1. CREATE DATA:

```
/* Creating the dataset */
DATA Employee_Data;
    /* Defining the variables and inputting data */
    INPUT Name $ Gender $ Employee_ID Age Height Weight Salary;

    /* DATALINES or CARDS is used to input the data */
    DATALINES;
    Alice     Female  1001  28  165  60  55000
    Bob       Male    1002  34  175  78  72000
    Charlie   Male    1003  29  180  85  68000
    David     Male    1004  40  170  82  80000
    Eva       Female  1005  35  160  55  63000
    Frank     Male    1006  25  185  90  50000
    Grace     Female  1007  30  168  58  71000
    Hannah    Female  1008  27  158  52  60000
    Ian       Male    1009  45  178  75  95000
    Judy      Female  1010  32  162  60  64000
    Karl      Male    1011  38  180  88  85000
    Laura     Female  1012  31  165  59  72000
    Michael   Male    1013  29  172  76  68000
    Nina      Female  1014  26  160  55  53000
    Oscar     Male    1015  36  182  80  78000
    Paula     Female  1016  33  170  62  69000
    Quincy    Male    1017  41  177  77  88000
    Rachel    Female  1018  28  166  58  61000
    Steve     Male    1019  37  180  85  83000
    Tina      Female  1020  29  162  57  67000
    Uma       Female  1021  30  168  59  72000
    Victor    Male    1022  35  182  82  74000
    Wendy     Female  1023  26  160  54  58000
    Xander    Male    1024  44  176  83  92000
    Yara      Female  1025  28  164  60  62000
    Zack      Male    1026  39  178  80  81000
    Amber     Female  1027  34  167  61  70000
    Ben       Male    1028  32  174  79  76000
    Chloe     Female  1029  30  162  56  65000
    Dan       Male    1030  42  181  84  89000
    ;
RUN;

/* Display the data */

PROC PRINT DATA=Employee_Data;
RUN;

/* Using PROC REPORT Alignment and coluring */
PROC REPORT DATA=Employee_Data NOWD;
    COLUMN Name Gender Employee_ID Age Height Weight Salary;
    DEFINE Name / DISPLAY "Name" LEFT;
```

```
    DEFINE Gender / DISPLAY "Gender" CENTER
STYLE(COLUMN)={BACKGROUND=lightgreen};
    DEFINE Employee_ID / DISPLAY "Employee ID" CENTER
STYLE(COLUMN)={BACKGROUND=lightcyan};
    DEFINE Age / DISPLAY "Age" CENTER;
    DEFINE Height / DISPLAY "Height (cm)" RIGHT;
    DEFINE Weight / DISPLAY "Weight (kg)" RIGHT;
    DEFINE Salary / DISPLAY "Salary ($)" FORMAT=dollar12.2 RIGHT
STYLE(COLUMN)={BACKGROUND=lightcoral};
RUN;
```

NOWD: This option stands for "No Windowing." It tells SAS to run the report in non-interactive mode,
Dollar Format:
specifies the width of the output field.
Default 6
Range 2–32

specifies the number of digits to the right of the decimal point in the numeric value. This argument is optional.
Range 0–31
Requirementmust be less than *Whole n*

2. IMPORT DATA FROM EXCEL:

```
/* Generated Code (IMPORT) */
/* Source File: Employee.xlsx */
/* Source Path: /home/u61999975/sasuser.v94 */
/* Code generated on: 8/26/24, 11:56 AM */

%web_drop_table(WORK.IMPORT);


FILENAME REFFILE '/home/u61999975/sasuser.v94/Employee.xlsx';

PROC IMPORT DATAFILE=REFFILE
     DBMS=XLSX
     OUT=WORK.IMPORT;
     GETNAMES=YES;
RUN;

PROC CONTENTS DATA=WORK.IMPORT;
RUN;


%web_open_table(WORK.IMPORT);
PROC PRINT DATA=WORK.IMPORT;
RUN;
```

3. DESCRIPTIVE STATISTICS

```
/* Creating the dataset */
DATA Employee_Data;
```

```
    /* Defining the variables and inputting data */
    INPUT Name $ Gender $ Employee_ID Age Height Weight Salary;

    /* DATALINES or CARDS is used to input the data */
    DATALINES;
    Alice     Female  1001  28  165  60  55000
    Bob       Male    1002  34  175  78  72000
    Charlie   Male    1003  29  180  85  68000
    David     Male    1004  40  170  82  80000
    Eva       Female  1005  35  160  55  63000
    Frank     Male    1006  25  185  90  50000
    Grace     Female  1007  30  168  58  71000
    Hannah    Female  1008  27  158  52  60000
    Ian       Male    1009  45  178  75  95000
    Judy      Female  1010  32  162  60  64000
    Karl      Male    1011  38  180  88  85000
    Laura     Female  1012  31  165  59  72000
    Michael   Male    1013  29  172  76  68000
    Nina      Female  1014  26  160  55  53000
    Oscar     Male    1015  36  182  80  78000
    Paula     Female  1016  33  170  62  69000
    Quincy    Male    1017  41  177  77  88000
    Rachel    Female  1018  28  166  58  61000
    Steve     Male    1019  37  180  85  83000
    Tina      Female  1020  29  162  57  67000
    Uma       Female  1021  30  168  59  72000
    Victor    Male    1022  35  182  82  74000
    Wendy     Female  1023  26  160  54  58000
    Xander    Male    1024  44  176  83  92000
    Yara      Female  1025  28  164  60  62000
    Zack      Male    1026  39  178  80  81000
    Amber     Female  1027  34  167  61  70000
    Ben       Male    1028  32  174  79  76000
    Chloe     Female  1029  30  162  56  65000
    Dan       Male    1030  42  181  84  89000
    ;
RUN;
PROC PRINT DATA=Employee_Data;
RUN;
/* Calculate the mean of a variable */
proc means data=Employee_Data mean;
   var Age Height Weight Salary;
run;
proc means data=Employee_Data mean maxdec=2;
   var Age Height Weight Salary;
run;
/* Calculate the median of a variable */
proc means data=Employee_Data median maxdec=1;
   var Age Height Weight Salary;
run;
```

```
/* Step 1: Calculate frequencies using PROC FREQ */
proc freq data=Employee_Data noprint;
   tables Height / out=freq_count (keep=Height count);
run;

/* Step 2: Sort the frequency table in descending order by count */
proc sort data=freq_count;
   by descending count;
run;

/* Step 3: Print the first observation, which is the mode */
proc print data=freq_count(obs=1);
   title "Mode of Variable Height";
run;
```

noprint: Suppresses the default output of the frequency table to the results window. This is useful when you do not need to display the entire frequency table and are only interested in saving the output to a dataset.
/ out=freq_count: The / indicates that options follow for the tables statement. The out= option creates a new output dataset named freq_count that contains the frequency table results.
(keep=Height count): This specifies that the output dataset freq_count should only include the Height variable and the count variable (which contains the frequency counts for each height value). The keep= option limits the variables in the output dataset to only those specified.

```
/* Calculate variance and standard deviation using PROC MEANS */
proc means data=Employee_Data std var;
    var Age Height Weight Salary;
run;
/* Calculate variance, standard deviation, and quartile deviation using
PROC UNIVARIATE */
proc univariate data=Employee_Data;
    var Age Height Weight Salary;
    output out=stats
        std=std
        var=variance
        qrange=iqr; /* Interquartile range (Q3 - Q1) */
run;
```

qrange: Represents the interquartile range (IQR), which is the difference between the third quartile (Q3) and the first quartile (Q1). The IQR is a measure of statistical dispersion, indicating the spread of the middle 50% of the data.

```
/* Calculate Quartile Deviation */
data stats;
    quartile_deviation = iqr / 2;
run;

proc print data=stats;
    var std variance iqr quartile_deviation;
run;
```

4. DIAGRAMATIC REPRESENTATION:

```
/* Example Data: Create a dataset with Month, Revenue, Expenses, Profit,
and Category */
data sales_data;
    input Month $ Revenue Expenses Profit Category $;
    datalines;
Jan 45000 30000 15000 A
Feb 48000 31000 17000 B
Mar 47000 32000 15000 A
Apr 50000 33000 17000 B
May 52000 34000 18000 A
Jun 51000 35000 16000 B
Jul 53000 36000 17000 A
Aug 54000 37000 17000 B
Sep 52000 38000 14000 A
Oct 55000 39000 16000 B
Nov 56000 40000 16000 A
Dec 58000 41000 17000 B
;
run;
PROC PRINT DATA=Sales_Data;
RUN;
/* Create an Enhanced Line Chart with Data Point Numbers */
proc sgplot data=sales_data;
    series x=Month y=Revenue / markers lineattrs=(color=blue thickness=2);
    scatter x=Month y=Revenue / datalabel=Revenue
markerattrs=(symbol=circlefilled size=10 color=red);
    title "Monthly Revenue Trend with Data Points";
    xaxis label="Month" discreteorder=data;
    yaxis label="Revenue" grid;
run;
```

proc gchart data=sashelp.cars;: This statement starts the GCHART procedure to create graphs
proc sgplot: The SGPLOT procedure is a powerful SAS procedure for creating single-cell plots such as scatter plots, series plots, bar charts, histograms, etc. It is highly customizable and widely used for data visualization.
/ markers: Adds markers at each data point along the series line, making each individual point visible.
lineattrs=(color=blue thickness=2): Customizes the line's appearance:
color=blue: Sets the color of the line to blue.
thickness=2: Sets the thickness of the line to 2 units, making it thicker than the default line width.
/ datalabel=Revenue: Adds labels to each data point showing the actual Revenue value next to the point.
markerattrs=(symbol=circlefilled size=10 color=red): Customizes the appearance of the markers:
symbol=circlefilled: Specifies that each data point should be marked with a filled circle.
size=10: Sets the size of the markers to 10 units, making them larger and more visible.
color=red: Sets the color of the markers to red.
discreteorder=data: Ensures that the x-axis categories (months) are displayed in the order they appear in the data. This is particularly useful for months or other ordered categorical data to ensure they are not sorted alphabetically or numerically.

```
/* Create a Histogram */
proc sgplot data=sales_data;
    histogram revenue / transparency=0.5;
```

```
    density revenue / type=kernel;
    title "Distribution of Revenue";
run;
```

/ transparency=0.5: This option controls the transparency level of the histogram bars:
transparency=0.5: Sets the transparency of the histogram bars to 50%, making them semi-transparent. This allows other graphical elements (such as the density plot) to be seen more clearly when overlaid on the histogram.
/ type=kernel: Specifies the type of density plot to be used:
type=kernel: Uses a kernel density estimator to create a smooth curve that represents the distribution of the Revenue variable. The kernel density plot provides a more refined view of the distribution than the histogram alone, allowing for a better understanding of the data's underlying distribution pattern.

```
/* Create a Scatter Plot */
proc sgplot data=sales_data;
    scatter x=revenue y=profit / group=category
markerattrs=(symbol=circlefilled);
    title "Scatter Plot of Profit vs. Revenue";
run;
```

/: This is an option delimiter that allows you to specify various options for the scatter statement.
group=category: This option groups the data points by the variable category. This means that each unique value of the category variable will be represented by a different color or marker symbol in the scatter plot. Grouping by category allows you to distinguish between different categories of data visually, making it easier to identify patterns or relationships that vary between categories.

Circle: A simple, hollow circle.
Symbol Name: circle
Filled Circle: A solid, filled circle.
Symbol Name: circlefilled
Square: A simple, hollow square.
Symbol Name: square
Filled Square: A solid, filled square.
Symbol Name: squarefilled
Triangle: A simple, hollow triangle pointing upward.
Symbol Name: triangle
Filled Triangle: A solid, filled triangle pointing upward.
Symbol Name: trianglefilled
Triangle Down: A simple, hollow triangle pointing downward.
Symbol Name: triangledown
Filled Triangle Down: A solid, filled triangle pointing downward.
Symbol Name: triangledownfilled
Plus Sign: A simple plus sign (+).
Symbol Name: plus
Cross: An x shape.
Symbol Name: x
Star: A hollow star.
Symbol Name: star
Filled Star: A solid, filled star.
Symbol Name: starfilled

Diamond: A simple, hollow diamond.
Symbol Name: diamond
Filled Diamond: A solid, filled diamond.
Symbol Name: diamondfilled
Hash: A # symbol.
Symbol Name: hash
Dot: A very small filled circle, smaller than circlefilled.
Symbol Name: dot

```
/* Create a Box Plot */
proc sgplot data=sales_data;
    vbox expenses / category=category;
    title "Box Plot of Expenses by Category";
run;

/* Create a Heat Map */
proc sgplot data=sales_data;
    heatmap x=revenue y=expenses;
    title "Heat Map of Revenue vs. Expenses";
run;
/* Create a Bubble Plot */
proc sgplot data=sales_data;
    bubble x=revenue y=profit size=expenses / transparency=0.5;
    title "Bubble Plot of Revenue, Profit, and Expenses";
run;

/* Create a Series Plot with Confidence Limits */
data timeseries;
    input time y lcl ucl;
    datalines;
1 10 8 12
2 15 13 17
3 20 18 22
4 18 16 20
5 25 23 27
;
run;

/* Create a Series Plot with Confidence Limits */
data sales_with_limits;
    set sales_data;
    Lower = Profit - 2000; /* Example lower limit */
    Upper = Profit + 2000; /* Example upper limit */
run;

proc sgplot data=sales_with_limits;
    band x=month lower=lower upper=upper / transparency=0.5;
    series x=month y=profit / lineattrs=(thickness=2);
    title "Profit Over Time with Confidence Limits";
    xaxis label="Month" discreteorder=data;
run;
```

```
/* Set the graphics environment */
goptions reset=all cback=white border htitle=12pt htext=10pt;

title1 "Types of Vehicles Produced Worldwide";

proc gchart data=sashelp.cars;
   pie type / other=0
              midpoints="Truck" "SUV" "Sedan" "Wagon" "Sports" "Hybrid"
              value=none
              percent=arrow
              slice=arrow
              noheading
              plabel=(font='Albany AMT/bold' h=1.3 color=depk);
run;
quit;
```

reset=all: Resets all graphics options to their default settings.
cback=white: Sets the background color of the graph to white.
border: Adds a border around the graph area.
htitle=12pt: Sets the height of the title text to 12 points.
htext=10pt: Sets the height of the text (such as axis labels and footnotes) to 10 points.

proc gchart data=sashelp.cars;: This statement starts the GCHART procedure to create graphs
other=0: Excludes any categories not listed in midpoints from being grouped into an "Other" category in the pie chart.
midpoints="Truck" "SUV" "Sedan" "Wagon" "Sports" "Hybrid": Specifies the categories of the type variable to be included in the pie chart. Only the types "Truck," "SUV," "Sedan," "Wagon," "Sports," and "Hybrid" will be shown.
value=none: Suppresses the display of raw data values (such as counts or frequencies) on the pie slices.
percent=arrow: Displays the percentage of each category with an arrow pointing from the percentage label to the corresponding pie slice.
slice=arrow: Adds arrows to connect each slice to its label.
noheading: Removes the default heading from the pie chart.
plabel=(font='Albany AMT/bold' h=1.3 color=depk);: This customizes the appearance of the pie slice labels:
font='Albany AMT/bold': Sets the font of the pie slice labels to 'Albany AMT' in bold.
h=1.3: Sets the height (size) of the label text to 1.3 times the default size.
color=depk: Sets the color of the pie slice labels to DEPK, which is a deep pink color.
quit;:

This statement ends the PROC GCHART procedure. While quit is not always necessary, it is a good practice to use it to explicitly close the procedure and free up system resources.

```
/* Set the graphics environment */
goptions reset=all border cback=white
         htitle=12pt htext=8pt;

 /* Create data set TOTALS */
data totals;
   length Dept $ 7 Site $ 8;
```

```
   input Dept Site Quarter Sales;
   datalines;
Parts   Sydney  1 4043.97
Parts   Atlanta 1 6225.26
Parts   Paris   1 3543.97
Repairs Sydney  1 5592.82
Repairs Atlanta 1 9210.21
Repairs Paris   1 8591.98
Tools   Sydney  1 1775.74
Tools   Atlanta 1 2424.19
Tools   Paris   1 5914.25
Parts   Sydney  2 3723.44
Parts   Atlanta 2 11595.07
Parts   Paris   2 9558.29
Repairs Sydney  2 5505.31
Repairs Atlanta 2 4589.59
Repairs Paris   2 7538.56
Tools   Sydney  2 2945.17
Tools   Atlanta 2 1903.99
Tools   Paris   2 7868.34
Parts   Sydney  3 8437.96
Parts   Atlanta 3 6847.91
Parts   Paris   3 6789.85
Repairs Sydney  3 4426.46
Repairs Atlanta 3 5011.66
Repairs Paris   3 6510.38
Tools   Sydney  3 3767.10
Tools   Atlanta 3 3048.52
Tools   Paris   3 9017.96
Parts   Sydney  4 6065.57
Parts   Atlanta 4 9388.51
Parts   Paris   4 8509.08
Repairs Sydney  4 3012.99
Repairs Atlanta 4 2088.30
Repairs Paris   4 5530.37
Tools   Sydney  4 3817.36
Tools   Atlanta 4 4354.18
Tools   Paris   4 6511.70
;
run;
PROC PRINT DATA=totals;
RUN;

 /* Define the title for the chart */
title1 'Site Sales By Dept (Details)';

 /* Generate detail pie chart */
proc gchart data=totals;
   pie site / sumvar=sales
              detail=dept
              detail_percent=best
```

```
              detail_value=none
              detail_slice=best
              legend;
run;
quit;
```

sumvar=sales: This option tells SAS to use the sales variable to determine the size of each slice of the pie chart. The size of each slice is proportional to the sum of sales for each site category.
4. detail=dept
detail=dept: This option adds further detail to each slice of the pie by displaying the breakdown of the dept variable (department) within each site. This allows for a more granular view of the data within each slice, showing how sales are distributed across different departments within each site.
5. detail_percent=best
detail_percent=best: This option specifies that the percentage representation of each dept within each site slice should be displayed using the "best" format. The "best" format ensures that the most readable and concise percentage values are used, typically rounding percentages to a few decimal places.
6. detail_value=none
detail_value=none: This option specifies that the actual sales values for each dept within each site slice will not be displayed on the pie chart. This setting helps to declutter the chart if only percentage values are of interest.
7. detail_slice=best
detail_slice=best: This option determines how the slices representing each dept within the site pie slices are best displayed to optimize readability and visual appeal. The "best" option usually ensures that slices are displayed in a visually effective manner, often determined automatically by SAS.
8. legend
legend: This option adds a legend to the pie chart, which typically describes the categories represented by each slice and the detailed subcategories (if any). The legend helps viewers understand the color or pattern associated with each slice of the pie chart and each detailed subcategory.

SAMPLING METHODS:

```
data random_numbers;
    /* Set the random seed for reproducibility */
    call streaminit(12345);

    /* Generate random numbers */
    do i = 1 to 100; /* Number of random numbers to generate */
uniform_num = rand("Uniform");    /* Uniform(0,1) */
normal_num = rand("Normal", 0, 1); /* Normal(0,1) */
binomial_num = rand("Binomial", 0.5, 50); /* Binomial with p=0.5, n=10 */
output;
end;
run;

proc print data=random_numbers (obs=100); /* Print first 10 observations
*/
run;

DATA Employee_Data;
    /* Defining the variables and inputting data */
    INPUT Name $ Gender $ Employee_ID Age Height Weight Salary;
```

```
    /* DATALINES or CARDS is used to input the data */
    DATALINES;
    Alice     Female  1001  28  165  60  55000
    Bob       Male    1002  34  175  78  72000
    Charlie   Male    1003  29  180  85  68000
    David     Male    1004  40  170  82  80000
    Eva       Female  1005  35  160  55  63000
    Frank     Male    1006  25  185  90  50000
    Grace     Female  1007  30  168  58  71000
    Hannah    Female  1008  27  158  52  60000
    Ian       Male    1009  45  178  75  95000
    Judy      Female  1010  32  162  60  64000
    Karl      Male    1011  38  180  88  85000
    Laura     Female  1012  31  165  59  72000
    Michael   Male    1013  29  172  76  68000
    Nina      Female  1014  26  160  55  53000
    Oscar     Male    1015  36  182  80  78000
    Paula     Female  1016  33  170  62  69000
    Quincy    Male    1017  41  177  77  88000
    Rachel    Female  1018  28  166  58  61000
    Steve     Male    1019  37  180  85  83000
    Tina      Female  1020  29  162  57  67000
    Uma       Female  1021  30  168  59  72000
    Victor    Male    1022  35  182  82  74000
    Wendy     Female  1023  26  160  54  58000
    Xander    Male    1024  44  176  83  92000
    Yara      Female  1025  28  164  60  62000
    Zack      Male    1026  39  178  80  81000
    Amber     Female  1027  34  167  61  70000
    Ben       Male    1028  32  174  79  76000
    Chloe     Female  1029  30  162  56  65000
    Dan       Male    1030  42  181  84  89000
    ;
RUN;

/* Display the data */
PROC PRINT DATA=Employee_Data;
RUN;
/* Simple Random Sampling Without Replacement */
proc surveyselect data=Employee_Data
                  method=srs /* Simple random sampling */
                  sampsize=10 /* Number of random numbers to generate */
                  seed=12345  /* Seed for reproducibility */
                  out=random_sample; /* Output dataset */
run;

proc print data=random_sample;
run;

proc print data=sashelp.class;
```

```
run;


proc surveyselect data=sashelp.class /* Using built-in dataset */
		method=srs /* Simple random sampling */
		sampsize=10 /* Number of random numbers to generate */
		seed=12345  /* Seed for reproducibility */
		out=random_sample; /* Output dataset */
run;

proc print data=random_sample;
run;
/* Simple Random Sampling With Replacement */
proc surveyselect data=sashelp.class
		method=urs	/* Unrestricted Random Sampling (with replacement) */
		sampsize=5
		seed=12345
		out=urs_sample;
run;

proc print data=urs_sample;
run;
proc surveyselect data=sashelp.class
		method=sys	/* Systematic Sampling */
		sampsize=5
		seed=12345
		out=sys_sample;
run;

proc print data=sys_sample;
run;
```