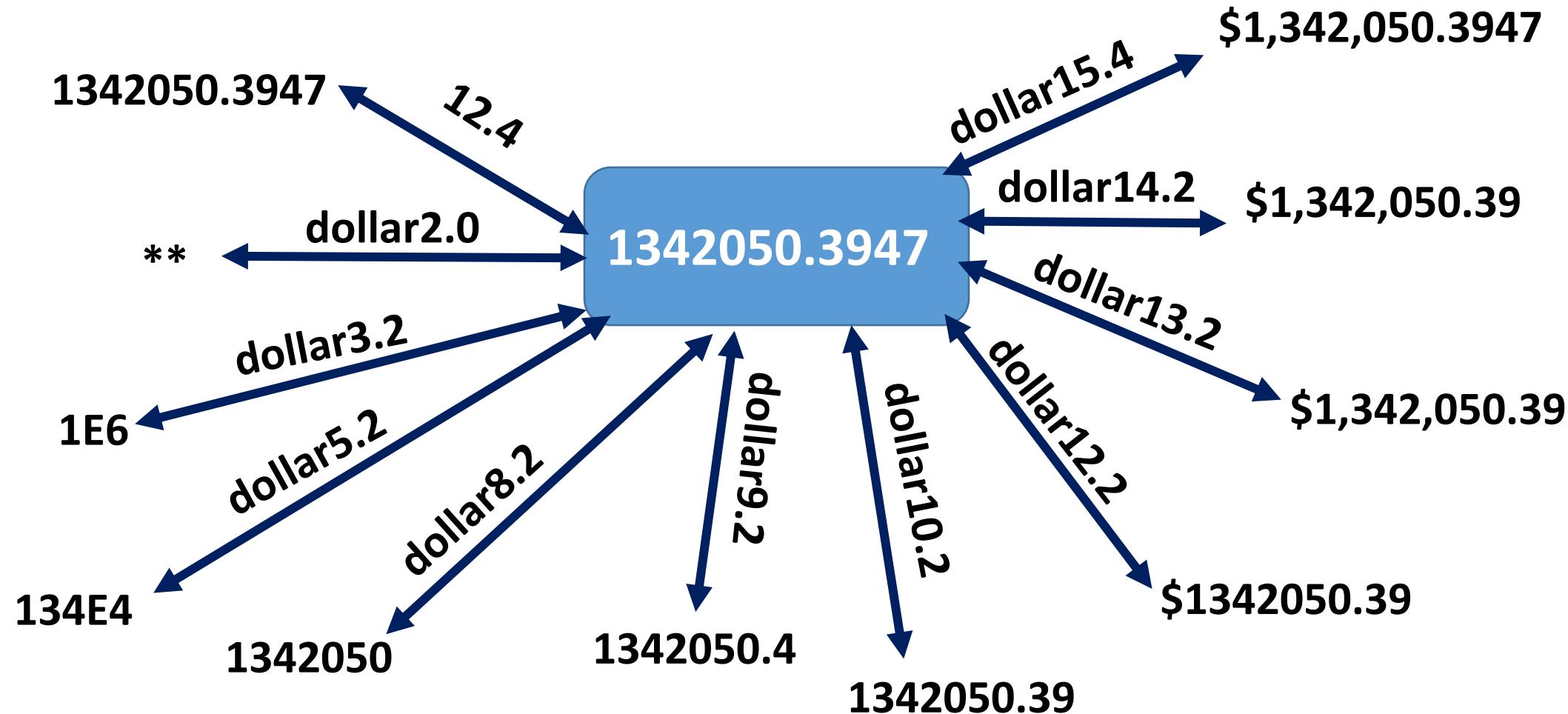
16191 → date9. → 30 APR2004

Some generic forms of SAS formats

Format	Definition
\$w.	Writes standard character data
w.d	Writes standard numeric data
COMMAw.d	Writes numeric values with a comma that separates every three digits and a period that separates the decimal fraction
COMMAXw.d	Writes numeric values with a period that separates every three digits and a comma that separates the decimal fraction
DOLLARw.d	Writes numeric values with a leading dollar sign, a comma that separates every three digits, and a period that separates the decimal fraction
EUROXw.d	Writes numeric values with a leading euro symbol (€), a period that separates every three digits, and a comma that separates the decimal fraction

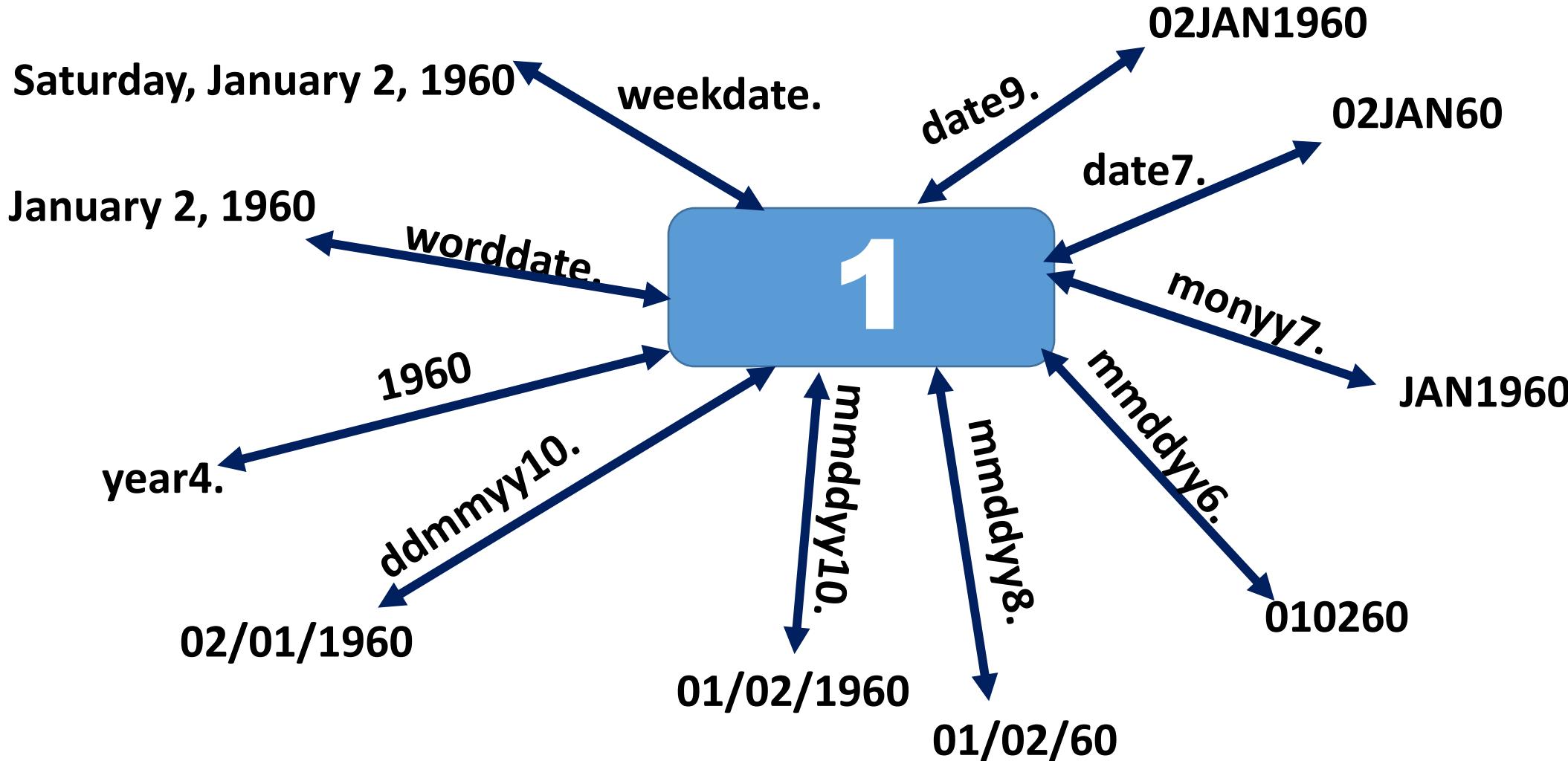
Format width for numbers

- Format width should be appropriate
- If it's not, SAS will try to display approximately correct value, as far as possible



Formats for date

- SAS stores date as number of days calculated from 1st Jan. 1960
- So the value stored is '0' for 1st Jan. 1960



SAS Format

Step1: Make permanent format

```
data work.air;  
set sashelp.air;  
format date year.;  
run;
```

Step2: Check descriptor

```
Proc contents data=air;  
run;
```

#	Variable	Type	Len	Format
2	AIR	Num	8	
1	DATE	Num	8	YEAR.

Override and Print the data

```
proc print data=air;  
format date date9.;  
run;
```

Formats are used automatically.

Step3 :Print the data

```
proc print data=air;  
run;
```

Obs	DATE	AIR
1	1949	112
2	1949	118
3	1949	132
4	1949	129
5	1949	121
6	1949	135
7	1949	148
8	1949	148
9	1949	136
10	1949	119

Obs	DATE	AIR
1	01JAN1949	112
2	01FEB1949	118
3	01MAR1949	132
4	01APR1949	129
5	01MAY1949	121
6	01JUN1949	135
7	01JUL1949	148
8	01AUG1949	148

Assignment1

- Print data set sashelp.failure and notice the in built format for the variable ‘DAY’.
- Use weekdate. Format while printing the sashelp.failure data set for the ‘DAY’ variable.
- What is the difference you find in the output ?

Assignment2

- Print first 10 observations from the data set sashelp.failure
- Notice if there is any label.
- Apply the label option and print again (only first 10 observations) and notice if any change
- Now apply label “Failure due to” for the variable CAUSE and notice that it is applied.

Creating new variables

DataScienceLab

Task : Create new variables in the data set

HR department need to declare bonus for all employees . Bonus would be given on their work anniversary:

Create a new variable for following tasks :

- Bonus Do arithmetic operation
- Full_name By concatenating FNAME and LNAME
- Anniversary month of an employee Use function
- Flag for Employees with job title “manager” Use function

Update variables by rounding off their values:

age and serviceyrs Use function

Creating a new variable in the data set

```
data emps;  
set datalib.emps;  
bonus = salary * 0.1;  
run;
```

Alien variable → 'newly created variable'

Task : Check the log and the output dataset . What type of the bonus variable is created? Why is so?

Bonus will be created as numeric in this case because it was result of multiplication i.e. arithmetic calculation.

The new variable is defined as numeric is – if it is assign directly to a numeric variable or value, or is a result of arithmetic calculation or function that results in number.

On the same way, it would have been character -if it is assign directly to a character variable or a string , or is a result of character operator or function that results in string/character.

Creating new variables in the data set

```
data emps;  
set datalib.emps;  
bonus = salary * 0.1;  
Full_name=FNAME||" "||LNAME;  
annivmonth=month(HIRE_DATE);  
if find(Upcase(job_title),"MANAGER")>0 then Flag_M=1;  
else Flag_M=0 ;  
Age=round(age);  
Serviceyrs=round(serviceyrs);  
run;
```

Variable	Type	Len
Full_name	Char	52
annivmonth	Num	8
Flag_M	Num	8

Full_name	annivmonth	Flag_M
First101 LAST101	10	1
First102 LAST102	10	0
First103 LAST103	11	0
First104 LAST104	6	0
First105 LAST105	8	0
First106 LAST106	5	0
First107 LAST107	6	0
First108 LAST108	7	0
First109 LAST109	9	0
First110 LAST110	10	0
First111 LAST111	10	0
First112 LAST112	12	0
First113 LAST113	10	1
First114 LAST114	6	1
First115 LAST115	11	1
First116 LAST116	1	1
First117 LAST117	2	1
First118 LAST118	7	1
First119 LAST119	8	1
First120 LAST120	1	1
First121 LAST121	7	0

Creating new variables in the data set

```
data emps;  
set datalib.emps;  
bonus = salary * 0.1;  
Full_name=FNAME||" "||LNAME;  
annivmonth=month(HIRE_DATE);  
if scan(Upcase(job_title),-1)= "MANAGER" then Flag_M=1;  
else Flag_M=0 ;  
Age=round(age);  
Serviceyrs=round(serviceyrs);  
run;
```

Variable	Type	Len
Full_name	Char	52
annivmonth	Num	8
Flag_M	Num	8

Full_name	annivmonth	Flag_M
First101 LAST101	10	1
First102 LAST102	10	0
First103 LAST103	11	0
First104 LAST104	6	0
First105 LAST105	8	0
First106 LAST106	5	0
First107 LAST107	6	0
First108 LAST108	7	0
First109 LAST109	9	0
First110 LAST110	10	0
First111 LAST111	10	0
First112 LAST112	12	0
First113 LAST113	10	1
First114 LAST114	6	1
First115 LAST115	11	1
First116 LAST116	1	1
First117 LAST117	2	1
First118 LAST118	7	1
First119 LAST119	8	1
First120 LAST120	1	1
First121 LAST121	7	0

Accumulation using The SUM statement

```
data Jan2014;  
set datalib.fish_oil_daily;  
Totalvalue+value;  
run;  
proc print data=Jan2014;  
run;
```

The SUM statement :

- ✓ It initialized the variable, Totalvalue to zero
- ✓ The value of Totalvalue is automatically retained
- ✓ Totalvalue is increased by the value of Value for each observation
- ✓ The sum statement ignores missing values of Value

Assignments

Read sashelp.stocks in work library as new dataset stocks_analysis.
(already created in last section)

- The data set should have following new variables price_low, price_high, price_adjclose using relevant rates and volume.
- Drop open and close rates before the calculations
- Drop low, high and adjclose rates from the output
- Subset the data if the volume is greater than 10,000,000
- The output should contain only if the price_adjclose to be greater than 280,000,000

Data Exploration ,
Validation and Analysis

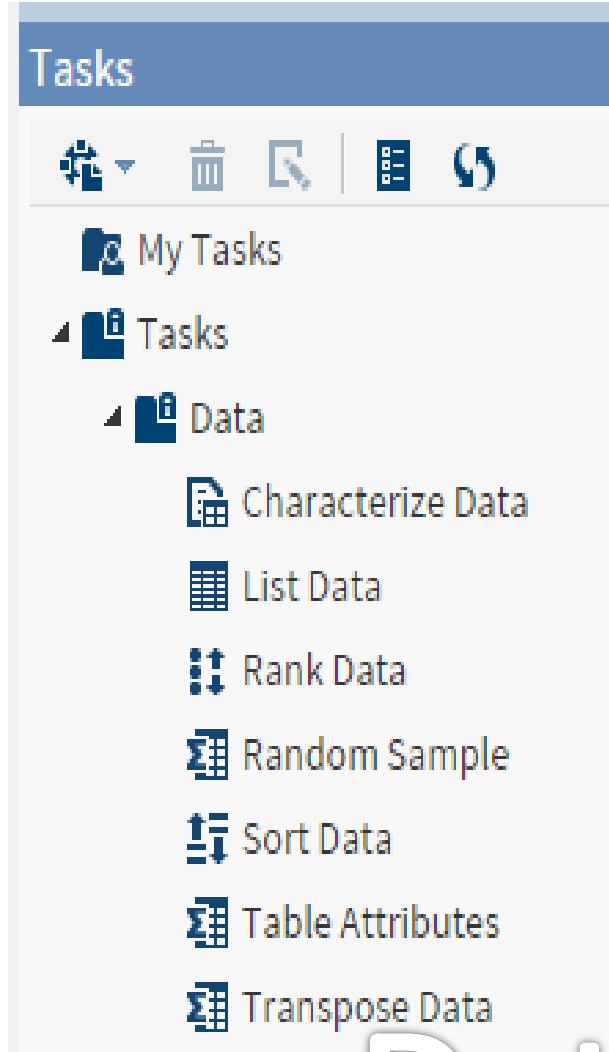
DatascientistLab

Point and click tasks
through
SAS Studio

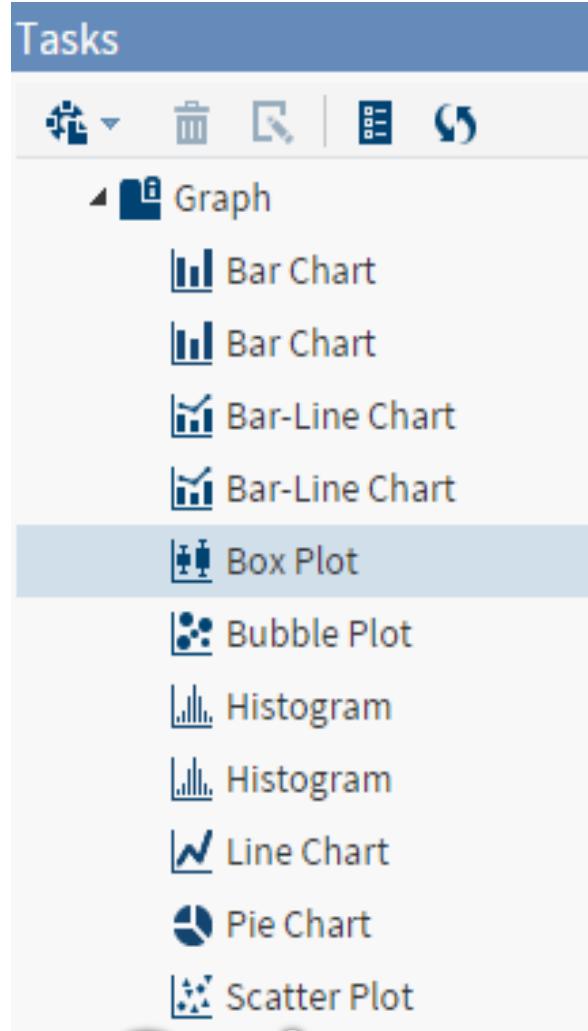
DataScienceLab

List of Tasks

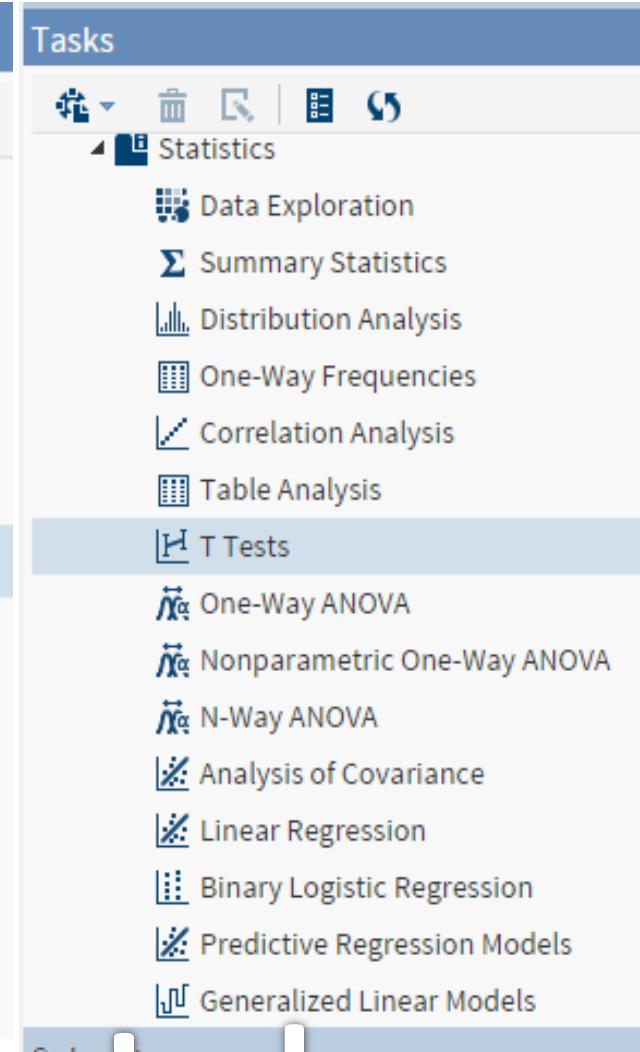
DATA



GRAPH



STATISTICS



DataScienceLab

DataScienceLab

DataTasks

TRANSPOSE DATA : Task

The screenshot shows the SAS Studio interface with the following details:

- SAS Studio Header:** Includes "SAS® Studio", "Program 1 x", "Transpose Data x", "SAS Programmer", and "Sign Out".
- Folders:** "My Tasks", "Tasks", "Data", "Transpose Data" (selected), "Econometrics", "Snippets", "Libraries", and "File Shortcuts".
- Task Configuration:** "Transpose Data" tab selected. "DATA" section shows "SASHelp.CARS". "ROLES" section shows "Transpose variables: Column". "Copy variables:" and "Group analysis by:" sections are empty. "New column names: (1 item)" is listed.
- Code View:** Shows the generated SAS code:

```
1 /*  
2 *  
3 * Code cannot be generated because the fo.  
4 * roles are not set:  
5 *  
6 * 1 Transpose variables  
7 *  
8 *  
9 */  
10  
11
```
- Log View:** Shows the log output: "Line 1, Column 1" and "User: sasdemo".

DataScienceLab

TRANSPOSE DATA : Options

The screenshot shows the SAS Studio interface with the 'Transpose Data' task selected in the left sidebar. The main area displays the 'OPTIONS' tab of the task configuration, which includes fields for 'Name' (Source) and 'Label' (Label), and a checked 'Use prefix' option with 'Column' entered. A note states: 'Note: The prefix check box is checked and disabled by default when no variable is assigned to the "New column names" role.' To the right, the 'CODE' tab shows the generated SAS code:

```
1 /*  
2 *  
3 * Code cannot be generated because the fo.  
4 * roles are not set:  
5 *  
6 * 1 Transpose variables  
7 *  
8 *  
9 */  
10  
11
```

The status bar at the bottom right indicates 'Line 1, Column 1' and 'User: sasdemo'.

DataScienceLab

TRANSPOSE DATA : Task

The screenshot shows the SAS Studio interface with two main panes: DATA and CODE.

DATA Pane:

- Tab bar: Settings, Code/Results (selected), Split.
- Toolbar icons: User, Help, Refresh, Stop.
- Code tab bar: DATA, OPTIONS, INFORMATION.
- DATA section: dropdown set to SASHHELP.CARS.
- ROLES section: A red oval highlights the "ROLES" heading. Underneath, "Transpose variables:" list includes Type, Origin, and MPG_City.
- Copy variables: list includes Column.
- Group analysis by: list includes Origin.
- New column names: (1 item).

CODE Pane:

- Tab bar: CODE, LOG, RESULTS.
- Toolbar icons: Run, Stop, Log, Results, Line #, Edit.
- Code area:

```
18  
19proc sort data=SASHHELP.CARS out=WORK.SORT  
20   by Origin;  
21run;  
22  
23proc transpose data=WORK.SORTTempTableSor  
24   out=WORK.Transpose0002(label='Tra  
25   name=Source label=Label;  
26   by Origin;  
27   var Type Origin MPG_City MPG_Highway;  
28run;  
29  
30quit;  
31title;  
32footnote;  
33%web_drop_table(WORK.SORTTempTableSorted)  
34%web_open_table(WORK.Transpose0002);
```
- Message bar at the bottom: Line 30, Column 6.

TRANSPOSE DATA : Results

Total rows: 12 Total columns: 161				◀◀	◀	Rows 1-12	▶	▶▶
	Origin	Source	Label	Column1	Column2	Column3	Column4	Column5
1	Asia	Type		SUV	Sedan			
2	Asia	Origin		Asia	Asia			
3	Asia	MPG_City	MPG (City)	17	24			
4	Asia	MPG_Highway	MPG (Highway)	23	31			
5	Europe	Type		Sedan	Sedan			
6	Europe	Origin		Europe	Europe			
7	Europe	MPG_City	MPG (City)	22	23			
8	Europe	MPG_Highway	MPG (Highway)	31	30			
9	USA	Type		SUV	SUV			
10	USA	Origin		USA	USA			
11	USA	MPG_City	MPG (City)	15	19			
12	USA	MPG_Highway	MPG (Highway)	21	26			

DataScienceLab

Assignment1

Create a new dataset work.sort by using
datalib.emps and sort the data by gender
within department

Assignment2

- 1) Randomly select 1000 sample from
sashelp.heart by using seed value as 26542 .
- 2) Do stratified sampling of origin variable from
sashelp.cars.

Assignment3

Restructure the data sashelp.class

DataScienceLab

Data Exploration

Data Set Name	SASHHELP.CARS	Observations	428
Member Type	DATA	Variables	15
Engine	V9	Indexes	0
Created	12/06/2013 20:09:53	Observation Length	152
Last Modified	12/06/2013 20:09:53	Deleted Observations	0
Protection		Compressed	NO
Data Set Type		Sorted	YES
Label	2004 Car Data		
Data Representation	SOLARIS_X86_64, LINUX_X86_64, ALPHA_TRU64, LINUX_IA64		
Encoding	us-ascii ASCII (ANSI)		

Obs	Make	Model	Type	Origin	DriveTrain	MSRP	Invoice	EngineSize
1	Acura	MDX	SUV	Asia	All	\$36,945	\$33,337	3.5
2	Acura	RSX Type S 2dr	Sedan	Asia	Front	\$23,820	\$21,761	2.0
3	Acura	TSX 4dr	Sedan	Asia	Front	\$26,990	\$24,647	2.4
4	Acura	TL 4dr	Sedan	Asia	Front	\$33,195	\$30,299	3.2
5	Acura	3.5 RL 4dr	Sedan	Asia	Front	\$43,755	\$39,014	3.5

DataScienceLab

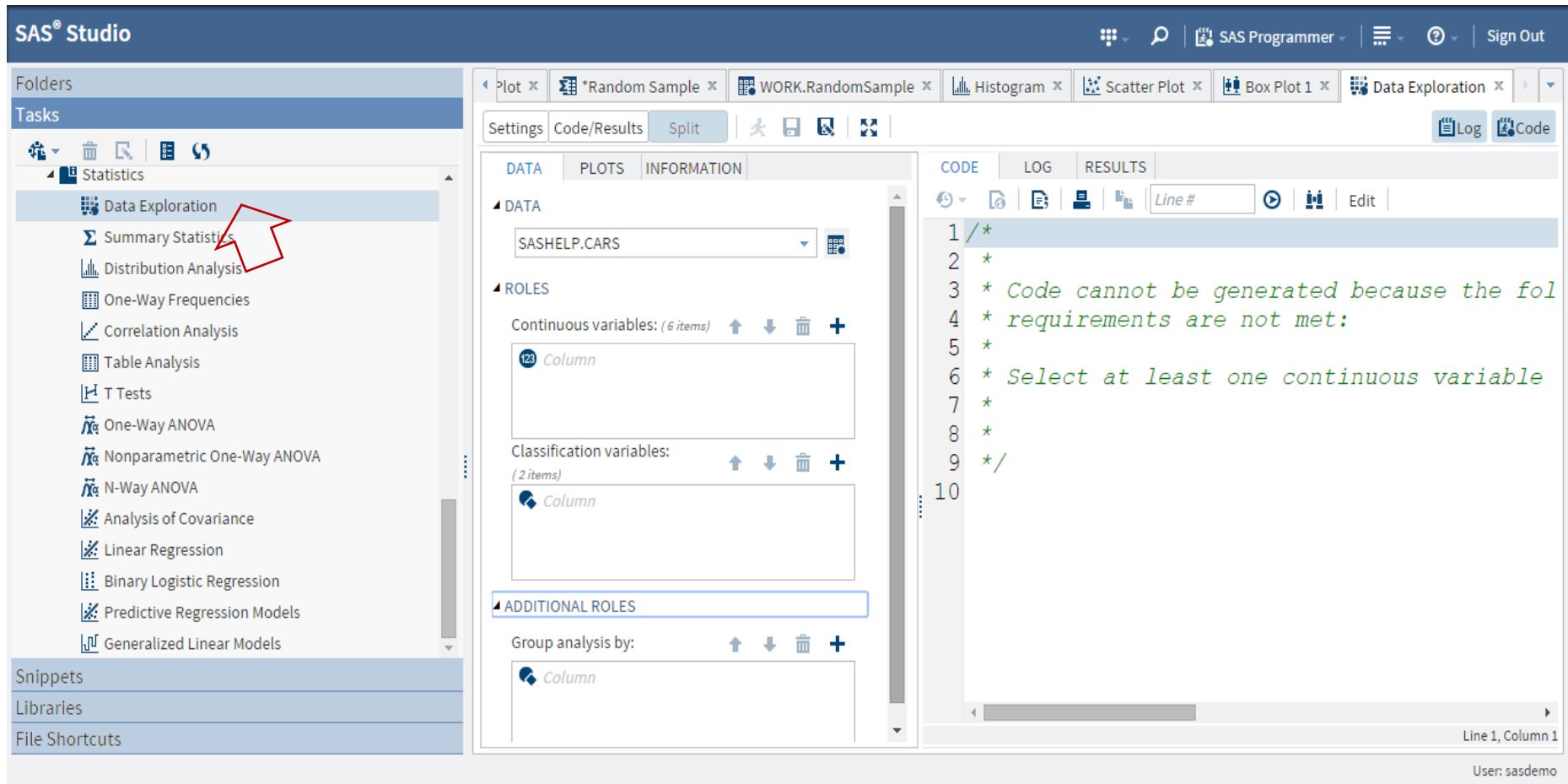
Data Exploration

Alphabetic List of Variables and Attributes					
#	Variable	Type	Len	Format	Label
9	Cylinders	Num	8		
5	DriveTrain	Char	5		
8	EngineSize	Num	8		Engine Size (L)
10	Horsepower	Num	8		
7	Invoice	Num	8	DOLLAR8.	
15	Length	Num	8		Length (IN)
11	MPG_City	Num	8		MPG (City)
12	MPG_Highway	Num	8		MPG (Highway)
6	MSRP	Num	8	DOLLAR8.	
1	Make	Char	13		
2	Model	Char	40		
4	Origin	Char	6		

Alphabetic List of Variables and Attributes					
#	Variable	Type	Len	Format	Label
3	Type	Char	8		
13	Weight	Num	8		Weight (LBS)
14	Wheelbase	Num	8		Wheelbase (IN)

DataScienceLab

Data Exploration: Task



The screenshot shows the SAS Studio interface. The title bar reads "SAS® Studio". The top menu bar includes "Plot", "Random Sample", "WORK.RandomSample", "Histogram", "Scatter Plot", "Box Plot 1", "Data Exploration", and "Sign Out". The left sidebar has sections for "Folders", "Tasks", "Snippets", "Libraries", and "File Shortcuts". The "Tasks" section is expanded, showing various statistical analysis options: Summary Statistics, Distribution Analysis, One-Way Frequencies, Correlation Analysis, Table Analysis, T Tests, One-Way ANOVA, Nonparametric One-Way ANOVA, N-Way ANOVA, Analysis of Covariance, Linear Regression, Binary Logistic Regression, Predictive Regression Models, and Generalized Linear Models. A red arrow points to the "Data Exploration" option. The main workspace is titled "DATA" and shows "SASHelp.CARS" selected under "Continuous variables". It also displays "Classification variables" and "Group analysis by". The "CODE" tab is active, showing the following code:

```
1 /*  
2 *  
3 * Code cannot be generated because the fol  
4 * requirements are not met:  
5 *  
6 * Select at least one continuous variable  
7 *  
8 *  
9 */
```

The status bar at the bottom right indicates "Line 1, Column 1" and "User: sasdemo".

DataScienceLab

Data Exploration: Plots

The screenshot shows the SAS Studio interface with two main panes. The left pane is titled 'PLOTS' and contains a list of plot types with checkboxes. The 'Scatter plot matrix' checkbox is checked. The right pane shows generated SAS code.

PLOTS Tab Content:

- SCATTER PLOT MATRIX
 - Scatter plot matrix
 - Add histograms
 - Add prediction ellipses
- PAIRWISE SCATTER PLOTS
 - Pairwise scatter plots
- REGRESSION SCATTER PLOTS
 - Regression scatter plots
- MOSAIC PLOT
 - Mosaic plot
- HISTOGRAM
 - Histogram
- BOX PLOT

Available when classification variable is

Generated SAS Code:

```
1 /*  
2 *  
3 * Task code generated by SAS Studio 3.3  
4 *  
5 * Generated on '03/12/2015 12:10'  
6 * Generated by 'sasdemo'  
7 * Generated on server 'LOCALHOST'  
8 * Generated on SAS platform 'Linux LIN X6  
9 * Generated on SAS version '9.04.01M2P072  
10 * Generated on browser 'Mozilla/5.0 (Winc  
11 * Generated on web client 'http://192.168  
12 *  
13 */  
14  
15 ods noproctitle;  
16 ods graphics / imagemap=on;  
17
```

Line 1, Column 1

User: sasdemo

DataScienceLab

Data Exploration: Selecting Variables

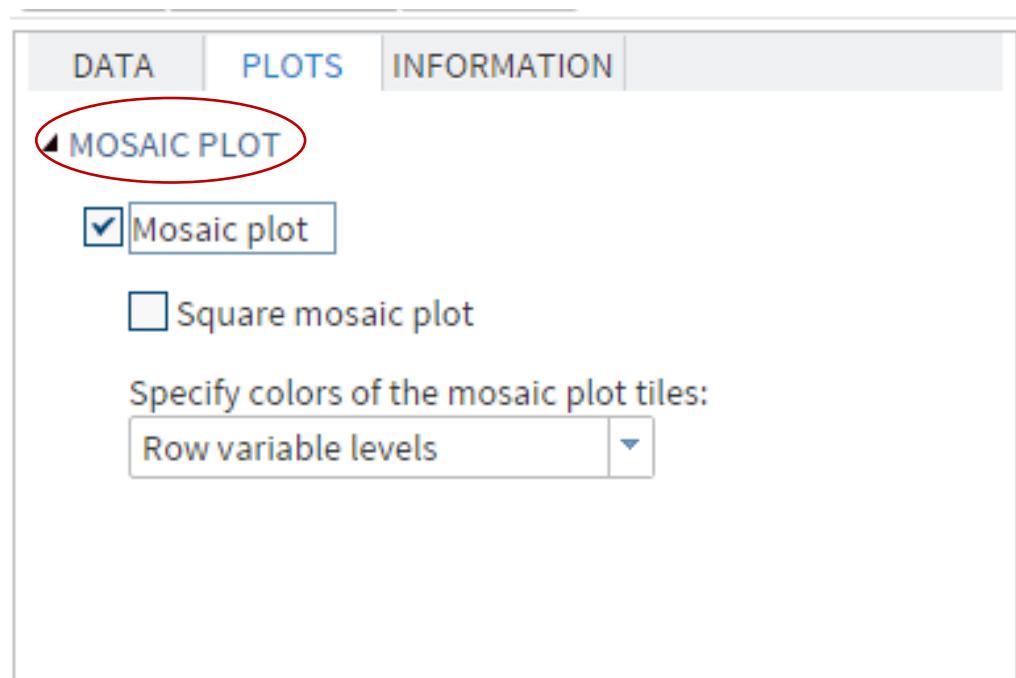
The image shows the SAS Studio interface. On the left, the Data Explorer pane displays the 'DATA' tab with 'SASHELP.CARS' selected. Under 'CONTINUOUS VARIABLES', there are 6 items. Under 'CLASSIFICATION VARIABLES', there are 2 items: 'Origin' and 'Type'. The 'Classification variables' section is highlighted with a red oval. On the right, the Code Editor pane shows SAS code for generating a mosaic plot:

```
16 ods graphics / imagemap=on;
17
18 %macro byGroupMosaic(data=, rowVar=, colV
19   ods select MosaicPlot;
20
21 proc freq data=&data;
22   tables (&rowVar)*(&colVar) / plot
23   run;
24
25 ods select all;
26 %mend byGroupMosaic;
27
28 %macro mosaicPlot(data=, numClassVars=, c
29   %local i j colVar rowVar;
30
31 /* Mosaic plot for class variables */
32
```

Line 46, Column 1

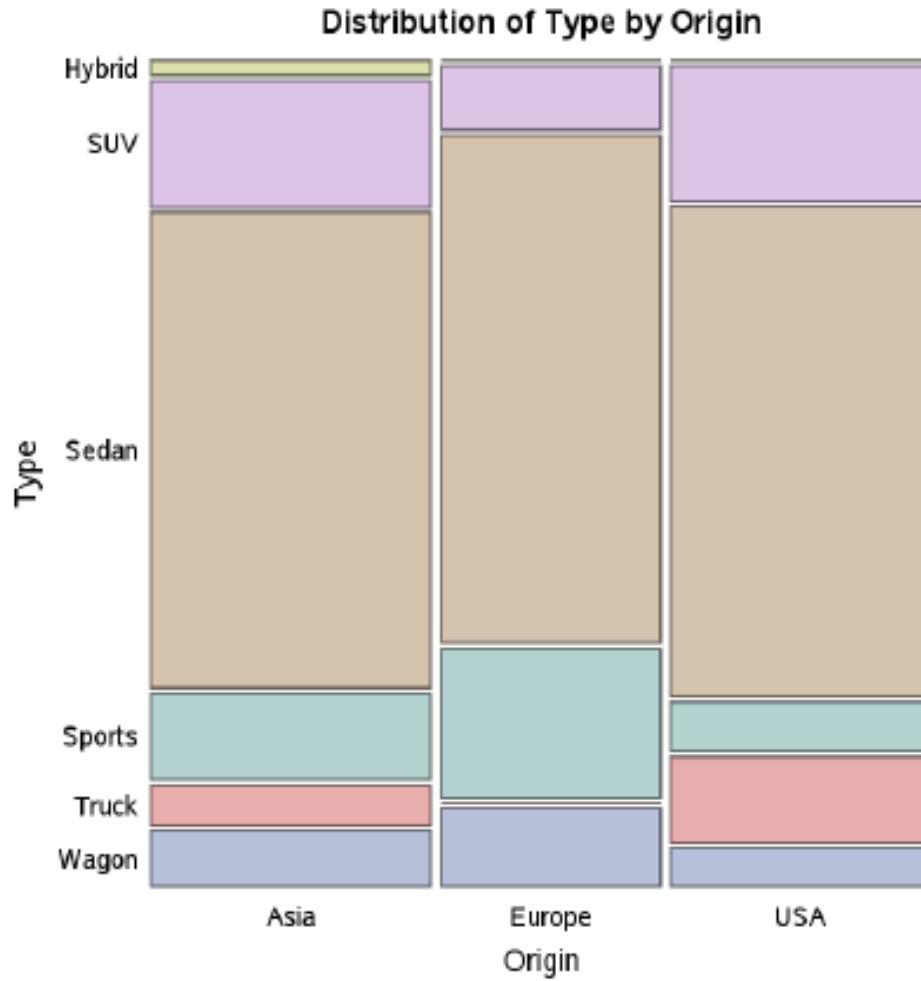
DataScienceLab

Data Exploration: Plots



DataScienceLab

Data Exploration: Results



In mosaic plot we compare, two categorical variables here, type and origin.

From the plot , it looks like, Asia region has more sedan cars than Europe and USA .

Whereas Europe makes more Sports cars than Asia and USA and less SUV than other regions.

Statistical Tasks

DataScienceLab

Correlation

- Correlation can tell you something about the relationship between variables.

It is used to understand:

- whether the relationship is positive or negative
- the strength of relationship.
- Statistical correlation is measured by what is called coefficient of correlation (r).
- Its numerical value ranges from +1.0 to -1.0. It gives us an indication of the strength of relationship.

Task

Find out correlation between EngineSize & MPG_Highway
using sashelp.cars dataset.

DataScienceLab

Correlation

SAS® Studio

Folders

Tasks

- My Tasks
- Tasks
 - Data
 - Econometrics
 - Graph
 - Combinatorics and Probability
 - Statistics
 - Data Exploration
 - Summary Statistics
 - Distribution Analysis
 - One-Way Frequencies
 - Correlation Analysis
 - Table Analysis
 - T Tests
 - One-Way ANOVA
 - Nonparametric One-Way ANOVA
- Snippets
- Libraries
- File Shortcuts

Program 1 x SASHHELP.CARS x *Correlation Analysis x *Correlation Analysis 1 x Correlation Analysis 2 x

Settings Code/Results Split Log Code

DATA OPTIONS OUTPUT INFORMATION

DATA

SASHHELP.CARS

ROLES

Analysis variables:

Column

Correlate with:

Column

Partial variables:

Column

ADDITIONAL ROLES

CODE LOG RESULTS

Line # Edit

```
1/*  
2*  
3 * Code cannot be generated because the following  
4 * requirements are not met:  
5 *  
6 * Select two or more analysis variables or one or more  
7 *  
8 *  
9 */
```

10

Line 1, Column 1

User: sasdemo

Correlation

The screenshot shows the SAS Studio 3.3 interface with the 'Correlation' task open. The left pane displays the task configuration, and the right pane shows the generated SAS code.

Task Configuration (Left Pane):

- DATA:** SASHHELP.CARS
- OPTIONS:** None
- OUTPUT:** None
- INFORMATION:** None
- DATA:** SASHHELP.CARS
- ROLES:**
 - Analysis variables:** EngineSize
 - Correlate with:** MPG_Highway
 - Partial variables:** Column
- ADDITIONAL ROLES:** None

Generated SAS Code (Right Pane):

```
1 /*  
2 *  
3 * Task code generated by SAS Studio 3.3  
4 *  
5 * Generated on '12/2/15, 2:24 PM'  
6 * Generated by 'sasdemo'  
7 * Generated on server 'LOCALHOST'  
8 * Generated on SAS platform 'Linux LIN X64 2.6.32-504  
9 * Generated on SAS version '9.04.01M2P07232014'  
10 * Generated on browser 'Mozilla/5.0 (Windows NT 6.3;  
11 * Generated on web client 'http://192.168.174.128/SAS  
12 *  
13 */  
14  
15 ods noproctitle;  
16 ods graphics / imagemap=on;  
17  
18 proc corr data=SASHHELP.CARS pearson nosimple noprob pl  
19   var EngineSize;  
20   with MPG_Highway;  
21 run;
```

User: sasdemo

Correlation : Result

Settings Code/Results Split | ⌂ ⌃ ⌚ ⌚ | ⌚ |

Log Code

CODE LOG RESULTS

CODE LOG DOWNLOAD | PRINT |

1 With Variables:	MPG_Highway
1 Variables:	EngineSize

	Engine Size
MPG_Highway	-0.71730
MPG (Highway)	

User: sasdemo

Linear Regression

In statistics, linear regression is an approach for modelling the relationship between a scalar dependent variable y and one or more explanatory variables (or independent variables) denoted X . The case of one explanatory variable is called simple linear regression.

Linear Regression Model :

$$Y_i = \beta_0 + \beta_1 + \varepsilon_i$$

Assumptions:

- The mean of Y is linearly related to X .
- Errors are normally distributed.
- Errors have equal variances.
- Errors are independent.

Linear Regression

SAS® Studio

Folders

Tasks

- Graph
- Combinatorics and Probability
- Statistics
 - Data Exploration
 - Summary Statistics
 - Distribution Analysis
 - One-Way Frequencies
 - Correlation Analysis
 - Table Analysis
 - T Tests
 - One-Way ANOVA
 - Nonparametric One-Way ANOVA
 - N-Way ANOVA
 - Analysis of Covariance
- Linear Regression
- Binary Logistic Regression
- Predictive Regression Models
- Generalized Linear Models

Snippets

Libraries

File Shortcuts

Program 1 x SASHHELP.HEART x *Linear Regression x SASHHELP.CLASS x Linear Regression 1 x

Settings Code/Results Split

DATA MODEL OPTIONS SELECTION O

DATA

SASHHELP.CLASS

ROLES

* Dependent variable: (1 item) Column +

Classification variables: Column +

Parameterization of Effects

Treatment of Missing Values

Continuous variables: Column +

ADDITIONAL ROLES

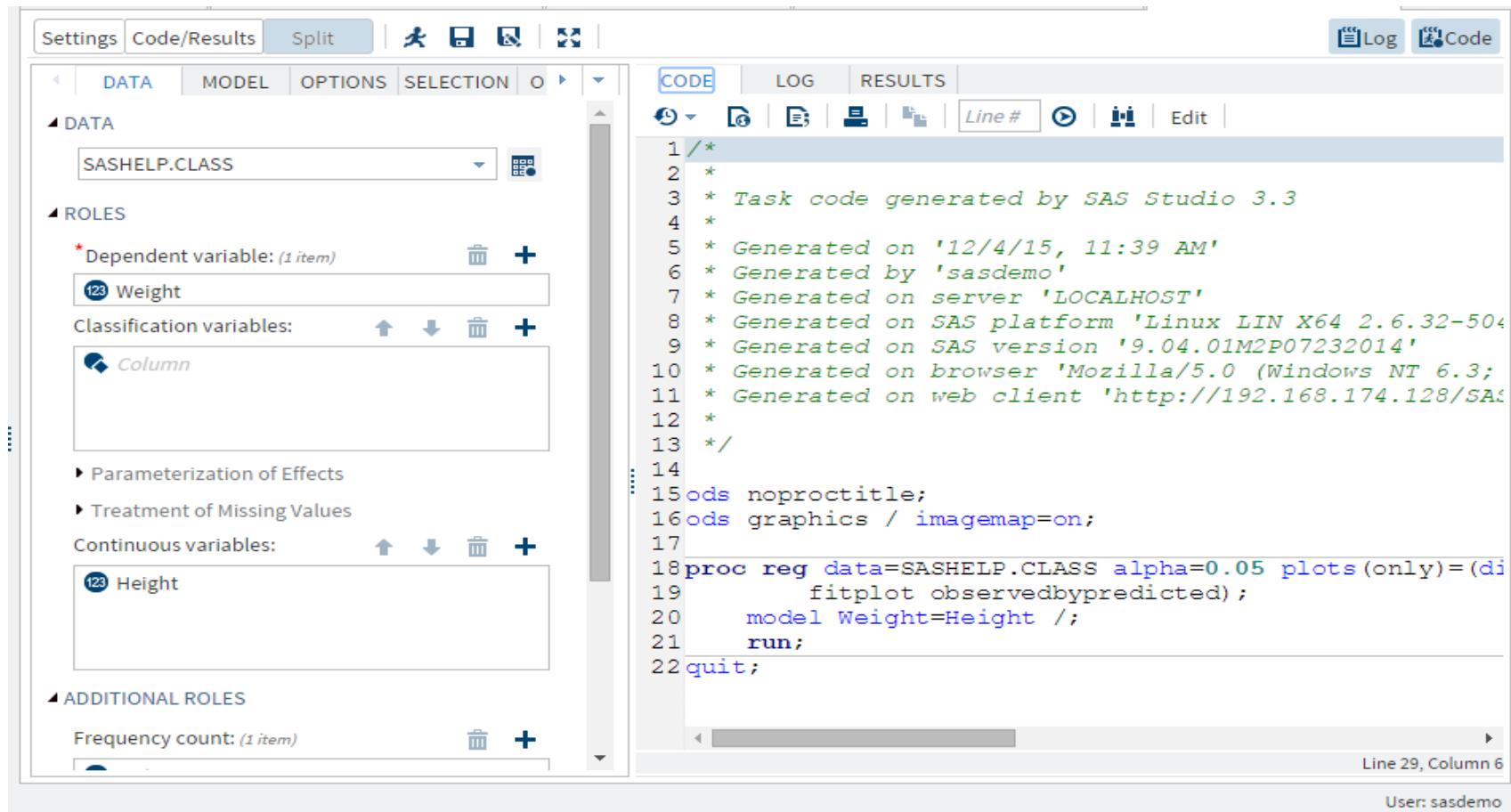
CODE LOG RESULTS

Line # Edit

```
1/*
2*
3* Code cannot be generated because the following
4* roles are not set:
5*
6* 1 Dependent variable
7*
8*
9*/
10
11
12/*
13*
14* Code cannot be generated because the following
15* requirements are not met:
16*
17* Select at least one continuous or one classificati
18*
19*
20*/
21
```

Line 1, Column 1

Linear Regression



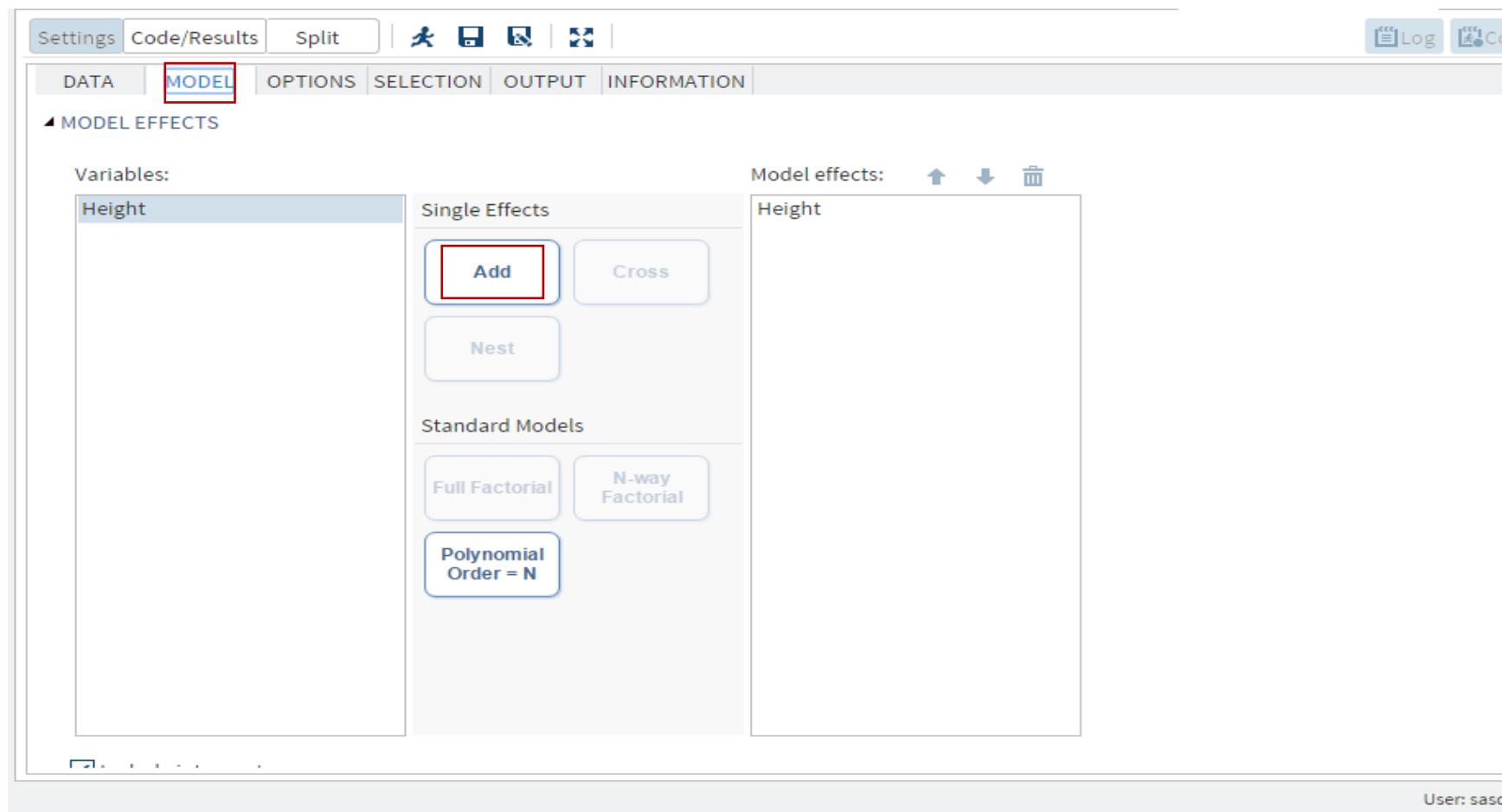
The image shows the SAS Studio interface with a 'Linear Regression' task open. On the left, the 'DATA' tab is selected, showing 'SASHelp.CLASS' as the input dataset. Under the 'ROLES' section, 'Weight' is listed as the dependent variable, and 'Height' is listed as a continuous variable. Other sections like 'Classification variables' and 'Continuous variables' are also present. On the right, the 'CODE' tab is selected, displaying the generated SAS code:

```
/*
 *
 * Task code generated by SAS Studio 3.3
 *
 * Generated on '12/4/15, 11:39 AM'
 * Generated by 'sasdemo'
 * Generated on server 'LOCALHOST'
 * Generated on SAS platform 'Linux LIN X64 2.6.32-504'
 * Generated on SAS version '9.04.01M2P07232014'
 * Generated on browser 'Mozilla/5.0 (Windows NT 6.3; '
 * Generated on web client 'http://192.168.174.128/SAS'
 *
 */
ods noproctitle;
ods graphics / imagemap=on;
proc reg data=SASHelp.CLASS alpha=0.05 plots(only)=(di
    fitplot observedbypredicted);
model Weight=Height /;
run;
quit;
```

Line 29, Column 6
User: sasdemo

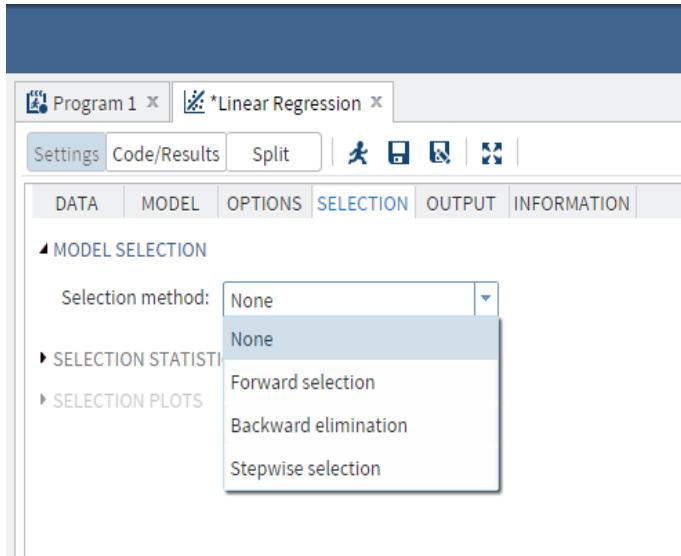
Classification variable specifies categorical variables that enter the regression model through the design matrix coding. Continuous variable specifies the numeric covariates (regressors) for the regression model. Both are independent variables.

Linear Regression



There are some more options for single effect like cross and nest. Also there are options for standard models also which are Full factorial, N-way factorial and Polynomial order = N.

Linear Regression



The **forward selection** technique begins with just the intercept and then sequentially adds the effect that most improves the fit. The process terminates when no significant improvement can be obtained by adding any effect

The **backward elimination** technique starts from the full model including all independent effects. Then effects are deleted one by one until a stopping condition is satisfied.

The **stepwise method** is a modification of the forward selection technique that differs in that effects already in the model do not necessarily stay there

Linear Regression

The screenshot shows a software interface for a statistical analysis. At the top, there are tabs for 'Settings', 'Code/Results' (which is selected), and 'Split'. Below these are icons for running, saving, and splitting data. On the right side, there are 'Log' and 'Code' buttons. A navigation bar at the bottom includes 'CODE', 'LOG', and 'RESULTS' (selected). Below the navigation bar are icons for file operations like open, save, and print.

Model: MODEL1
Dependent Variable: Weight

Number of Observations Read	19
Number of Observations Used	19

Analysis of Variance

Source	DF	Sum of Squares	Mean Square	F Value	Pr > F
Model	1	7193.24912	7193.24912	57.08	<.0001
Error	17	2142.48772	126.02869		
Corrected Total	18	9335.73684			

The ANOVA table provides an analysis of the variability observed in the data and the variability explained by the regression line.

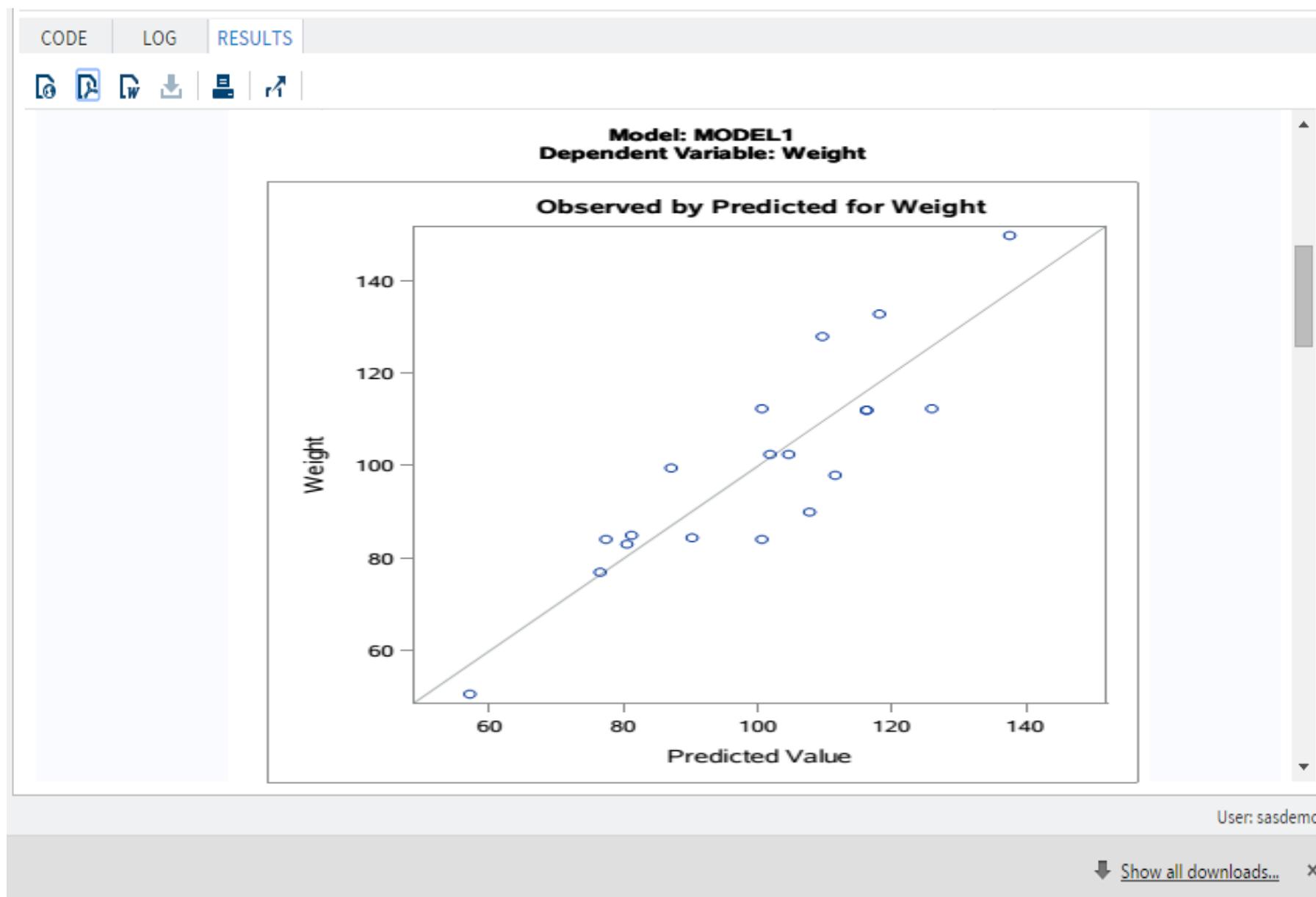
Linear Regression

Root MSE	11.22625	R-Square	0.7705
Dependent Mean	100.02632	Adj R-Sq	0.7570
Coeff Var	11.22330		

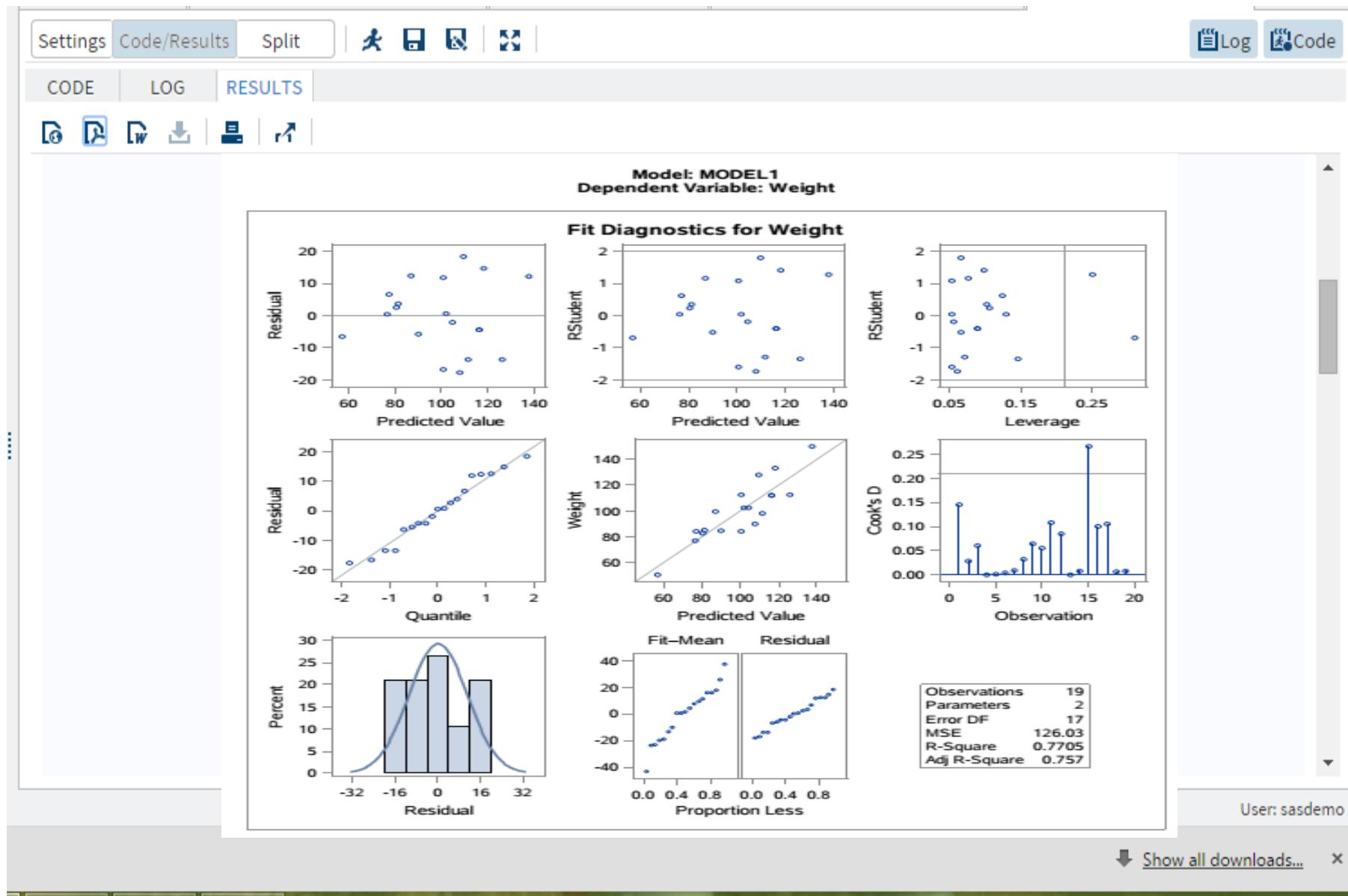
Parameter Estimates					
Variable	DF	Parameter Estimate	Standard Error	t Value	Pr > t
Intercept	1	-143.02692	32.27459	-4.43	0.0004
Height	1	3.89903	0.51609	7.55	<.0001

The parameter Estimates table defines the model for your data.

Linear Regression



Linear Regression



Assignment

- 1) Take a random sample which you have created using sashelp.heart. Check whether weight is based on height or not?

Data Science Lab

Data Visualization

Data visualization using Graph Tasks

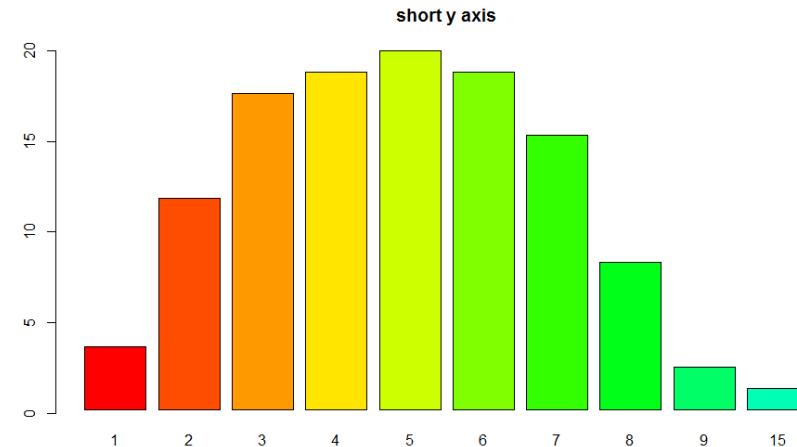
DataScienceLab

Purpose of Statistical Graphs

- To convey the data to the viewers in pictorial form
 - It is easier for most people to comprehend the meaning of data presented as a picture than data presented as a table. This is especially true if the viewers have little or no statistical knowledge
- To describe the data set
- To analyze the data set (Distribution of data set)
- To summarize a data set
- To discover a trend or pattern in a situation over a period of time
- To get the viewers' attention in a publication or speaking presentation

Bar Graphs

- Bar graph is a graphical display of data using bars of different heights.
- It is used in categorical data.
- Bar graph can be vertical and horizontal.



Bar Chart :Roles

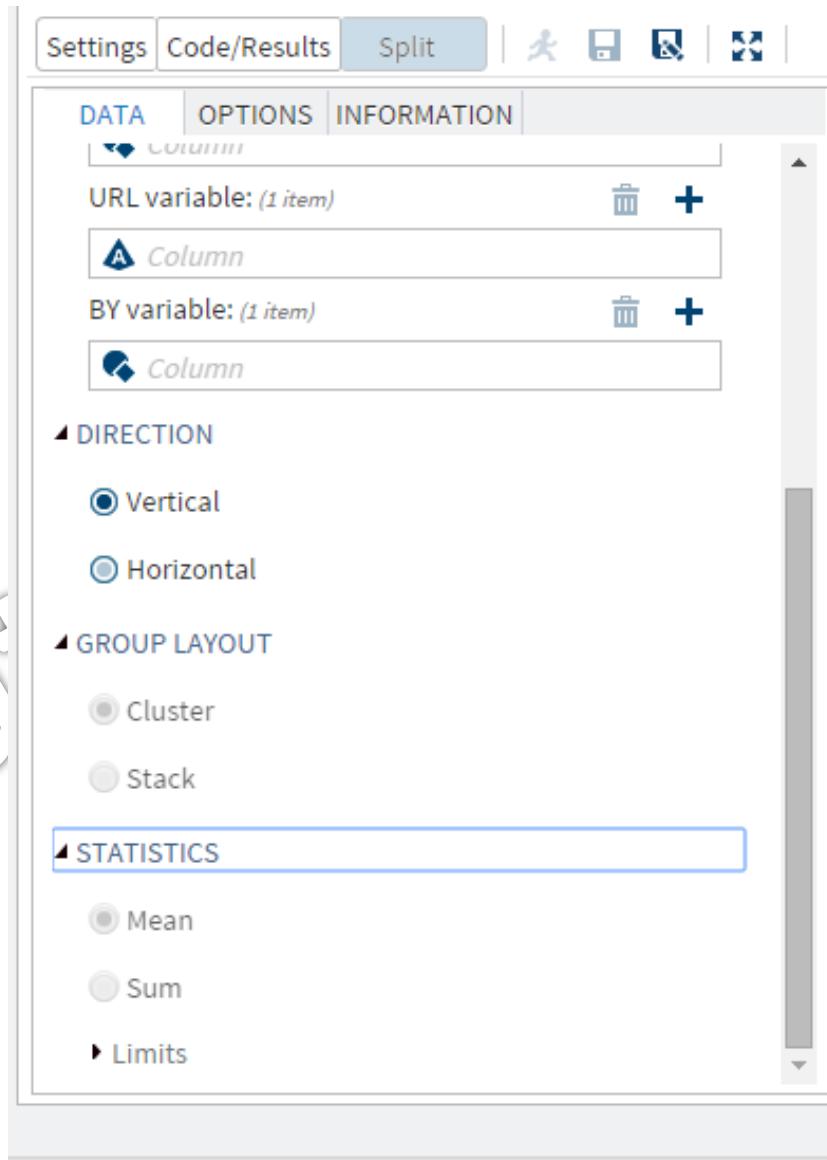
The screenshot shows the SAS Studio interface. In the center, a 'Bar Chart' task is being configured. The 'DATA' tab is selected, showing 'SASHHELP.CARS' as the source. Under the 'ROLES' section, there is one category variable ('Category variable: (1 item)') which is a 'Column'. The 'CODE' tab on the right displays the generated SAS code:

```
1 /*  
2 *  
3 * Code cannot be generated because the fol  
4 * roles are not set:  
*  
6 * 1 Category variable  
*/
```

A red arrow points from the 'Bar Chart' icon in the 'Graph' section of the 'Tasks' sidebar to the 'Bar Chart' task in the center.

DataScienceLab

Bar Chart : Data



Bar Chart:Selecting Variables

The screenshot shows the SAS Studio interface with the DATA tab selected. On the left, under the ROLES section, there are several variable assignments:

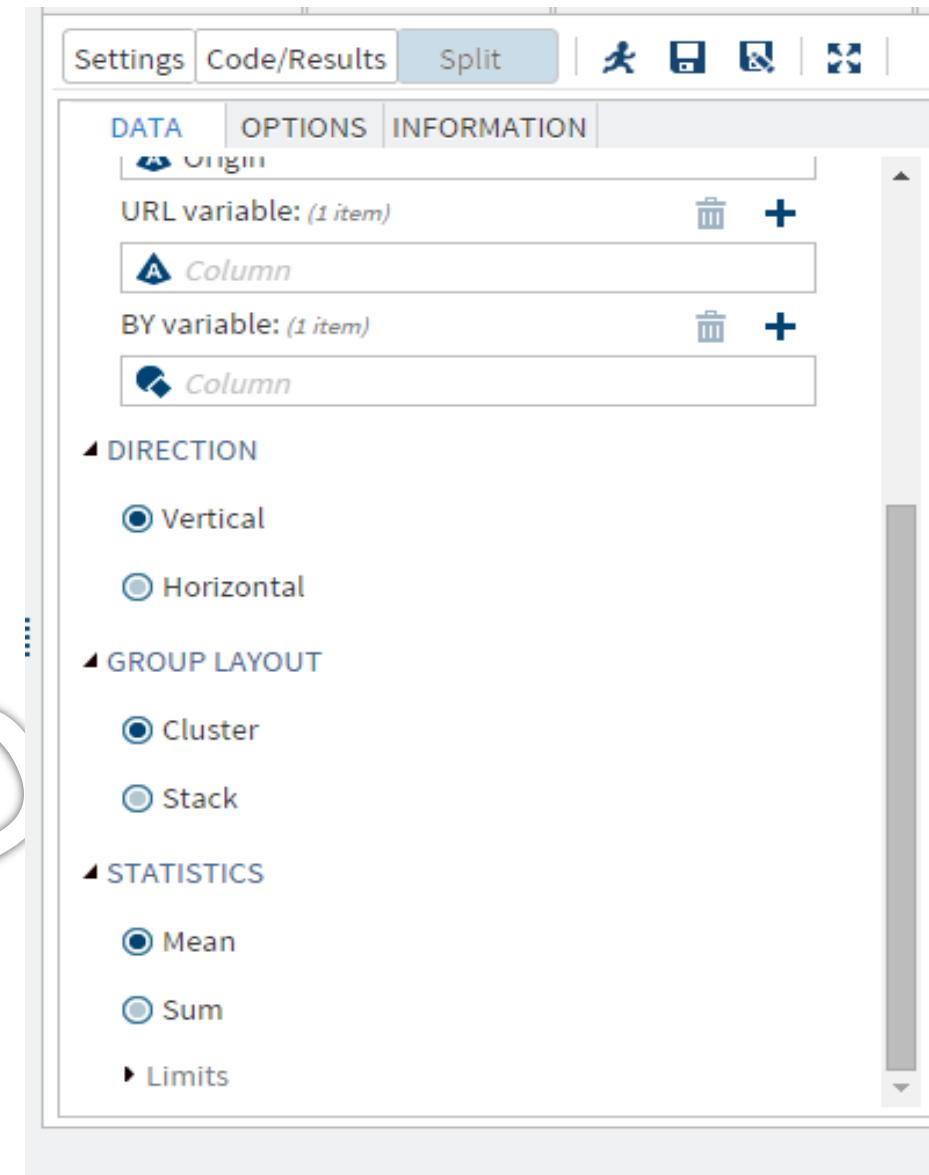
- * Category variable: (1 item) Type
- Response variable: (1 item) MPG_City
- Group variable: (1 item) Origin
- URL variable: (1 item) Column
- BY variable: (1 item) Origin

On the right, the CODE tab displays the generated SAS code:

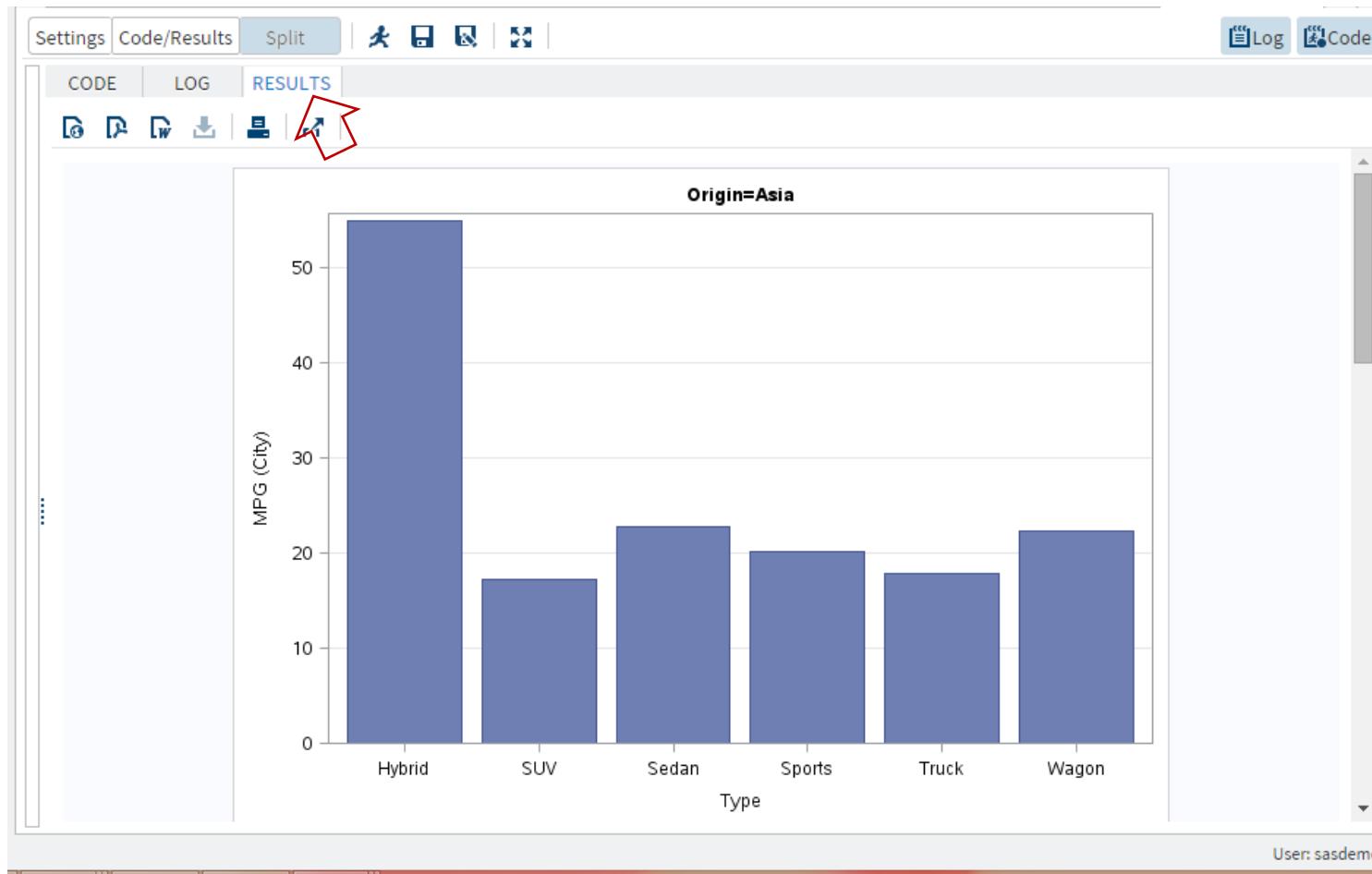
```
14  
15 /*--Sort data by BY variable--*/  
16 proc sort data=SASHELP.CARS out=_BarCha  
17   by Origin;  
18 run;  
19  
20 /*--Set output size--*/  
21 ods graphics / reset imagemap;  
22  
23 /*--SGPLOT proc statement--*/  
24 proc sgplot data=_BarChartTaskData noau  
25   /*--BY Variable--*/  
26   by Origin;  
27  
28   /*--Bar chart settings--*/  
29   vbar Type / response=MPG_City group  
30     name='Bar';  
31  
32   /*--Response Axis--*/  
33
```

DataScienceLab

Bar Chart:Selecting Variables



Bar Chart:Results

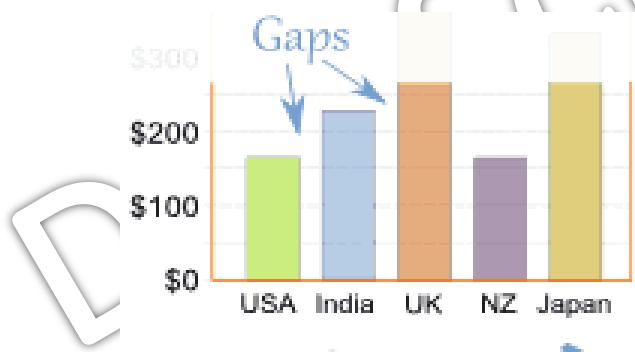


The bar plot is plotted for origin Asia .
In this plot hybrid type of car have good average .

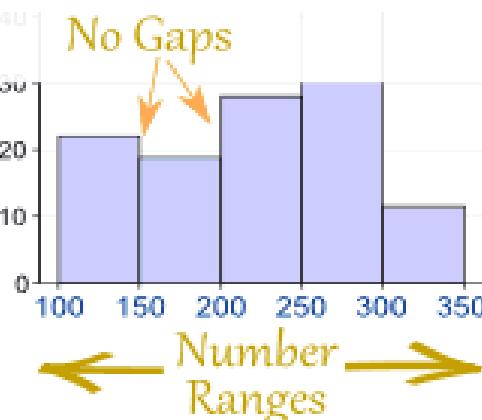
DataScienceLab

Histogram

- A bar graph that displays the data from a frequency distribution.
- It is used in continuous data.

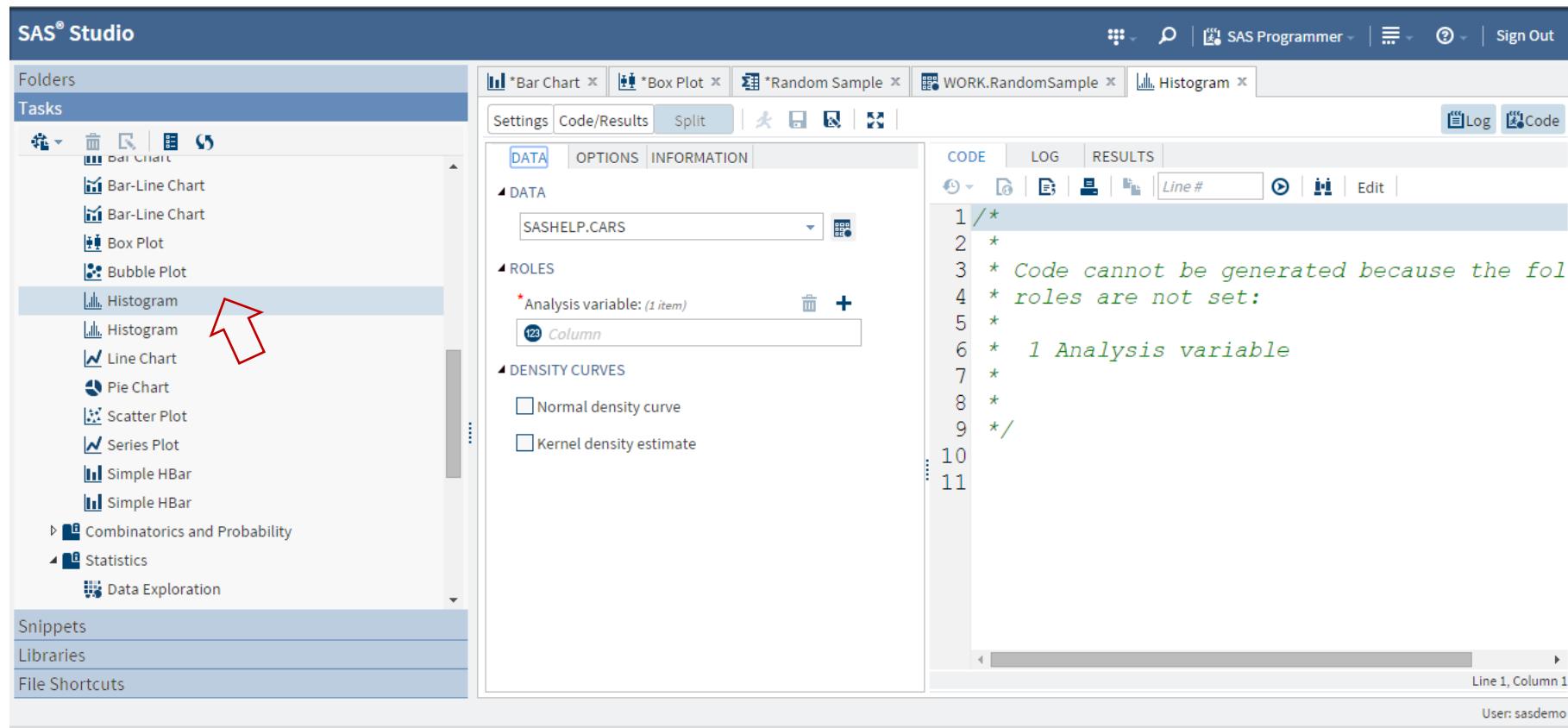


Bar Graph



Histogram

Histogram Plot :



DataScienceLab

Histogram Plot: Selecting Variables

The screenshot shows the SAS Studio interface with two main panes: DATA and CODE.

DATA Pane:

- SETTINGS tab is selected.
- DATA tab is selected.
- DATA source: SASHELP.CARS.
- ROLES section:
 - * Analysis variable: MPG_City (selected)
- DENSITY CURVES section:
 - Normal density curve
 - Kernel density estimate

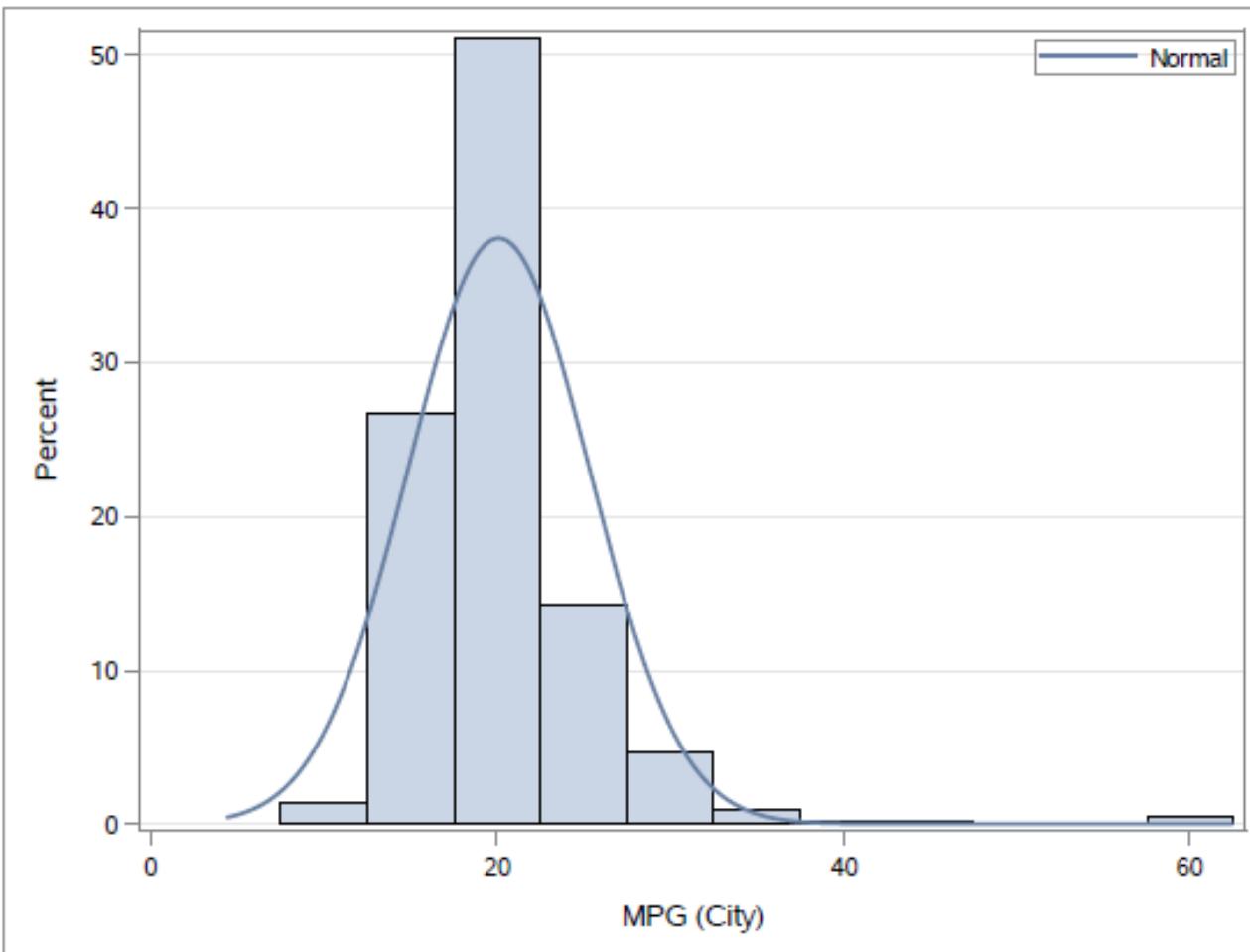
CODE Pane:

- CODE tab is selected.
- Code content:

```
14  
15 /* Option group 5 (GRAPH SIZE) parameters  
16 /**-Set output size--*/  
17 ods graphics / reset imagemap;  
18  
19 /**-SGPLOT proc statement--*/  
20 proc sgplot data=SASHELP.CARS;  
21     /*-Histogram settings--*/  
22     histogram MPG_City;  
23  
24     /*-Normal Density plot settings--*/  
25     density MPG_City;  
26  
27     /*-Vertical or Response Axis--*/  
28     yaxis grid;  
29     discretelegend "DENSITY" / location=i  
30 run;  
31  
32 ods graphics / reset;
```
- Log and Code buttons are visible at the top right.

DataScienceLab

Histogram Plot : Results

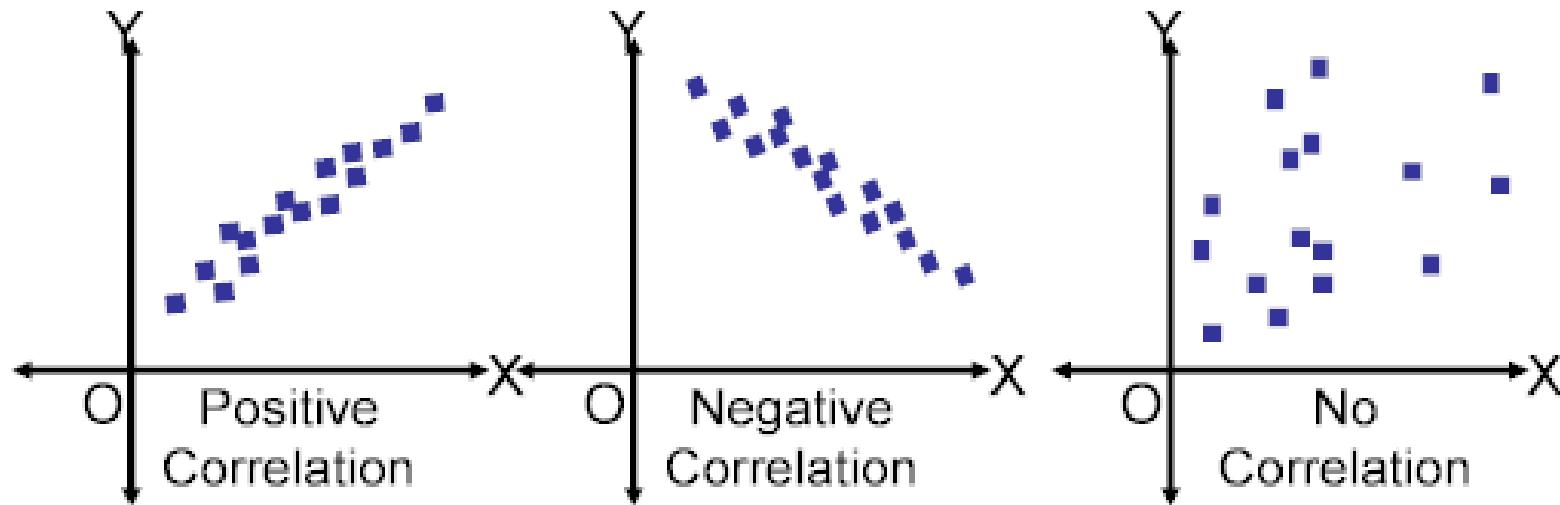


DataScienceLab

Scatter plot

- A scatter plot has points that show the relationship between two sets of data.
- Correlation :- when two set of data are strongly linked together.

SCATTER PLOT EXAMPLES



Scatter plot:Data

The screenshot shows the SAS Studio interface. The left sidebar has a 'Tasks' section with various chart types listed, and a red arrow points to the 'Scatter Plot' item, which is highlighted with a blue selection bar. The main workspace shows a 'Scatter Plot' tab selected in the top navigation bar. The central area has tabs for 'DATA', 'OPTIONS', and 'INFORMATION'. Under 'DATA', 'SASHelp.CARS' is selected. The 'ROLES' section contains fields for X variable (Column), Y variable (Column), Group variable (Column), Marker label variable (Column), and URL variable (Column). The 'CODE' tab on the right displays SAS code related to the scatter plot, indicating roles are not set:

```
/*
*
3 * Code cannot be generated because the fol
4 * roles are not set:
*
6 * 1 X variable
7 * 1 Y variable
*
*/

```

User: sasdemo

DataScienceLab

Scatter plot: Fit plots

The screenshot shows the SAS Studio interface with two main panes: DATA and CODE.

DATA Pane: Contains tabs for Settings, Code/Results, Split, and three icons. It also has tabs for DATA, OPTIONS, INFORMATION, and FIT PLOTS, with FIT PLOTS highlighted and circled in red. Below these are several options with checkboxes and input fields:

- Regression
- Confidence limits for means
- Prediction limits for individuals
- Alpha: 0.05
- Degree: 1
- Loess
- Confidence limits for means
- Alpha: 0.05
- PBSpline
- Confidence limits for means
- Prediction limits for individuals
- Alpha: 0.05

CODE Pane: Contains tabs for CODE, LOG, and RESULTS. The CODE tab is selected. It shows a code editor with the following content:

```
1 /*  
2 *  
3 * Code cannot be generated because the fol  
4 * roles are not set:  
5 *  
6 * 1 X variable  
7 * 1 Y variable  
8 *  
9 *  
10 */  
11  
12
```

Below the code editor, status information includes "Line 1, Column 1" and "User: sasdemo".

DataScienceLab

Scatter plot : Selecting Variables

The screenshot shows the SAS Studio interface with two main panels: the left panel displays the configuration for a scatter plot, and the right panel shows the generated SAS code.

Left Panel (Configuration):

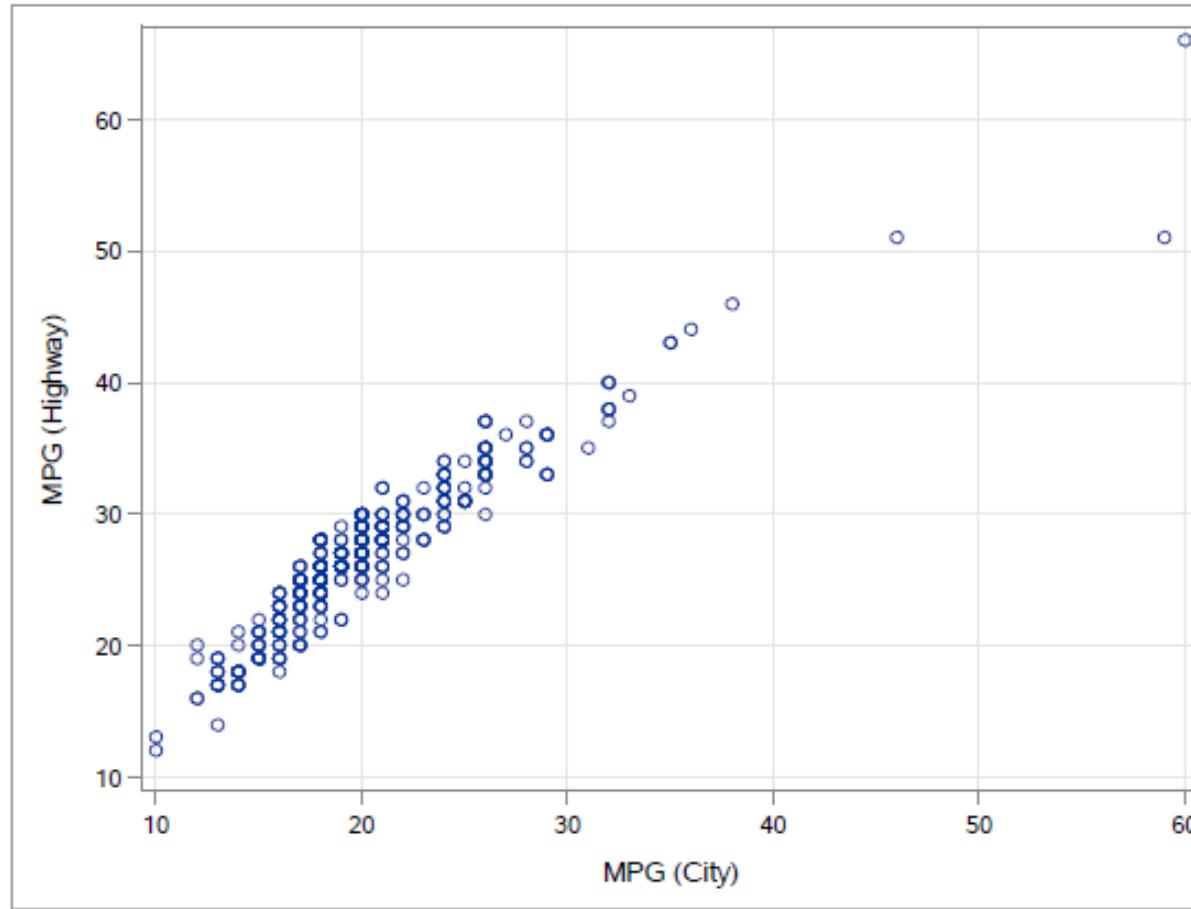
- DATA:** SASHELP.CARS
- ROLES:** X variable: MPG_City, Y variable: MPG_Highway
- FIT PLOTS:** Regression selected

Right Panel (Code):

```
Line # Generated on Wed, 2018-01-10 12:00:12  
12 *  
13 */  
14  
15 /*--Set output size--*/  
16 ods graphics / reset imagemap;  
17  
18 /*--SGPLOT proc statement--*/  
19 proc sgplot data=SASHELP.CARS;  
20     /*--Scatter plot settings--*/  
21     scatter x=MPG_City y=MPG_Highway / transparent  
22  
23     /*--X Axis--*/  
24     xaxis grid;  
25  
26     /*--Y Axis--*/  
27     yaxis grid;  
28 run;  
29  
30
```

DataScienceLab

Scatter plot



In this plot we understand it is a positive relationship between mpg_city and mpg_highway.

DataScienceLab

Scatter plot

The screenshot shows the SAS Studio interface with two main panes: DATA and CODE.

DATA Pane:

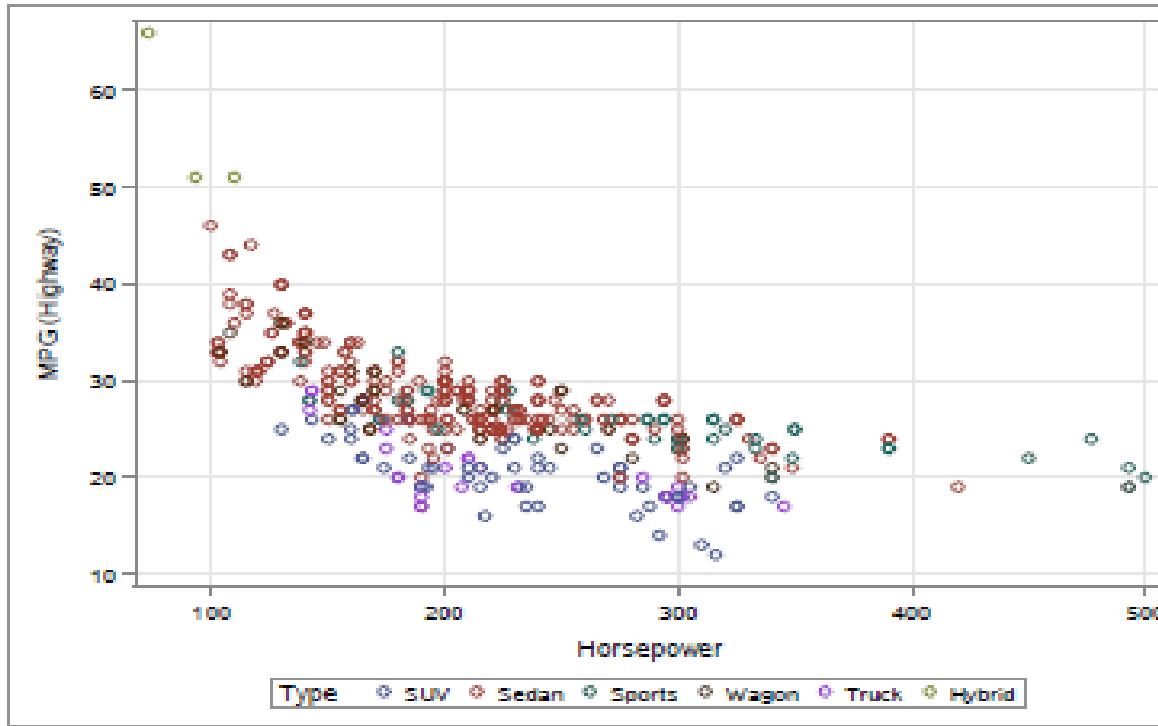
- DATA tab selected.
- Dataset: SASHHELP.CARS
- ROLES section circled in red:
 - X variable: Horsepower
 - Y variable: MPG_Highway
 - Group variable: Type
 - Marker label variable: Column
 - URL variable: Type

CODE Pane:

```
15 /*--Set output size--*/
16 ods graphics / reset imagemap;
17
18 /*--SGPLOT proc statement--*/
19 proc sgplot data=SASHHELP.CARS;
20   /*--Scatter plot settings--*/
21   scatter x=Horsepower y=MPG_Highway /
22     name='Scatter';
23
24   /*--X Axis--*/
25   xaxis grid;
26
27   /*--Y Axis--*/
28   yaxis grid;
29 run;
30
31 ods graphics / reset;
```

DataScienceLab

Scatter plot

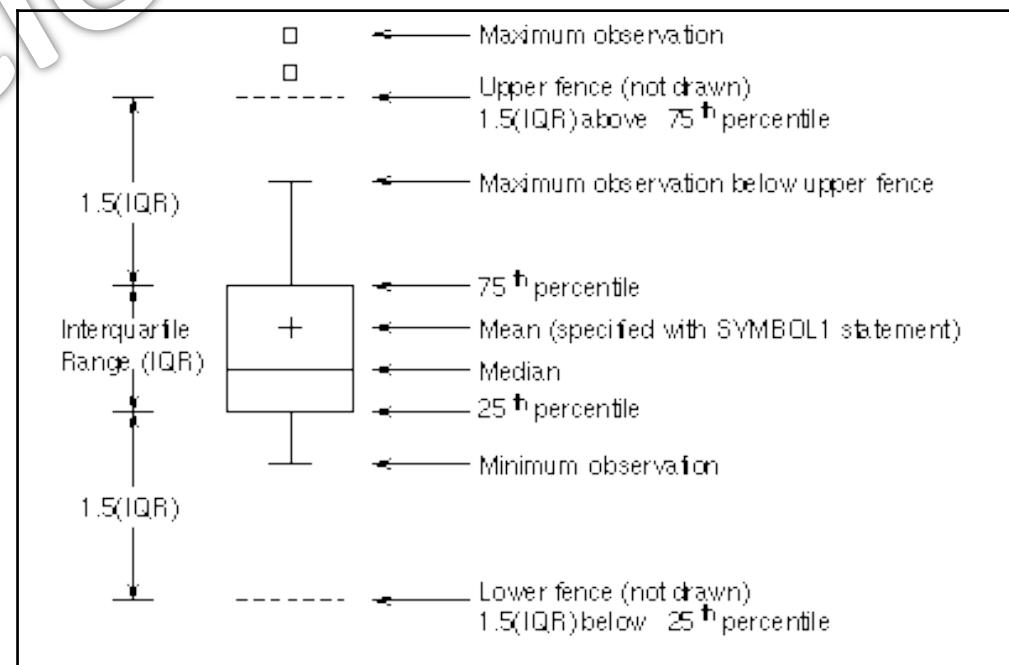


In this plot we understand that there is negative relationship between mpg_highway and horsepower according to their types of cars .

DataScienceLab

Box and Whisker plot

1. The center horizontal line in each box corresponds to the sample median and the central plus sign corresponds to the sample mean.



Box and Whisker plot:Data

The screenshot shows the SAS Studio interface. The left sidebar contains a 'Tasks' section with various chart types: Bar Chart, Box Plot (which is selected and highlighted with a red arrow), Bubble Plot, Histogram, Line Chart, Pie Chart, Scatter Plot, Series Plot, Simple HBar, Combinatorics and Probability, Statistics, and Data Exploration. The main workspace has three panes: 'DATA' (set to SASHelp.CARS), 'CODE' (containing generated code for a box plot), and 'LOG' (empty). The 'CODE' pane shows:

```
1 /*  
2 *  
3 * Code cannot be generated because the fol  
4 * roles are not set:  
5 *  
6 * 1 Analysis variable  
7 *  
8 *  
9 */  
10  
11
```

User: sasdemo

DataScienceLab

Box and Whisker plot:Options

The screenshot shows the SAS Visual Analytics interface with the 'OPTIONS' tab selected in the top navigation bar. The left panel contains various configuration options for a box and whisker plot:

- Box width: 0.4
- Fill
-
- Data skin: None
- Transparency: 0
- Set cap shape
- Cap Shape: Serif
- Notches

Below these options are expandable sections:

- ▶ GROUP LAYOUT
- ▶ CATEGORY AXIS
- ▶ ANALYSIS AXIS
- ▶ LEGEND DETAILS
- ▶ GRAPH SIZE

The right panel displays the generated code:

```
1 /*  
2 *  
3 * Code cannot be generated because the fo.  
4 * roles are not set:  
5 *  
6 * 1 Analysis variable  
7 *  
8 *  
9 */  
10  
11
```

At the bottom right, it says "User: sasdemo".

DataScienceLab

Box and Whisker plot

The screenshot shows the SAS Studio interface with two main panes: DATA and CODE.

DATA Pane:

- WHERE CLAUSE FILTER:**
 - Apply where clause
 - Where string:
Type in ("Sedan", "Sports")
 - Include as footnote
- ROLES (circled in red):**
 - * Analysis variable: (1 item) MPG_City
 - Category variable: (1 item) Origin
 - Group variable: (1 item) Type
 - BY variable: (1 item) Column
- DIRECTION:**

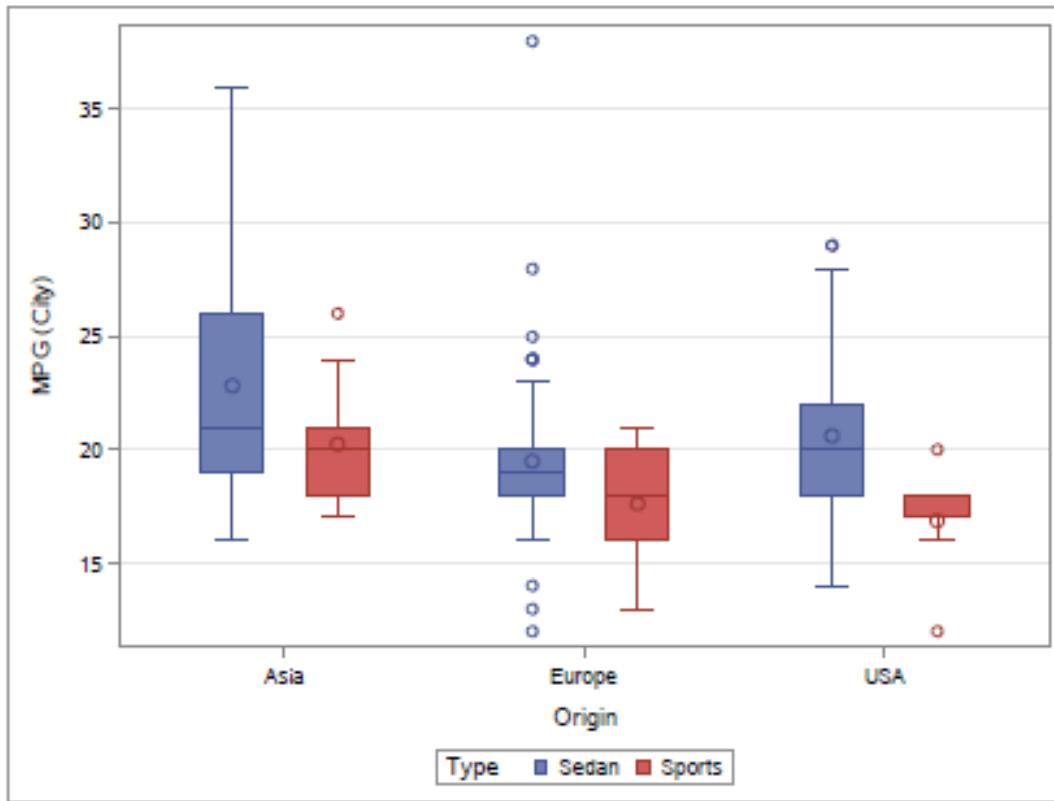
CODE Pane:

```
14 /*--Set output size--*/
15 ods graphics / reset imagemap;
16
17
18 /*--SGPLOT proc statement--*/
19 proc sgplot data=SASHELP.CARS (whe
20      /*--TITLE and FOOTNOTE--*/
21      /*--Box Plot settings--*/
22      vbox MPG_City / category=Origin group
23
24      /*--Category Axis--*/
25      xaxis fitpolicy=splitrotate;
26
27      /*--Response Axis--*/
28      yaxis grid;
29 run;
30
31
```

Line 1, Column 1
User: sasdemo

DataScienceLab

Box and Whisker plot



In this plot we understand that sedan is more in Asia .

The box is plotted according to the type of cars and their respective mpg(miles per gallon).The sedan type of car give good average then sports car in Asia.

The sample mean is identify by the dot and median is the central tendency of the plot.

Assignment1

Plot a bar chart for chol_status by using sashelp.heart

Assignment2

Plot a histogram for height by using sashelp.class

Assignment3

Plot a scatter plot of x variable as weight and y variable as height from sashelp.class. Explore the graph and note down the points that you understand and also find the relationship between them .

Assignment4

Plot a box plot of analysis variable as weight and category variable as chol_status from sashelp.heart.

DataScienceLab

Thank you!