Unit 5: Data Visualization and Macros

Course: Statistical Data Analytics and Integration

These lecture notes cover SAS reporting, visualization, and the SAS Macro Facility. The explanations are written in simple undergraduate-level language with examples and SAS code.

5.1 Creating a Default List Report

Meaning

A **list report** displays data values in rows and columns. In SAS, the simplest way to create a default list report is by using PROC PRINT .

Syntax

```sas
PROC PRINT DATA=dataset_name;
RUN;
```

Example Dataset

```sas
DATA students;
   INPUT ID Name $ Gender $ Marks Age;
   DATALINES;
1 Arun M 78 20
2 Meena F 85 19
3 Ravi M 67 21
4 Priya F 92 20
;
RUN;
```

Default List Report

```sas
PROC PRINT DATA=students;
RUN;
```

Output Structure

Obs	ID	Name	Gender	Marks	Age
1	1	Arun	M	78	20
2	2	Meena	F	85	19

Obs	ID	Name	Gender	Marks	Age
3	3	Ravi	M	67	21
4	4	Priya	F	92	20

Important Options

```sas
PROC PRINT DATA=students NOOBS;
RUN;
```

NOOBS removes the observation number column.

```sas
PROC PRINT DATA=students LABEL;
  LABEL Marks = "Exam Marks"
        Age = "Student Age";
RUN;
```

LABEL displays variable labels instead of variable names.

5.2 Windowing / Non-Windowing Mode

Windowing Mode

Windowing mode refers to an interactive SAS environment where users can work through windows such as:

- Program Editor
- Log Window
- Output Window
- Explorer Window
- Results Window

This is common in traditional SAS Display Manager.

Non-Windowing Mode

Non-windowing mode means SAS runs programs without interactive windows. The program is submitted through command line, batch mode, or server execution.

Comparison

Feature	Windowing Mode	Non-Windowing Mode
Interaction	Interactive	Batch or command-based
Windows available	Yes	No
Suitable for	Learning, debugging	Production jobs

Feature	Windowing Mode	Non-Windowing Mode
Output	Displayed in Output/Results window	Saved to files/logs
Example use	Classroom practice	Automated reporting

Example: Batch Execution

```bash
sas analysis.sas
```

This runs the SAS program without opening interactive windows.

5.3 DEFINE Statement

Meaning

The DEFINE statement is used in PROC REPORT to describe how each variable should appear in the report.

Syntax

```sas
DEFINE variable / usage-option attributes;
```

Common Usage Options

Option	Meaning
DISPLAY	Shows variable values as they are
ORDER	Orders rows based on values
GROUP	Groups observations with same value
ACROSS	Creates columns from values
ANALYSIS	Performs statistics such as sum or mean
COMPUTED	Creates a calculated variable

Example

```sas
PROC REPORT DATA=students NOWD;
  COLUMN Gender Name Marks;
  DEFINE Gender / GROUP "Gender";
  DEFINE Name / DISPLAY "Student Name";
  DEFINE Marks / ANALYSIS MEAN "Average Marks";
RUN;
```

Explanation

- Gender is used to group students.
- Name is displayed as a normal variable.
- Marks is analyzed using the mean statistic.

5.4 Defining Variables

In SAS reporting, variables can be defined according to their role.

Types of Variable Usage in PROC REPORT

Usage	Description	Example
Display	Prints values directly	Name
Order	Sorts report rows	Age
Group	Combines rows with same value	Gender
Analysis	Computes statistics	Marks
Computed	New calculated variable	Grade
Across	Creates cross-tab columns	Department

Example

```sas
PROC REPORT DATA=students NOWD;
  COLUMN Gender Marks;
  DEFINE Gender / GROUP;
  DEFINE Marks / ANALYSIS MEAN;
RUN;
```

Output Concept

Gender	Mean Marks
F	88.5
M	72.5

5.5 COLUMN Statement

Meaning

The COLUMN statement in PROC REPORT tells SAS which variables to include in the report and their order.

Syntax

```sas
COLUMN variable1 variable2 variable3;
```

Example

```sas
PROC REPORT DATA=students NOWD;
  COLUMN Name Gender Marks;
  DEFINE Name / DISPLAY;
  DEFINE Gender / DISPLAY;
  DEFINE Marks / DISPLAY;
RUN;
```

Output

Name	Gender	Marks
Arun	M	78
Meena	F	85
Ravi	M	67
Priya	F	92

Using Computed Columns

```sas
PROC REPORT DATA=students NOWD;
  COLUMN Name Marks Grade;
  DEFINE Name / DISPLAY;
  DEFINE Marks / DISPLAY;
  DEFINE Grade / COMPUTED;

  COMPUTE Grade / CHARACTER LENGTH=10;
    IF Marks >= 80 THEN Grade = "Good";
    ELSE Grade = "Average";
  ENDCOMP;
RUN;
```

5.6 Selecting Observations

Meaning

Selecting observations means filtering rows from a dataset.

In SAS, this can be done using:

- WHERE statement
- IF statement

- Dataset options

Using WHERE

```sas
PROC PRINT DATA=students;
  WHERE Marks >= 80;
RUN;
```

This prints only students with marks greater than or equal to 80.

Using IF in DATA Step

```sas
DATA high_marks;
  SET students;
  IF Marks >= 80;
RUN;
```

Difference Between WHERE and IF

Feature	WHERE	IF
Used in	DATA step and PROC step	DATA step only
Timing	Before data enters program data vector	After data enters program data vector
Efficiency	Usually faster	Less efficient for large data
Can use computed variables?	Usually no	Yes

Worked Example

```sas
DATA passed;
  SET students;
  IF Marks >= 50;
RUN;

PROC PRINT DATA=passed;
RUN;
```

This creates a new dataset called passed .

5.7 Group Processing Variables

Meaning

Group processing means analyzing data by categories or groups.

Examples:

- Average marks by gender
- Sales by region
- Patients by treatment group

Using BY Statement

Before using BY , data must be sorted.

```sas
PROC SORT DATA=students OUT=sorted_students;
  BY Gender;
RUN;

PROC PRINT DATA=sorted_students;
  BY Gender;
RUN;
```

Using CLASS Statement

Some procedures allow grouping without sorting.

```sas
PROC MEANS DATA=students;
  CLASS Gender;
  VAR Marks;
RUN;
```

Using GROUP in PROC REPORT

```sas
PROC REPORT DATA=students NOWD;
  COLUMN Gender Marks;
  DEFINE Gender / GROUP;
  DEFINE Marks / ANALYSIS MEAN;
RUN;
```

Example Output

Gender	Mean Marks
F	88.5
M	72.5

5.8 Computing Variables

Meaning

A computed variable is a new variable created from existing variables.

Example Dataset

```sas
DATA sales;
   INPUT Product $ Quantity Price;
   DATALINES;
A 10 50
B 5 100
C 8 75
;
RUN;
```

Formula

Total Sales:

$$\text{Total Sales} = \text{Quantity} \times \text{Price}$$

SAS Code

```sas
DATA sales_calc;
   SET sales;
   TotalSales = Quantity * Price;
RUN;

PROC PRINT DATA=sales_calc;
RUN;
```

Output

Product	Quantity	Price	TotalSales
A	10	50	500
B	5	100	500
C	8	75	600

Computing Variables in PROC REPORT

```sas
PROC REPORT DATA=sales NOWD;
   COLUMN Product Quantity Price TotalSales;
   DEFINE Product / DISPLAY;
   DEFINE Quantity / DISPLAY;
   DEFINE Price / DISPLAY;
```

```
  DEFINE TotalSales / COMPUTED;

  COMPUTE TotalSales;
    TotalSales = Quantity * Price;
  ENDCOMP;
RUN;
```

5.9 Bar Chart

Meaning

A **bar chart** is used to compare values across categories.

Examples:

- Number of students by gender
- Sales by region
- Average marks by department

Diagram

```
Marks by Student

Priya | ████████████████████ 92
Meena | ██████████████████   85
Arun  | ████████████████     78
Ravi  | ████████████         67
```

SAS Code Using PROC SGPLOT

```sas
PROC SGPLOT DATA=students;
  VBAR Name / RESPONSE=Marks;
  TITLE "Marks by Student";
RUN;
```

Horizontal Bar Chart

```sas
PROC SGPLOT DATA=students;
  HBAR Name / RESPONSE=Marks;
  TITLE "Marks by Student";
RUN;
```

Grouped Bar Chart

```sas
```

```
PROC SGPLOT DATA=students;
  VBAR Gender / RESPONSE=Marks STAT=MEAN;
  TITLE "Average Marks by Gender";
RUN;
```

Key Options

Option	Meaning
VBAR	Vertical bar chart
HBAR	Horizontal bar chart
RESPONSE=	Numeric variable shown by bars
STAT=MEAN	Shows average
GROUP=	Groups bars by another variable

5.10 Box Plot

Meaning

A **box plot** shows the distribution of numerical data.

It displays:

- Minimum
- First quartile
- Median
- Third quartile
- Maximum
- Outliers

Important Formulas

Range:

$$\text{Range} = \text{Maximum} - \text{Minimum}$$

Interquartile Range:

$$IQR = Q_3 - Q_1$$

Outlier rule:

$$\text{Lower Limit} = Q_1 - 1.5(IQR)$$

$$\text{Upper Limit} = Q_3 + 1.5(IQR)$$

Diagram

```
Minimum    Q1      Median     Q3     Maximum
  |--------[----------|----------]---------|
```

SAS Code

```sas
PROC SGPLOT DATA=students;
  VBOX Marks / CATEGORY=Gender;
  TITLE "Box Plot of Marks by Gender";
RUN;
```

Interpretation

A box plot helps answer:

- Which group has a higher median?
- Which group has more variation?
- Are there any outliers?

5.11 Line Plot

Meaning

A **line plot** shows change over time or ordered values.

Examples:

- Monthly sales
- Temperature over days
- Stock price over time

Example Dataset

```sas
DATA monthly_sales;
  INPUT Month $ Sales;
  DATALINES;
Jan 100
Feb 120
Mar 150
Apr 130
May 180
;
RUN;
```

SAS Code

```sas
PROC SGPLOT DATA=monthly_sales;
  SERIES X=Month Y=Sales / MARKERS;
```

```
  TITLE "Monthly Sales Trend";
RUN;
```

Diagram

```
Sales
180 |                    *
150 |           *
130 |               *
120 |      *
100 | *
    ---------------------------
     Jan Feb Mar Apr May
```

Interpretation

The line plot shows whether sales are increasing, decreasing, or fluctuating.

5.12 Spaghetti Plot

Meaning

A **spaghetti plot** shows many individual lines on the same graph.

It is often used in longitudinal studies, where each subject is measured repeatedly over time.

Example

Suppose patients are measured at Week 0, Week 4, and Week 8.

```
sas
DATA patient_response;
  INPUT PatientID Week Response;
  DATALINES;
1 0 50
1 4 55
1 8 60
2 0 48
2 4 52
2 8 58
3 0 60
3 4 62
3 8 65
;
RUN;
```

SAS Code

```
sas
```

```
PROC SGPLOT DATA=patient_response;
  SERIES X=Week Y=Response / GROUP=PatientID MARKERS;
  TITLE "Spaghetti Plot of Patient Response Over Time";
RUN;
```

Diagram

```
Response
65 |                    / Patient 3
60 |           /----------/
55 |      /----/ Patient 1
50 | ----/ Patient 2
   ----------------------------
     0       4       8 Weeks
```

Interpretation

Each line represents one subject. The plot helps show individual patterns and differences.

5.13 Survival Plot

Meaning

A **survival plot** shows the probability that a subject survives beyond a certain time.

It is common in:

- Medical studies
- Reliability analysis
- Clinical trials
- Time-to-event data analysis

Key Terms

Term	Meaning
Survival time	Time until event occurs
Event	Death, failure, relapse, etc.
Censoring	Event not observed during study
Survival probability	Probability of surviving beyond a time

Survival Function

$$S(t) = P(T > t)$$

Where:

- $S(t)$ = survival probability at time t
- T = time until event

Example Dataset

```sas
DATA survival_data;
  INPUT Patient Time Status Group $;
  DATALINES;
1 5 1 A
2 8 0 A
3 12 1 A
4 6 1 B
5 10 0 B
6 15 1 B
;
RUN;
```

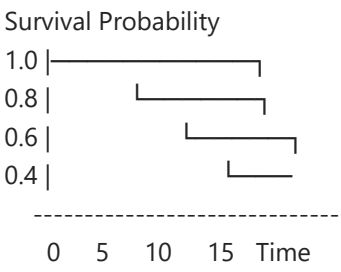
Here:

- Status = 1 means event occurred.
- Status = 0 means censored.

SAS Code

```sas
PROC LIFETEST DATA=survival_data PLOTS=SURVIVAL;
  TIME Time*Status(0);
  STRATA Group;
  TITLE "Survival Plot by Group";
RUN;
```

Diagram



Interpretation

A survival curve that stays higher indicates better survival.

5.14 Waterfall Plot

Meaning

A **waterfall plot** displays ordered changes from a baseline value. It is commonly used in clinical trials to show percentage change in tumor size for each patient.

Formula

Percentage change from baseline:

$$\text{Percentage Change} = \frac{\text{Follow-up Value} - \text{Baseline Value}}{\text{Baseline Value}} \times 100$$

Example Dataset

```sas
DATA tumor;
  INPUT Patient $ Baseline Followup;
  PercentChange = ((Followup - Baseline) / Baseline) * 100;
  DATALINES;
P1 100 80
P2 120 90
P3 90 95
P4 110 70
P5 100 130
;
RUN;
```

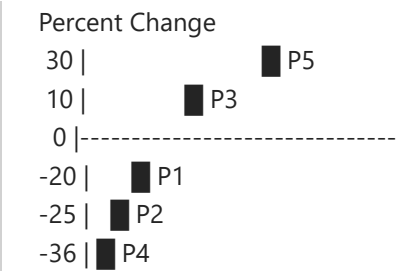
Sort Data

```sas
PROC SORT DATA=tumor OUT=tumor_sorted;
  BY PercentChange;
RUN;
```

SAS Code

```sas
PROC SGPLOT DATA=tumor_sorted;
  VBAR Patient / RESPONSE=PercentChange;
  REFLINE 0 / AXIS=Y LINEATTRS=(COLOR=BLACK);
  TITLE "Waterfall Plot of Tumor Percentage Change";
RUN;
```

Diagram



Interpretation

- Negative bars show reduction.

- Positive bars show increase.
- A larger negative value may indicate better treatment response.

5.15 Forest Plot

Meaning

A **forest plot** displays estimated effects and confidence intervals for different groups or studies.

It is common in:

- Meta-analysis
- Clinical research
- Subgroup analysis

Key Components

Component	Meaning
Point estimate	Estimated effect, such as odds ratio
Confidence interval	Range of likely values
Reference line	No-effect line
Group label	Study or subgroup name

Confidence Interval Formula

For an estimate:

$$CI = \text{Estimate} \pm Z \times SE$$

For a 95% confidence interval:

$$CI = \text{Estimate} \pm 1.96 \times SE$$

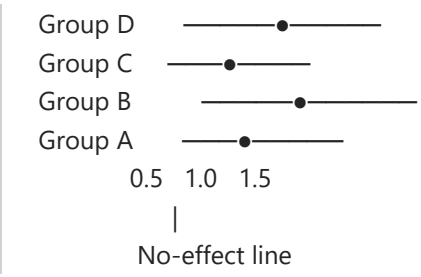
Example Dataset

```sas
DATA forest;
  INPUT Group $ Estimate Lower Upper;
  DATALINES;
A 0.80 0.60 1.05
B 1.20 0.90 1.60
C 0.65 0.45 0.95
D 1.10 0.80 1.40
;
RUN;
```

SAS Code

```sas
PROC SGPLOT DATA=forest;
  SCATTER Y=Group X=Estimate / XERRORLOWER=Lower XERRORUPPER=Upper;
  REFLINE 1 / AXIS=X LINEATTRS=(PATTERN=SHORTDASH);
  XAXIS LABEL="Effect Estimate";
  YAXIS LABEL="Group";
  TITLE "Forest Plot";
RUN;
```

Diagram



Interpretation

- If the confidence interval crosses the reference line, the effect may not be statistically significant.
- If the confidence interval is entirely on one side, the effect may be significant.

5.16 Introduction to Macro Facility

Meaning

The **SAS Macro Facility** is a tool that helps automate SAS programs.

It allows users to:

- Reuse code
- Reduce repetition
- Create dynamic programs
- Use macro variables
- Write flexible reports

Why Use Macros?

Without macro:

```sas
PROC PRINT DATA=sales_jan;
RUN;

PROC PRINT DATA=sales_feb;
RUN;
```

```
PROC PRINT DATA=sales_mar;
RUN;
```

With macro:

```sas
%MACRO printdata(ds);
  PROC PRINT DATA=&ds;
  RUN;
%MEND;

%printdata(sales_jan)
%printdata(sales_feb)
%printdata(sales_mar)
```

Important Macro Concepts

Concept	Meaning
Macro variable	Stores text values
Macro program	Reusable block of SAS code
Macro processor	Converts macro code into SAS code
Macro trigger	% or & symbols

5.17 SAS Programs and Macro Processing

Normal SAS Processing

A regular SAS program is compiled and executed by SAS directly.

Example:

```sas
PROC PRINT DATA=students;
RUN;
```

Macro Processing

Macro statements are processed before normal SAS code is compiled.

Flow Diagram

```
Macro Code
   ↓
Macro Processor
   ↓
Generated SAS Code
   ↓
```

> SAS Compiler
> ↓
> Execution
> ↓
> Output

Example

```sas
%LET dataset = students;

PROC PRINT DATA=&dataset;
RUN;
```

Before SAS executes the program, the macro processor replaces `&dataset` with `students`.

Generated code:

```sas
PROC PRINT DATA=students;
RUN;
```

5.18 Macro Variables

Meaning

A **macro variable** stores text that can be substituted into SAS code.

Creating Macro Variables

Using %LET

```sas
%LET name = students;
```

Using the macro variable:

```sas
PROC PRINT DATA=&name;
RUN;
```

Macro Variable Reference

A macro variable is referenced using `&`.

```sas
&name
```

Example

```sas
%LET var = Marks;

PROC MEANS DATA=students;
  VAR &var;
RUN;
```

Generated SAS code:

```sas
PROC MEANS DATA=students;
  VAR Marks;
RUN;
```

Automatic Macro Variables

SAS also provides automatic macro variables.

Examples:

Macro Variable	Meaning
&SYSDATE	Current date
&SYSTIME	Current time
&SYSUSERID	User ID
&SYSVER	SAS version

Example:

```sas
%PUT Today is &SYSDATE;
```

5.19 Macro Processing

Macro Triggers

SAS identifies macro code using two triggers:

Trigger	Use
%	Macro statements and macro calls
&	Macro variable references

Example

```sas
%LET course = Statistics;
%PUT Course name is &course;
```

Output in log:

> Course name is Statistics

Macro Program

```sas
%MACRO report;
  PROC PRINT DATA=students;
  RUN;
%MEND report;

%report
```

Macro with Parameters

```sas
%MACRO report(ds, var);
  PROC MEANS DATA=&ds;
    VAR &var;
  RUN;
%MEND report;

%report(students, Marks)
```

Generated code:

```sas
PROC MEANS DATA=students;
  VAR Marks;
RUN;
```

5.20 Scopes of Macro Variables

Meaning

Scope means where a macro variable is available.

There are two main scopes:

1. Global scope
2. Local scope

Global Macro Variable

A global macro variable can be used anywhere in the SAS session.

```sas
%LET dataset = students;

PROC PRINT DATA=&dataset;
RUN;
```

Here, `&dataset` is global.

Local Macro Variable

A local macro variable exists only inside a macro.

```sas
%MACRO test;
  %LET message = Hello;
  %PUT &message;
%MEND;

%test
```

Here, `message` is local to the macro.

Example Showing Scope

```sas
%LET x = Global;

%MACRO demo;
  %LET y = Local;
  %PUT x is &x;
  %PUT y is &y;
%MEND;

%demo

%PUT Outside macro x is &x;
%PUT Outside macro y is &y;
```

The last statement may produce a warning because `y` is not available outside the macro.

Scope Diagram

```
Global Symbol Table
-------------------
x = Global

Macro Local Symbol Table
------------------------
y = Local
```

5.21 Macro Expressions

Meaning

Macro expressions are expressions evaluated by the macro processor.

They are often used in:

- %IF
- %DO
- %EVAL
- %SYSEVALF

Integer Expression Using %EVAL

```sas
%LET a = 10;
%LET b = 20;
%LET sum = %EVAL(&a + &b);

%PUT Sum is &sum;
```

Output:

> Sum is 30

Decimal Expression Using %SYSEVALF

```sas
%LET x = 10.5;
%LET y = 2;
%LET result = %SYSEVALF(&x / &y);

%PUT Result is &result;
```

Output:

> Result is 5.25

Conditional Macro Expression

```sas
%LET marks = 85;

%MACRO grade;
  %IF &marks >= 80 %THEN %PUT Grade is A;
  %ELSE %PUT Grade is B;
%MEND;
```

```
%grade
```

Looping Expression

```sas
%MACRO repeat;
  %DO i = 1 %TO 5;
    %PUT This is repetition &i;
  %END;
%MEND;

%repeat
```

5.22 Macro Quoting

Meaning

Macro quoting hides special characters from the macro processor.

Special characters include:

> , ; + - * / = < > & %

Sometimes these characters may confuse the macro processor.

Common Macro Quoting Functions

Function	Use
%STR()	Masks special characters during compilation
%NRSTR()	Masks special characters including & and %
%QUOTE()	Masks resolved values during execution
%NRQUOTE()	Masks resolved values including & and %
%BQUOTE()	Masks unmatched quotes and parentheses
%SUPERQ()	Safely references macro variable values

Example Using %STR

```sas
%LET text = %STR(Statistics; Data Analytics);

%PUT &text;
```

The semicolon is treated as text, not as the end of the statement.

Example Using %NRSTR

```sas
%LET example = %NRSTR(This uses &name and %macro);
%PUT &example;
```

Here, &name and %macro are treated as plain text.

5.23 Interfaces with Macro Facility

Meaning

The Macro Facility can interact with different SAS components.

Main Interfaces

Interface	Purpose
%LET	Creates macro variables
CALL SYMPUT	Creates macro variables from DATA step
CALL SYMPUTX	Improved version of CALL SYMPUT
SYMGET	Gets macro variable value inside DATA step
PROC SQL INTO	Stores SQL result into macro variable
%SYSFUNC	Uses SAS functions in macro code

Example: CALL SYMPUTX

```sas
DATA _NULL_;
  SET students;
  IF Name = "Priya" THEN CALL SYMPUTX("priya_marks", Marks);
RUN;

%PUT Priya's marks are &priya_marks;
```

Example: PROC SQL INTO

```sas
PROC SQL;
  SELECT MEAN(Marks)
  INTO :avg_marks
  FROM students;
QUIT;

%PUT Average marks = &avg_marks;
```

Example: %SYSFUNC

```sas
%LET today = %SYSFUNC(TODAY(), DATE9.);
%PUT Today is &today;
```

5.24 Storing and Reusing Macros

Meaning

Macros can be stored and reused in later SAS programs.

This improves productivity and consistency.

Method 1: Store Macro in Same Program

```sas
%MACRO print_report(ds);
  PROC PRINT DATA=&ds;
  RUN;
%MEND print_report;

%print_report(students)
```

Method 2: Store Macro in External File

Create a file called `macros.sas` :

```sas
%MACRO print_report(ds);
  PROC PRINT DATA=&ds;
  RUN;
%MEND print_report;
```

Use it in another SAS program:

```sas
%INCLUDE "C:\SAS\macros.sas";

%print_report(students)
```

Method 3: Autocall Macro Library

An autocall library allows SAS to automatically find stored macros.

```sas
OPTIONS SASAUTOS=("C:\SAS\MacroLibrary", SASAUTOS);
```

If a file named `print_report.sas` exists in the folder, SAS can call it directly:

```sas
%print_report(students)
```

Method 4: Stored Compiled Macro Facility

This stores compiled macros for faster execution.

```sas
OPTIONS MSTORED SASMSTORE=mylib;
```

```sas
LIBNAME mylib "C:\SAS\StoredMacros";

%MACRO print_report(ds) / STORE;
  PROC PRINT DATA=&ds;
  RUN;
%MEND;
```

Later use:

```sas
LIBNAME mylib "C:\SAS\StoredMacros";
OPTIONS MSTORED SASMSTORE=mylib;

%print_report(students)
```

Worked Example: Complete Report and Visualization

Problem

Create a report showing average marks by gender and draw a bar chart.

Dataset

```sas
DATA students;
  INPUT ID Name $ Gender $ Marks Age;
  DATALINES;
1 Arun M 78 20
2 Meena F 85 19
3 Ravi M 67 21
4 Priya F 92 20
5 Kumar M 74 22
6 Divya F 88 19
;
RUN;
```

Step 1: Report Using PROC REPORT

```sas
PROC REPORT DATA=students NOWD;
  COLUMN Gender Marks;
  DEFINE Gender / GROUP "Gender";
  DEFINE Marks / ANALYSIS MEAN "Average Marks";
  TITLE "Average Marks by Gender";
RUN;
```

Step 2: Bar Chart

```sas
PROC SGPLOT DATA=students;
  VBAR Gender / RESPONSE=Marks STAT=MEAN;
  TITLE "Average Marks by Gender";
RUN;
```

Step 3: Macro to Repeat Analysis

```sas
%MACRO group_mean(ds, groupvar, numvar);
  PROC REPORT DATA=&ds NOWD;
    COLUMN &groupvar &numvar;
    DEFINE &groupvar / GROUP;
    DEFINE &numvar / ANALYSIS MEAN;
    TITLE "Mean of &numvar by &groupvar";
  RUN;

  PROC SGPLOT DATA=&ds;
    VBAR &groupvar / RESPONSE=&numvar STAT=MEAN;
    TITLE "Bar Chart of Mean &numvar by &groupvar";
  RUN;
%MEND;

%group_mean(students, Gender, Marks)
```

Quick Revision Points

Reporting

- PROC PRINT creates a default list report.
- PROC REPORT creates customized reports.
- COLUMN controls the variables displayed.
- DEFINE controls how each variable is used.
- GROUP combines rows with same values.

- ANALYSIS calculates statistics such as mean, sum, minimum, and maximum.
- COMPUTED creates new variables inside a report.

Filtering

- WHERE filters observations before processing.
- IF filters observations inside a DATA step.
- WHERE can be used in both DATA and PROC steps.
- IF is used only in DATA steps.

Graphs

- Bar chart compares categories.
- Box plot shows distribution and outliers.
- Line plot shows trends over time.
- Spaghetti plot shows many individual trends.
- Survival plot shows time-to-event probabilities.
- Waterfall plot shows ordered changes from baseline.
- Forest plot shows estimates and confidence intervals.

Macro Facility

- Macro variables are referenced using &.
- Macro statements begin with %.
- %LET creates macro variables.
- %MACRO and %MEND define a macro.
- %EVAL handles integer calculations.
- %SYSEVALF handles decimal calculations.
- Macro quoting prevents special characters from being misinterpreted.
- CALL SYMPUTX and PROC SQL INTO create macro variables dynamically.
- Macros can be reused using %INCLUDE, autocall libraries, or stored compiled macros.

Important SAS Procedures in Unit 5

Procedure	Purpose
PROC PRINT	Simple list report
PROC REPORT	Customized report
PROC SORT	Sort data
PROC MEANS	Descriptive statistics

Procedure	Purpose
PROC SGPLOT	Statistical graphs
PROC LIFETEST	Survival analysis
PROC SQL	SQL queries and macro variable creation

Common SAS Code Templates

Default Report

```sas
PROC PRINT DATA=dataset;
RUN;
```

Customized Report

```sas
PROC REPORT DATA=dataset NOWD;
  COLUMN var1 var2 var3;
  DEFINE var1 / DISPLAY;
  DEFINE var2 / GROUP;
  DEFINE var3 / ANALYSIS MEAN;
RUN;
```

Bar Chart

```sas
PROC SGPLOT DATA=dataset;
  VBAR category / RESPONSE=numeric_variable STAT=MEAN;
RUN;
```

Box Plot

```sas
PROC SGPLOT DATA=dataset;
  VBOX numeric_variable / CATEGORY=group_variable;
RUN;
```

Line Plot

```sas
PROC SGPLOT DATA=dataset;
  SERIES X=time_variable Y=response_variable / MARKERS;
RUN;
```

Macro Template

```sas
%MACRO macro_name(parameter1, parameter2);
  SAS statements using &parameter1 and &parameter2;
%MEND macro_name;

%macro_name(value1, value2)
```

---

10 Exam Questions with Answers

Question 1

Define a default list report in SAS. Write the SAS procedure used to create it.

Answer

A default list report displays data in rows and columns without much customization. In SAS, it is created using `PROC PRINT`.

```sas
PROC PRINT DATA=students;
RUN;
```

Question 2

What is the difference between windowing and non-windowing mode in SAS?

Answer

Windowing mode is an interactive SAS environment with windows such as Program Editor, Log, and Output. Non-windowing mode runs SAS programs in batch or command mode without interactive windows.

Windowing Mode	Non-Windowing Mode
Interactive	Non-interactive
Has windows	No windows
Useful for learning/debugging	Useful for production jobs

Question 3

What is the purpose of the `DEFINE` statement in `PROC REPORT`?

Answer

The `DEFINE` statement specifies how a variable should be used and displayed in a report.

Example:

```sas
PROC REPORT DATA=students NOWD;
  COLUMN Gender Marks;
  DEFINE Gender / GROUP;
  DEFINE Marks / ANALYSIS MEAN;
RUN;
```

Here, Gender groups the data and Marks is summarized using mean.

Question 4

Differentiate between WHERE and IF in SAS.

Answer

WHERE	IF
Can be used in DATA and PROC steps	Used only in DATA step
Filters before data enters processing	Filters during DATA step execution
More efficient for large datasets	Useful with computed variables

Example:

```sas
PROC PRINT DATA=students;
  WHERE Marks > 80;
RUN;
```

```sas
DATA high;
  SET students;
  IF Marks > 80;
RUN;
```

Question 5

Write SAS code to create a bar chart showing average marks by gender.

Answer

```sas
PROC SGPLOT DATA=students;
  VBAR Gender / RESPONSE=Marks STAT=MEAN;
```

```
   TITLE "Average Marks by Gender";
RUN;
```

Question 6

What is a box plot? Mention any three statistics shown in a box plot.

Answer

A box plot is a graph that shows the distribution of numerical data.

It displays:

- Minimum
- First quartile
- Median
- Third quartile
- Maximum
- Outliers

The interquartile range is:

$$IQR = Q_3 - Q_1$$

Question 7

What is a spaghetti plot? Where is it commonly used?

Answer

A spaghetti plot shows multiple individual lines on the same graph. Each line usually represents one subject or unit.

It is commonly used in longitudinal studies, clinical trials, and repeated-measure data.

Example:

```sas
PROC SGPLOT DATA=patient_response;
   SERIES X=Week Y=Response / GROUP=PatientID MARKERS;
RUN;
```

Question 8

Define a SAS macro variable. How is it created and referenced?

Answer

A SAS macro variable stores text that can be substituted into SAS code.

It is created using %LET and referenced using &.

Example:

```sas
%LET dataset = students;

PROC PRINT DATA=&dataset;
RUN;
```

Here, &dataset is replaced by students.

Question 9

Explain global and local macro variables.

Answer

A global macro variable is available throughout the SAS session. A local macro variable is available only inside the macro where it is created.

Example:

```sas
%LET x = Global;

%MACRO demo;
   %LET y = Local;
   %PUT &x;
   %PUT &y;
%MEND;

%demo
```

Here, x is global and y is local.

Question 10

Write a SAS macro to print any dataset.

Answer

```sas
%MACRO printdata(ds);
   PROC PRINT DATA=&ds;
      TITLE "Report for &ds";
   RUN;
%MEND printdata;
```

```
%printdata(students)
```

This macro takes a dataset name as input and prints that dataset.

Summary

Unit 5 focuses on two major areas:

1. **Data visualization and reporting**
    - Creating reports using PROC PRINT and PROC REPORT
    - Using COLUMN , DEFINE , grouping, filtering, and computed variables
    - Creating graphs such as bar charts, box plots, line plots, survival plots, waterfall plots, and forest plots
2. **SAS Macro Facility**
    - Creating macro variables
    - Writing reusable macro programs
    - Understanding macro processing
    - Using macro expressions and macro quoting
    - Storing and reusing macros

A good way to master this unit is to practice writing small SAS programs for each graph and then convert repetitive code into macros.