Learning outcomes:

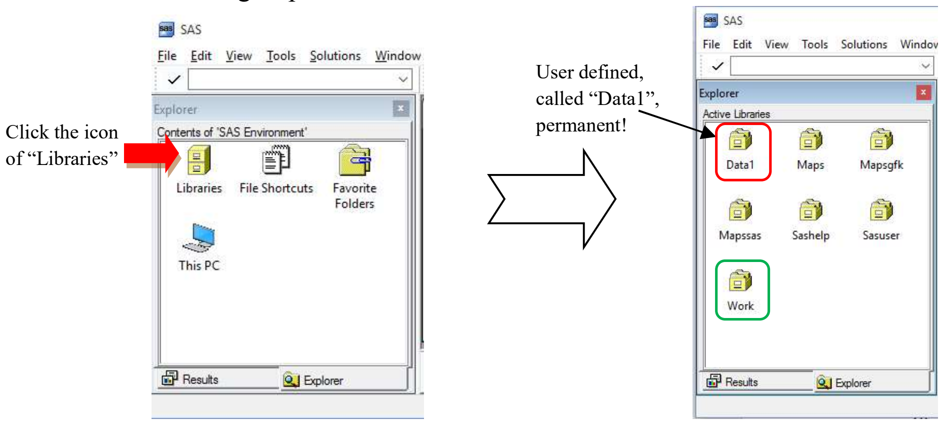
1. SAS libraries
2. SAS datasets
3. Creating SAS data sets from in-stream data
4. Reading a SAS data set
5. Subsetting a SAS data set
6. Creating permanent format for SAS data

SAS components learnt:

1. PROC CONTENTS
2. LIBNAME statement
3. PROC PRINT
4. VIEWTABLE window
5. DATA steps
6. DATALINES statement (DATALINES4)
7. LABEL statement
8. FORMAT statement
9. PROC FORMAT

2.1 SAS Libraries

- Browsing a SAS data library
  - Using Explorer Window


Click the icon of “Libraries”
SAS
File Edit View Tools Solutions Window
Explorer
Contents of 'SAS Environment'
Libraries
File Shortcuts
Favorite Folders
This PC
Results
Explorer
User defined, called “Data1”, permanent!
Active Libraries
Data1
Maps
Mapsgfk
Mapssas
Sashelp
Sasuser
Work

- Note that the user defined permanent library is created by the following LIBNAME statement:

```
LIBNAME DATA1 ‘D:\SAS_DATASETS’;
```

  - Using Procedure CONTENTS

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

NODS is used when “NO Descriptor portion” is required and it is used ONLY in conjunction with _ALL_ keyword.

Ex. 2.1: Use procedure CONTENTS to list all the SAS files under permanent library DATA1 and give the output.

SAS program:

```
libname DATA1 "D:\SAS_Datasets";

Proc Contents
run;
```

Output window:

```
                            The SAS System          11:31 Monday, June 11, 2016   2
                          The CONTENTS Procedure

                                 Directory
                      Libref          DATA1
                      Engine          V9
                      Physical Name   D:\SAS_Datasets
                      Filename        D:\SAS_Datasets


                         Member   File
           #  Name          Type     Size  Last Modified

           1  ELECTRICITY  DATA     9216  10Jul12:17:52:49
```

Directory Information

Library members

- Permanent and temporary libraries:
  - SAS stores the data files into libraries.
  - A SAS data set name consists of two parts: <Libref>.<SAS data set file name>, where "Libref" is the name of the library.

Library Name	**Storage nature**
-"Sasuser" - user-defined in **LIBNAME** statement	Permanent
Work (or Libref not specified)	Temporary

  - LIBNAME statement is used to define a **permanent** library. Once the LIBNAME statement is submitted, the library is always available during subsequent SAS sessions unless the LIBNAME statement is changed or clear.

```
LIBNAME Libref 'physical data files location';
```

  - The name of library (libref):
    - must be **8 characters or less**;
    - must begin with a letter or underscore;
    - must have letters, numerals, or underscores as the remaining characters.
  - The **temporary** library remains in effect until you end the SAS session. All of files are deleted when the SAS session ends.
  - "sas7bdat"is the extension of SAS data sets. Other files can be stored in the directory, but only the files that have SAS file extensions are recognized as part of the SAS library.
  - To disassociate a Libref, use LIBNAME statement:

```
Libname libref clear;
```

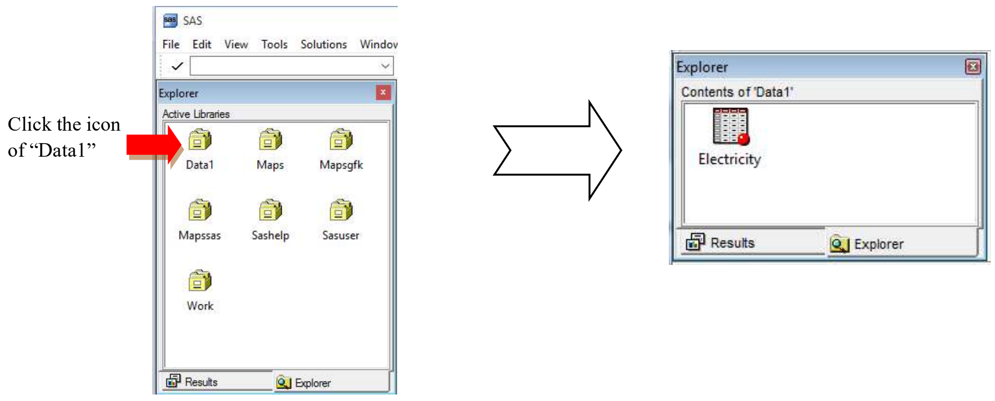
    SAS disconnects from the data source and closes any resources that are associated with that libref's connection.

  - An example - SAS program p1_electricity.sas:

```
libname DATA1 'D:\SAS_Datasets';

data testing;
set DATA1.electricity;
…
```

testing ⟺ Work.testing



- DATA1 is a **permanent** library and is defined in the location "d:\SAS_Datasets".
- "Electricity" is the dataset and it is referenced as "DATA1. ELECTRICITY" in SAS program. Its file name in d drive is "electricity.sas7bdat".
- Data set "testing" is **temporary**.

2.2 SAS data sets

- SAS data set names and variable names
  - must be 1 to 32 characters long
  - must begin with a letter (A-Z, either uppercase or lowercase) or an underscore (_)
  - continue with any combination of numbers, letters, or underscores.
  - are NOT case sensitive.
  - Examples of valid data set names and variable names: Payroll, LABDATA_1997, _EstimatedTuitionFee3.

- Terminology of data set in SAS
  - Each data set is tabulated in a SAS table as follows.

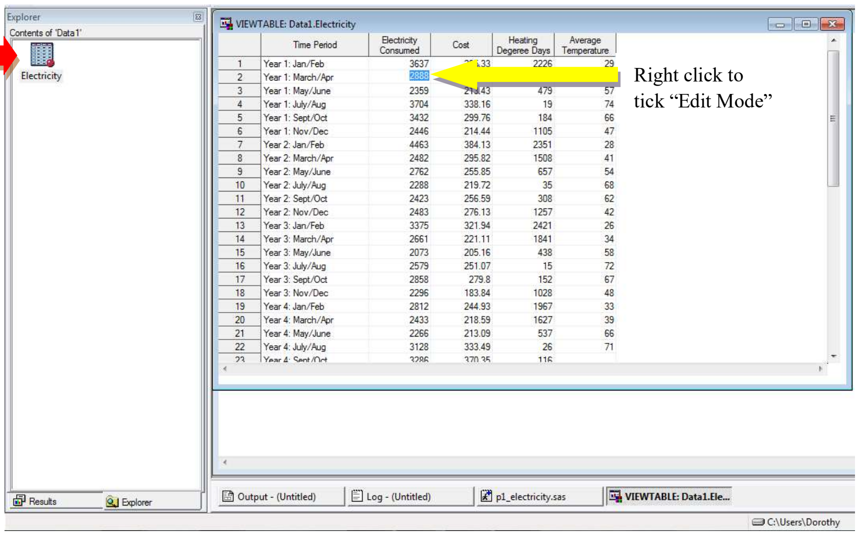
1st observation

	Variable 1	Variable 2	......		
1					
⁝					

The observations are tabulated as rows and each column represents a variable.

- Browsing the data
  - Use VIEWTABLE by double clicking the SAS data set icon in the explorer window. Then highlight a data value on the VIEWTABLE window and then right click to switch on "Edit Mode".

Double click the SAS data set icon.



	Time Period	Electricity Consumed	Cost	Heating Degeree Days	Average Temperature
1	Year 1: Jan/Feb	3637	[illegible]	2226	29
2	Year 1: March/Apr	2888			
3	Year 1: May/June	2359	[illegible]	479	57
4	Year 1: July/Aug	3704	338.16	19	74
5	Year 1: Sept/Oct	3432	299.76	184	66
6	Year 1: Nov/Dec	2446	214.44	1105	47
7	Year 2: Jan/Feb	4463	384.13	2351	28
8	Year 2: March/Apr	2482	295.82	1508	41
9	Year 2: May/June	2762	255.85	657	54
10	Year 2: July/Aug	2288	219.72	35	68
11	Year 2: Sept/Oct	2423	256.59	308	62
12	Year 2: Nov/Dec	2483	276.13	1257	42
13	Year 3: Jan/Feb	3375	321.94	2421	26
14	Year 3: March/Apr	2661	221.11	1841	34
15	Year 3: May/June	2073	205.16	438	58
16	Year 3: July/Aug	2579	251.07	15	72
17	Year 3: Sept/Oct	2858	279.8	152	67
18	Year 3: Nov/Dec	2296	183.84	1028	48
19	Year 4: Jan/Feb	2812	244.93	1967	33
20	Year 4: March/Apr	2433	218.59	1627	39
21	Year 4: May/June	2266	213.09	537	66
22	Year 4: July/Aug	3128	333.49	26	71
23	Year 4: Sept/Oct	3286	370.35	116	

Right click to tick "Edit Mode"

Right click again to add/delete row(s) or column(s). Note that once the changes are made, the changes are saved automatically.

- Use PROC PRINT for listing of all observations, all variables and an OBS column on the left:

```
PROC PRINT DATA=SAS-data-set;
run;
```

- Use PROC PRINT for listing of a portion of data:

```
PROC PRINT DATA=SAS-data-set <(FIRSTOBS=m OBS=n) NOOBS>;
  VAR variable1 variable2 variable3 …;
run;
```

  - NOOBS option suppresses the observation numbers column in the report and VAR statement selects variables that appear in the report and determines their order. NOOBS option can be used alone.
  - FIRSTOBS specifies the number of the first observation and OBS specifies the number of the last observation to be printed in output window.

Ex. 2.2: Use procedure PRINT to list the following in the output window:
(a) all observations of variables "Time_period", "electricity_consumed" and "average_temperature" of DATA1.ELECTRICITY.
(b) Observations 11 to 24 of part (a).

```
/*p2ex2.sas*/
* a program to list electricity data ;
libname DATA1 "D:\SAS_Datasets";

* part (a);
Proc Print data=DATA1.electricity;
  var time_period
run;

* part (b);
Proc Print                        (firstobs=          );
  var
run;
```

Output window of part (b):

```
                             The SAS System          11:31 Monday, June 11, 2012  13

                                              Electricity_     Average_
          Obs       Time_Period                 Consumed      Temperature

           11    Year 2: Sept/Oct                 2423             62
           12    Year 2: Nov/Dec                  2483             42
           13    Year 3: Jan/Feb                  3375             26
           14    Year 3: March/Apr                2661             34
           15    Year 3: May/June                 2073             58
           16    Year 3: July/Aug                 2579             72
           17    Year 3: Sept/Oct                 2858             67
           18    Year 3: Nov/Dec                  2296             48
           19    Year 4: Jan/Feb                  2812             33
           20    Year 4: March/Apr                2433             39
           21    Year 4: May/June                 2266             66
           22    Year 4: July/Aug                 3128             71
           23    Year 4: Sept/Oct                 3286              .
           24    Year 4: Nov/Dec                  2749              .
```

- Descriptor Portion
  - The descriptor portion of a SAS data set contains information about the data set, including
    - General information of the data set, e.g. the name of the data set, the date and time that the data set was created, the number of observations, the number of variables, etc.
    - Data information, e.g. variable name, type and length.
  - PROC CONTENTS is used to display the descriptor portion.

```
PROC CONTENTS DATA=SAS-Data-Set;
RUN;
```

  - An example of data set: DATA1.ELECTRICITY

```
PROC CONTENTS DATA=DATA1.ELECTRICITY; /*p2_contents.sas*/
RUN;
```

The output:

```
                                        The SAS System          11:31 Monday, June 11, 2012  14

                                    The CONTENTS Procedure

    Data Set Name        DATA1.ELECTRICITY                     Observations          48
    Member Type          DATA                                  Variables             5
    Engine               V9                                    Indexes               0
    Created              Monday, March 24, 2008 05:30:15 PM    Observation Length    56
    Last Modified        Monday, March 24, 2008 05:30:15 PM    Deleted Observations  0
    Protection                                                 Compressed            NO
    Data Set Type                                              Sorted                NO
    Label
    Data Representation  WINDOWS_32
    Encoding             wlatin1  Western (Windows)


                               Engine/Host Dependent Information

              Data Set Page Size          8192
              Number of Data Set Pages    1
              First Data Page             1
              Max Obs per Page            145
              Obs in First Data Page      48
              Number of Data Set Repairs  0
              Filename                    d:\SAS_Datasets\electricity.sas7bdat
              Release Created             3.2TK
              Host Created                XP_PR


                           Alphabetic List of Variables and Attributes

              #    Variable                 Type    Len    Label

              5    Average_Temperature      Num       8    Average Temperature
              3    Cost                     Num       8    Cost
              2    Electricity_Consumed     Num       8    Electricity Consumed
              4    Heating_Degree_Days      Num       8    Heating Degree Days
              1    Time_Period              Char     17    Time Period
```

In alphabetical order, not in the variable list order

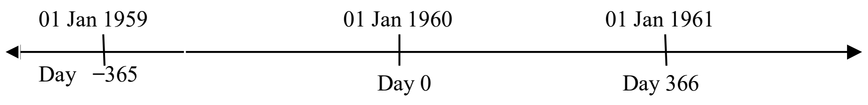
- Data Portion
  - The data portion of a SAS data set is a rectangular table of character and/or numeric data values. The row of the table represents information of an observation while the columns of the table represent the variables.
  - Types of variables

Type	
character	• Contain any values including numbers, special characters and blanks. • Character values are stored with a length of 1 to 32,767 bytes. • One byte equals one character. • Character constant is represented using quotes in the assignment statement. E.g. Gender='M';
numeric	• 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 • Numeric constant is represented a sequence of numbers with or without '.' in the assignment statement. E.g. price=10;

  - SAS date values
    - are numeric
    - are represented with a quote together with 'd' at the end for date constant in the assignment statement. E.g. Hire_date='01JAN1959'd;
    - where the reference date is 01JAN1960 and is stored as 0.
    - of date after 01JAN1960 is stored as number of days of the date starting from the reference date (positive numbers).
    - of date before 01JAN1960 is stored as number of days of the date counting back from the reference date (negative numbers).
    - Examples:

Date	Value stored	Display
01JAN1960*	0	01/01/1960
01JAN1959	-365	01/01/1959
01JAN1961	366	01/01/1961

*1960 is a leap year



  - Missing data
    - A value must exist for every variable for each observation. Missing values are valid in SAS data set.

Type	Display in data set
character	A blank
numeric	A period, i.e. '.'

- One blank record is added to data sets DATA1.Electricity and is listed as follows.

```
                                                     Heating_
                              Electricity_           Degree_     Average_
 Obs     Time_Period           Consumed       Cost     Days     Temperature
  48    Year 8: Nov/Dec          2327       229.05    1053          .
  49                               .           .        .           .
```

2.3 Creating a SAS data set from in-stream data

Complete syntax

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

DATA output-SAS-data-set;
       LENGTH variables <$>length;
       INPUT variable-list;
       DATALINES;
        a block of data-values

                ;
RUN;
```

This semicolon is required to stand alone by the end of data lines in version older than 9.4!!!
In version 9.4: It can be placed by the end of the last data line or omitted when the data are followed by a RUN statement or another step.

- LENGTH statement specifies the number of bytes for storing a variable. “$” is added right after a variable of character type;
- INPUT statement specifies the variable name to be input for the data set.
- DATALINES statement is used before the data line. Every line between DATALINES statement and “;” stores one observation. When “;” is a part of data, replace DATALINES by DATALINES4.

We reconstruct a new data set called DATA1.SUB_ELECTRICITY from the first 2 and the last 2 observations values of DATA1.ELECTRICITY directly.

```
*p2ex3.sas (version 3);
LIBNAME Data1 “d:\SAS_datasets”;

DATA SUBSET_Electricity;
       LENGTH Time_period $17;
       INPUT Time_period & $ @24 Electricity_consumed cost
             heating_degree_days average_temperature;
       DATALINES;
      YEAR 1: Jan/Feb   3637 2975.33 2226 29
      YEAR 1: March/Apr 2888 230.08 1616 37
      YEAR 8: Sept/Oct 2710 329.72 228 .
      YEAR 8: Nov/Dec   2327 229.05 1053 .
       ;
RUN;
```

Ex. 2.3: Use procedure PRINT to list the data set SUBSET_Electricity and answer the following questions:
(a) How large in term of bytes is the variable “Time_period”?
(b) Use help menu to look at the INPUT statement. Make a guess on the meaning of ‘&’ and ‘@24’?

What is the missing value for the numeric variable “average_temperature”?

2.4 Reading a SAS data set

- Complete syntax

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

DATA output-SAS-data-set;
       SET input-SAS-data-set;
       WHERE where-expression;  }
       KEEP variable-list;      }  Subsetting a data set
       DROP variable-list;      }
       LABEL variable = 'label'    }
             variable = 'label'    }
                    ⁞              }  Labelling and formatting
             variable = 'label';   }
       FORMAT variable(s) format;  }
RUN;
```

  - SET statement specifies the source SAS data set and all observations and all variables of the source data set are used.
  - WHERE statement specifies the criteria of data to be input into the *output-data-set*.
  - KEEP statement specifies the variables to be written to the *output-data-set*.
  - DROP statement specifies the variable(s) to be omitted in the *output-data-set*.
  - Note that the sequence of using KEEP and DROP statement may generate different results. It will be discussed in the section of Program Data Vector (PDV).
  - LABEL statement specifies a label for variables
  - FORMAT statement specifies the format to be stored. E.g. 4 decimal places.

- Subsetting a data set
  - Subset observations by using the WHERE statement.
  - Subset variables by using KEEP statement or DROP statement.

- Adding permanent attribute to the data set
  - LABEL statement
    - A label can have as many as 256 characters.
    - Any number of variables can be associated with labels in a single LABEL statement.
    - Using a LABEL statement in a DATA step **permanently** associates labels with variables by storing the label in the descriptor portion of the SAS data set.
  - FORMAT statement
    - A format is an instruction that SAS uses to write data values.
    - Using a FORMAT statement in a DATA step permanently associates formats with variables by storing the format in the descriptor portion of the SAS data set.

2.5 WHERE statement

- Syntax

```
WHERE where-expression;
```

  - The where-expression is a sequence of operands and operators that form a set of instructions that define a condition for selecting observations.
  - Operands include constants and variables.
    - A constant operand is a fixed value:
      (a) A character operand requires single quotation marks, e.g. ‘Male’.
      (b) A numeric operand does not require quotation marks, e.g. 5000.
    - A variable operand must be a variable coming from an input data set, e.g. where Gender=’Male’, where Salary > 5000.

- Operators are symbols that request a comparison, arithmetic calculation, or logical operation.
- Comparison Operators:

Symbol	Mnemonic	Definition	Examples of *where-expression*
=	EQ	Equal to	a=1, a EQ 1
^= ¬= ~=	NE	Not equal to	a^=1, a NE 1
>	GT	Greater than	a>1, a GT 1
<	LT	Little than	a<1, a LT 1
>=	GE	Greater than or equal to	a>=1, a GE 1
<=	LE	Little than or equal to	a<=1, a LE 1
	IN	Equal to one of a list	IN (‘A’,’B’), IN (‘AU’), IN(‘A’ ‘B’), IN (5,6,7), IN (5:7)

- Arithmetic Operators:

Symbol	Definition	Priority	Example
**	Exponentiation	I	S=a**b;
-	Negative prefix	I	X=-a;
*	Multiplication	II	P=a*b;
/	Division	II	Ratio=a/b;
+	Addition	III	Y=a+b;
-	Subtraction	III	X=a-b;

- Logical Operators:

Symbol	Mnemonic	Definition	Example
&	AND	Logical AND	Gender ne ‘M’ and Salary >50000
|	OR	Logical OR	Gender ne ‘M’ or Salary >50000
^ ¬ ~	NOT	Logical NOT	Country not in (‘AU’ ‘US’)

- Special WHERE Operators:

Mnemonic	Symbol	Definition	Example
BETWEEN-AND		an inclusive range	WHERE salary BETWEEN 5000 AND 10000; (endpoints are inclusive!)
IS NULL		Missing value	WHERE employee_ID IS NULL;
IS MISSING		Missing value	WHERE employee_ID IS MISSING;
CONTAINS	?	Select an observations containing a character string	WHERE Job_Title CONTAINS ‘Rep’; (Case sensitive!)
LIKE		A character pattern	WHERE name LIKE ‘%N’; (‘%’ replaces any number of characters ‘_’ replaces one character!) Select the observation beginning with any number of characters and ending with an ‘N’!

Ex. 2.4: Write a SAS program, to create
(a) a temporary data set called TEMPMISS containing all observations with missing value in average temperature.
(b) a permanent data set in the location d:\SAS_datasets and with name COST_TEMP which contains the electricity cost is less than or equal to 300 and the average temperature is between 41 and 70 inclusive.
(c) a temporary data set called DATAYEAR1 containing all Year 1 observations.

The source SAS data file is "d:\SAS_datasets\electricity.sas7bdat".

```
*p2ex4.sas;
libname DATA1 'd:\sas_datasets';
data                 ; /* (a)*/
  set DATA1.electricity;
  where                        ;
run;

proc print data=          ;
run;

data                 ; /* (b) */
  set data1.             ;
  where (cost          )and (               between             );
run;

proc print data=DATA1.Cost_Temp;
run;

data               ;  /* (c) */
  set                    ;
  where Time_Period          'Year 1';
run;

proc print data=DATAYEAR1;
run;
```

2.6 SAS Formats

- SAS Format is an instruction used in FORMAT statement, how SAS write the data.

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

- General form of SAS formats:

```
<$>format<w>.<d>;
```

where

symbol	meaning
$	indicates a character format
format	names the SAS format or user-defined format
w	specifies the total format width including decimal places and special characters
d	specifies the number of decimal places in numeric formats

Note that the period '.' is always required.

- Select SAS formats

Format	SAS action
$*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
COMMAX*w.d*	writes numeric values with a period that separates every three digits and a comma 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
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

- List of examples:

Variable type	SAS Formats	Stored (in bold) and displayed value	Width of non-blank displayed value
Character		**Wednesday**	
$*w*.	$3.	Wed	3
	$4.	Wedn	4
	$10.	Wednesday (note: one space at the end)	
Numeric		**12345678.91**	
w.d	12.2	12345678.91	11
	9.2	12345679	8
z*w.d* (z represents with leading zeros)	z12.2	012345678.91	12
	z11.2	12345678.91	11
	z10.2	12345678.9	10
	z9.2	012345679	9
COMMA*w.d*	Comma13.2	12,345,678.91	13
	Comma11.2	12345678.91	11
	Comma9.2	12345679	8
	Comma7.2	1.235E7	7
COMMAX*w.d*	CommaX13.2	12.345.678,91	13
	CommaX12.2	12345678,91	11
	CommaX11.2	12345678,91	11
	CommaX9.2	12345679	9
Numeric (Currency)			
DOLLAR*w.d*	Dollar14.2	$12,345,678.91	14
	Dollar12.2	$12345678.91	12
	Dollar9.2	12345679	
EURO*w.d*	Euro14.2	€12,345,678.91	14
	Euro13.2	€12345678.91	12
	Euro12.2	€12345678.91	12
	Euro9.2	12345679	8
Numeric (Date)		**18991** (i.e. 30Dec2011)	
mmddyy*w.** (w:8-11)	mmddyy8.	12/30/11	8
	mmddyy9.	12/30/11	8
	mmddyy10.	12/30/2011	10
mmyy*w.* (*w*>=5)	mmyy5.	12M11	5
yymm*w.* (*w*>=5)	yymm5.	11M12	5

yyQ*w*.	yyQ6.	2011Q4	6
(*w*:4-6)	yyQ4.	11Q4	4
Date*w*.	Date5.	30DEC	5
(w:5-11)	Date6.	30DEC	5
	Date7.	30DEC11	7
	Date8.	30DEC11	7
	Date9.	30DEC2011	9
	Date11.	30-DEC-2011	
yymmdd*xw*.*	yymmddp10.	2011.12.30	10
(*x*:B/C/D/N/P/S)	yymmddb10.	2011 12 30	10
(*w*:6-10)	yymmddc10.	2011:12:30	10
B- blank	yymmddd10.	2011-12-30	10
C-colon	yymmddn8.	20111230 (width=10 is out of range)	8
D-dash	Yymmddn6.	111230	6
N-nothing	yymmdds10.	2011/12/30	
P- period	yymmdds8.	11/12/30	8
S –slash	yymmdds6.	111230	6
Day*w*.	Day2.	30	2
(*w* >=2)	Day6.	30	2
Month*w*.	Month2.	12	2
(*w* >=2)	Month6.	12	2
Year*w*.	Year2.	11	
(*w* >=2)	Year4.	2011	4
Nldate*w*.	Nldate20.	December 30, 2011	17
Nldatemn*w*.	Nldatemn10.	December	8
Nldatewn*w*.	Nldatewn10.	Friday	6

*Can also be applied to **ddmmyyx***w*, **mmddyyx***w*.

Ex. 2.5: The following program has a DATA step and a procedure PRINT step.
(a) Please make a guess of the meaning of INFORMAT statement.
(b) How large in terms of bytes is the size for the variable Day0?
(c) Modify the program to re-produce the results of the displayed value of the above table.
(d) Fill in the column, “Width (w) of non-blank displayed value”.

```
*p2_checkformat.sas;
data checkformat;
 length Day0 $10;
 informat Date0 DATE7.;
  input Day0 $ Number0  Date0 Money0;
  datalines;
  Wednesday 12345678.91 30Dec2011 12345678.91
    ;
run;

proc print data=checkformat;
  format Day0 $10. Date0 Date6. Number0 CommaX13.2 Money0 Euro12.2;
run;
```

2.7 User-defined Formats (PROC FORMAT)

Syntax

```
PROC FORMAT;
   VALUE format-name range1 = 'label '
                     range2 = 'label '
                      . . . ;
RUN;
```

- The *format-name* should satisfy the following criteria:
  - Less than 32 characters in SAS9
  - Must have a letter or underscore as the first character for numeric format type
  - Must have a dollar sign, ‘$’ as the first character and then followed by a letter or underscore afterwards for the character format type
  - Cannot be a SAS supplied format name, e.g. mmddyy8 or dollar8
  - Cannot be ended with a number
  - Cannot be ended with a period ‘.’ In the above VALUE statement.

- Range can be
  - single values
  - ranges of values
  - lists of values
  - other key words are listed in the table:

Type	**Symbol / Keywords**	**Meaning**	**Example(s)**
Character	‘ ‘	Missing value	`‘ ’= ‘missing’`
	-	A range of values in alphabetical order	`‘A’-‘C’=‘PASS’`
	,	Separate a set of values	`‘A’,’B’,‘C’=‘PASS’`
	Other	Other values	`Other =’the remaining’`
Numeric	.	Missing value	`.= ‘missing’`
	,	Separate a set of values	`1,2,3,5,7 =’prime number’`
	-	A range of values (both end values are inclusive)	`0-39 = ‘fail’`
	<	Greater than or Less than	`0-<40 =‘fail’` (40 is excluded) `39<-100 = ‘pass’` (39 is excluded) `4<-<7=‘special’` (4,7 are excluded)
	Other	Other non-specified values	`Other =’numerical values not specified’`
	Low	Encompasses of lowest possible value	`LOW - 1000 =’Extreme low values’`
	High	Encompasses of highest possible value	`9999 - HIGH =’Extreme high values’`

- Labels
  - can be up to 32,767 characters in length
  - are typically enclosed in quotation marks, although it is not required

Revisit of Electricity data

- A SAS program is given as follows so that the average temperature, electricity consumed and the cost are in special format:

  - The temperature of electricity data in Chapter 1 can be categorized into the following 4 groups:
    (a) Low : from 20 F to 40 F
    (b) Middle: from 41F to 55 F
    (c) High: 56 or above
    (d) Missing: No data is given

  - The electricity consumed and the cost are required to be displayed with comma and dollar sign.

```
*p2_format;
proc format;
 value temp_range  20 - < 41 ='Low'
                   41 -   55 ='Middle'
                   56 - HIGH ='High'
                          . = 'Missing';
run;

proc print data=data1.electricity;
format Electricity_Consumed Comma5. cost Dollar7.2 average_temperature
temp_range.;
run;
```

- The output are printed using PROC PRINT:

```
                                 The SAS System          16:47 Monday, June 18, 2012  22

                                                                  Heating_
                                         Electricity_             Degeree_      Average_
        Obs        Time_Period             Consumed        Cost     Days       Temperature

          1     Year 1: Jan/Feb              3,637      $295.33     2226         Low
          2     Year 1: March/Apr            2,888      $230.08     1616         Low
          3     Year 1: May/June             2,359      $213.43      479         High
          4     Year 1: July/Aug             3,704      $338.16       19         High
          5     Year 1: Sept/Oct             3,432      $299.76      184         High
          6     Year 1: Nov/Dec              2,446      $214.44     1105         Middle
          7     Year 2: Jan/Feb              4,463      $384.13     2351         Low
          8     Year 2: March/Apr            2,482      $295.82     1508         Middle
          9     Year 2: May/June             2,762      $255.85      657         Middle
         10     Year 2: July/Aug             2,288      $219.72       35         High
         11     Year 2: Sept/Oct             2,423      $256.59      308         High
         12     Year 2: Nov/Dec              2,483      $276.13     1257         Middle
         13     Year 3: Jan/Feb              3,375      $321.94     2421         Low
         14     Year 3: March/Apr            2,661      $221.11     1841         Low
         15     Year 3: May/June             2,073      $205.16      438         High
         16     Year 3: July/Aug             2,579      $251.07       15         High
         17     Year 3: Sept/Oct             2,858      $279.80      152         High
         18     Year 3: Nov/Dec              2,296      $183.84     1028         Middle
         19     Year 4: Jan/Feb              2,812      $244.93     1967         Low
         20     Year 4: March/Apr            2,433      $218.59     1627         Low
         21     Year 4: May/June             2,266      $213.09      537         High
         22     Year 4: July/Aug             3,128      $333.49       26         High
         23     Year 4: Sept/Oct             3,286      $370.35      116         Missing
         24     Year 4: Nov/Dec              2,749      $222.79     1457         Missing
         25     Year 5: Jan/Feb              3,427      $316.18      253         Missing
         26     Year 5: March/Apr              578       $77.39     1811         Missing
         27     Year 5: May/June             3,792      $385.44      632         Missing
         28     Year 5: July/Aug             3,348      $334.72       35         Missing
         29     Year 5: Sept/Oct             2,937      $330.47      215         Missing
         30     Year 5: Nov/Dec              2,774      $237.00     1300         Missing
         31     Year 6: Jan/Feb              3,016      $303.78     2435         Missing
         32     Year 6: March/Apr            2,458      $263.75     1540         Missing
         33     Year 6: May/June             2,395      $207.08      395         Missing
         34     Year 6: July/Aug             3,249      $304.83       26         Missing
         35     Year 6: Sept/Oct             3,003      $305.67      153         Missing
         36     Year 6: Nov/Dec              2,118      $197.65     1095         Missing
         37     Year 7: Jan/Feb              4,261      $470.02     2554         Missing
         38     Year 7: March/Apr            1,946      $217.36     1708         Missing
         39     Year 7: May/June             2,063      $217.08      569         Missing
         40     Year 7: July/Aug             4,081      $541.01        3         Missing
         41     Year 7: Sept/Oct             1,919      $423.17       58         Missing
         42     Year 7: Nov/Dec              2,360      $256.06     1232         Missing
         43     Year 8: Jan/Feb              2,853      $309.40     2070         Missing
         44     Year 8: March/Apr            2,174      $254.91     1620         Missing
         45     Year 8: May/June             2,370      $290.98      542         Missing
         46     Year 8: July/Aug             3,480      $370.74       29         Missing
         47     Year 8: Sept/Oct             2,710      $329.72      228         Missing
         48     Year 8: Nov/Dec              2,327      $229.05     1053         Missing
```

- Once the format is defined, it can be used in FORMAT statement in DATA steps, PROC PRINT, PROC FREQ, etc.

2.8 Summary

SAS library	- SASUSER is a permanent library - WORK is a temporary library, automatically created - Libname statement is used to define permanent library - Libref must have 8 characters or less - PROC CONTENTS DATA=libref._ALL NODS; RUN;
SAS data sets	- Name has **1** to **32** characters and begins with a **letter** or an **underscore** - Two levels name reference - Two portions: Descriptor and data - PROC CONTENTS
SAS Variables	- Name has **1** to **32** characters and begins with a **letter** or an **underscore** - Two variables: Character and numeric - Character variable values can be up to 32,767 characters long and use 1 byte(s) of storage per character - By default, numeric variables are stored in 8 bytes of storage - SAS date values are stored as numeric; The reference day is 1 Jan 1960 = Day 0 - Missing values: Blank for character variable and a period for numeric variable
Creating a data set using DATALINES	- WHERE statement is used for subsetting the observations of the data set - LABEL statement is used to give description of the variables in the data set - FORMAT statement is used to define permanent format for the data values
Format	- SAS formats on character, numeric for numbers, currency and date - User-defined format is defined in PROC FORMAT.

2.9 Quiz

1. What is the default size of a numeric variable?

   (a) 2 bytes (b) 8 bytes (c) 10 bytes (d) 16 bytes

2. What is the numeric value for the date 1 Jan 1960?

   (a) -1 (b) 0 (c) 1 (d) 010111960

3. Which of the following statement read the "Sales Date" correct?

Layout: Discount.data

Description	Column
Brand No.	1-4
Sales Date	5-12
Item Group	14-21
Discount	22-24

Input file:

```
         1         2         3
1---5----0----5----0----5----0
104012/02/17 Outdoors15%
201010/07/17 Golf      7%
100309/22/17 Shoes    10%
100309/22/17 Shirts   10%
200109/11/17 Skirts   15%
```

(a) input @1Brand_no 4. @5 Sales_dt mmddyy8. @14 Item_gp $8. @22 Discount percent3.;
(b) read @1Brand_no 4. @5 Sales_dt mmddyy8. @14 Item_gp $8. @22 Discount percent3.;
(c) input @1Brand_no 5. @6 Sales_dt date7. @14 Item_gp $8. @22 Discount percent3.;
(d) read @1Brand_no 5. @6 Sales_dt date7. @14 Item_gp $8. @22 Discount percent3.;