Assignment 1 - ANLY 608

Stone Leiker

# sas

**Question 1**

```
/*Create a library to store the health dataset*/
libname health 'C:\Users\leiker-s\Desktop\msa608_2024\Assignment 1\';
run;
/* Import the dataset and save in the health library as health1*/
PROC IMPORT DATAFILE='C:\Users\leiker-s\Desktop\msa608_2024\Assignment 1\Heath
Data.csv'
	OUT=health.health1
	DBMS=csv
	REPLACE;
	GETNAMES=YES;
RUN;
/* Create a dataset called t_health*/
data t_health;
	set health.health1;
run;
/* Calculate mean and standard deviation of all variables*/
proc means data=t_health mean std;
run;
/* Print the value of RBC for subject 11005*/
proc print data=t_health;
	where subj=110055;
	var rbc;
run;
/* Get summary statistics for RBC, Hcrit, WBC, and MCHC for each hospital*/
proc means data=t_health mean std min max;
	class hosp;
	var rbc hcrit wbc mchc;
run;
/* Output mean and median for each Hospital*/
proc means data=t_health mean median;
	class hosp;
	var rbc wbc hcrit;
	output out=Hospitals_mean_median mean=mean_rbc mean_wbc mean_hcrit
```

```
                                                                median=median_rbc
median_wbc median_hcrit;
run;
/* Create a histogram for WBC*/
proc sgplot data=t_health;
        histogram wbc;
        title "Histogram of WBC";
run;
/* Create a boxplot for WBC*/
proc sgplot data=t_health;
        vbox wbc;
        title "Boxplot of WBC";
run;
/* Create a scatterplot where X-axis is RBC and Y-axis is WBC*/
proc sgplot data=t_health;
        scatter x=rbc y=wbc;
        title "Scatterplot of RBC vs WBC";
run;
/*Create separate datasets for subject 210006, 310032, and 410010*/
data s210006;
        set t_health;
        where subj=210006;
run;

data s310032;
        set t_health;
        where subj=310032;
run;

data s410010;
        set t_health;
        where subj=410010;
Run;
```

**Question 2**

```
/*Create a library to store the grades dataset*/
libname grades 'C:\Users\leiker-s\Desktop\msa608_2024\Assignment 1';
run;
/* Import the dataset and save in the grades library as student_grades*/
```

```
PROC IMPORT DATAFILE= 'C:\Users\leiker-s\Desktop\msa608_2024\Assignment 1\Student
Grades.csv'
	OUT=grades.student_grades
	DBMS=csv
	REPLACE;
	GETNAMES=YES;
RUN;
/*Sort the data by student ID and grade*/
Proc sort data=grades.student_grades out=sorted_grades;
	by idno grade;
run;
/* Create a dataset with the lowest grade for each student*/
data lowest_grade;
	set sorted_grades;
	by idno;
	if first.idno then output;
run;
/* Print the lowest grades and the corresponding semesters*/
proc print data=lowest_grade;
	title 'Lowest Grades and Semester for each Student';
run;
/* Transpose the dataset from long to wide format*/
proc transpose data=grades.student_grades out=wide_grades;
	by idno;
	id gtype;
	var grade;
run;
/* Print the transposed student grades*/
proc print data=wide_grades;
	title 'Transposed Student Grades';
Run;
```

**Question 3**

```
/*Question 3 - Convert Celsius to Fahrenheit*/

libname m6082024 "'C:\Users\leiker-s\Desktop\msa608_2024";run;

PROC IMPORT DATAFILE = "'C:\Users\leiker-s\Desktop\msa608_2024\Assignment
1\Atmosphere.csv"
OUT=Atmosphere
```

```
DBMS=csv
REPLACE;
SHEET="Sheet1";
GETNAMES=YES;
RUN;

data fahrenheit2 (drop = i);
   set Work.Atmosphere;
        array amonths [12] jan feb mar apr may jun jul aug sep oct nov dec;
        do i = 1 to 12;
        amonths[i] = (1.8 * amonths[i]) + 32;
        end;
Run;
proc print data=fahrenheit2;
        title 'Celsius to Fahrenheit Conversion (fahrenheit2 dataset)';
run;
```

**Question 4**

```
/*Question 4.1*/

libname m6082024 "'C:\Users\leiker-s\Desktop\msa608_2024";run

PROC IMPORT DATAFILE = "'C:\Users\leiker-s\Desktop\msa608_2024\Assignment 1\d1.csv"
OUT=d1
DBMS=csv
REPLACE;
SHEET="Sheet1";
GETNAMES=YES;
RUN;

PROC IMPORT DATAFILE = "'C:\Users\leiker-s\Desktop\msa608_2024\Assignment 1\d2.csv"
OUT=d2
DBMS=csv
REPLACE;
SHEET="Sheet1";
GETNAMES=YES;
RUN;

DATA dmerged;
```

```
        MERGE d1 d2;
        BY id;
RUN;

/*Question 4.2*/
proc stdize data=dmerged out=demerged1 reponly method=mean;
   var id age visit outcome;
run;

proc print data=demerged1;
run;
```

# R

**Question 5**

```
#Q5: Create a variable named “weights of babies” with starting value=7
#and ending value=40, spaced by 1.5.

weights_of_babies <- seq(7, 40, 1.5)

#Find the mean and standard deviation of weights of babies.

mean(weights_of_babies)
sd(weights_of_babies)

#Create a histogram for weights of babies.
hist(weights_of_babies, main="Histogram of Weights of Babies", xlab="Weight",
col="lightblue")
```

**Question 6**

```
#Question 6

sales <- c(NA, "TP", 4, 6.7, 'c', NA, 12)

#Find NA in the variable:

is.na(sales)
```

#Identify NA's in Vector

```
which(is.na(sales))
```

#Identify total number of NA's:

```
sum(is.na(sales))
```

**Question 7**

```
# Create the dataframe
dataframe <- data.frame(
  Name = c("Bell", "Dia", "KKN", "Nia"),
  Physics = c(98, 87, 91, 94),
  Chemistry = c(NA, 84, 93, 87),
  Mathematics = c(91, 86, NA, NA)
)

# Function to replace missing values with the mean of the respective column
replace_na_with_mean <- function(column) {
  # Replace NA with the mean of the column
  column[is.na(column)] <- mean(column, na.rm = TRUE)
  return(column)
}

# Apply the function to all columns (except the 'Name' column)
dataframe$Physics <- replace_na_with_mean(dataframe$Physics)
dataframe$Chemistry <- replace_na_with_mean(dataframe$Chemistry)
dataframe$Mathematics <- replace_na_with_mean(dataframe$Mathematics)

# Print the dataframe with missing values replaced by the column mean
print(dataframe)

# Alternatively, to replace missing values with the mean of the respective row:
# Function to replace NA in each row by the row mean
replace_na_with_row_mean <- function(row) {
  row[is.na(row)] <- mean(row, na.rm = TRUE)
  return(row)
}

# Apply the function to each row (excluding the 'Name' column)
```

```
dataframe[, -1] <- t(apply(dataframe[, -1], 1, replace_na_with_row_mean))

# Print the dataframe with missing values replaced by row mean
print(dataframe)
```

**Question 8**

```
# Import the Titanic dataset
t1 <- read.csv('titanic.csv')

# Find the total number of survivors and print the result
num_survived <- sum(t1$Survived == 1, na.rm = TRUE)
print(paste("Total survived:", num_survived))

# Find the total number of deaths and print the result

num_dead <- sum(t1$Survived == 0, na.rm = TRUE)
print(paste("Total dead:", num_dead))

# Count the number of males and print the result
num_males <- sum(t1$Sex == "male", na.rm = TRUE)
print(paste("Number of males:", num_males))

# Count the number of females and print the result
num_females <- sum(t1$Sex == "female", na.rm = TRUE)
print(paste("Number of females:", num_females))

# Find the maximum age and print the result
max_age <- max(t1$Age, na.rm = TRUE)
print(paste("Maximum age:", max_age))

# Find the medium age and print the result
median_age <- median(t1$Age, na.rm = TRUE)
print(paste("Median age:", median_age))

# Count the number of missing age values and print the result
missing_ages <- sum(is.na(t1$Age))
```

```
print(paste("Number of missing age values:", missing_ages))

# Drop all rows with missing values
t2 <- na.omit(t1)

# Create a pie chart showing the proportion of survivors vs deaths
pie(table(t1$Survived), labels = c("Dead", "Survived"), main="Survivors vs Deaths",
col=c("red", "green"))

# Create a histogram of the survived people based on gender
survived_data <- t1[t1$Survived == 1, ]
hist(as.numeric(survived_data$Sex == "male"), breaks=2, main="Survived by Gender",
xlab="Gender (0=Female, 1=Male)", col="blue")
```

**Question 9**

```
# Create the data frame with player stats: player names, positions, points, and assists
df <- data.frame(
  player = c('A', 'B', 'C', 'D', 'E', 'F'),
  position = c('R1', 'R2', 'R3', 'R4', 'R5', NA),
  points = c(102, 105, 219, 322, 232, NA),
  assists = c(405, 407, 527, 412, 211, NA))

# Create a quality variable based on point
# high if points > 215, medium if points > 120, otherwise low
df$quality <- ifelse(df$points > 215, "high",
                                    ifelse(df$points > 120, "medium", "low"))

# Create a performance variable based on points and assists
# Great if points > 215 and assists > 10, good if points > 215 and assists > 5, otherwise average
df$performance <- ifelse(df$points > 215 & df$assists > 10, "great",
                         ifelse(df$points > 215 & df$assists > 5, "good", "average"))

# Print the updated data frame with the new variables
print(df)
```