Python Lab Assignment 9

# Aims

1. Find the missing values and give a report of it.

2. Fetch all the rows having Salary >80000

3. Fetch all the rows having Team = Legal and Finance

4. Find the missing values in Senior Management and replace it with NONE

5. Find the employees of having Bonus % more than 12 and reduce it with 2.

# Code

## Function definitions

import numpy as np

import pandas as pd

*# Converting dataframe to a 2D numpy array...*

data = pd.read\_csv('employees2.csv')

*#================================================*

*# GENERAL FUNCTIONS*

*# 1.1. Print rows by list of indices, for a given data frame*

**def** printRowsByIndicesAndDataframe(indices, df):

*# Using fancy indexing to retrieve*

*# all rows where this column element is null...*

isnull\_rows = df.loc[indices]

*# Printing the above list of locator objects*

*# as a Pandas data frame...*

print(pd.DataFrame(isnull\_rows))

*# 1.2.Print rows by list of indices, for the main data frame*

**def** printRowsByIndices(indices):

printRowsByIndicesAndDataframe(indices, data)

*# 2. Print a list using a separator...*

**def** printListWithSeparator(l, sep):

if len(l) == 0:

print("Empty list.")

return

elif len(l) == 1:

print(l[0])

return

else:

for i in l[0:-2]:

*# This loop iterates until the 2nd last index.*

*# This is so the last index can be printed without*

*# a comma at the end (i.e. this is just for formatting).*

print(i, end = sep)

*# Last element...*

print(l[-1])

*#================================================*

*# GETTING MISSING VALUE INDICES FOR COLUMN NAME*

*# --VITAL CODE--*

**def** getMissingValueIndices(colname):

*# Series whose elements correspond to*

*# whether the respective value in the*

*# actual data column in NaN or not...*

isnull\_indicator = data[colname].isnull()

*# Using fancy indexing to get*

*# the NaN elements in the data column...*

isnull\_elements = data[colname][isnull\_indicator]

*# Returning the indices of these NaN elements...*

isnull\_indices = isnull\_elements.keys()

return isnull\_indices

*# SEE COMPLETE MISSING VALUE ROWS*

*# --OPTIONAL CODE--*

*# (Printing the rows where this column's value is NaN)*

**def** printMissingValueRows(colname):

*# Retrieving the row indices of*

*# all NaN elements in the column...*

isnull\_indices = getMissingValueIndices(colname)

if len(isnull\_indices) == 0:

print("No rows with missing values in this column.")

else:

print("Rows with missing values for", colname)

printRowsByIndices(isnull\_indices)

**def** report\_missingValues():

*# --VITAL CODE SECTION--*

print("VIEW ROW INDICES OF MISSING VALUES BY COLUMN\n")

*# To cycle through each column...*

for colname in data.keys():

*# Retrieving the row indices of*

*# all NaN elements in the column...*

isnull\_indices = getMissingValueIndices(colname)

*# Printing the above indices...*

print("\_"\*12)

print("Column name:", colname)

print("-"\*6)

if len(isnull\_indices) == 0:

print("No rows with missing values in this column.")

else:

print("Rows with missing values in this column...",)

printListWithSeparator(isnull\_indices, ", ")

*# --OPTIONAL CODE SECTION--*

print("\n" + "="\*24)

if(input("View complete rows? (y/n) ") == "y"):

print("\n" + "="\*24)

print("VIEW COMPLETE MISSING VALUE ROWS BY COLUMN NAME")

*# A loop that enables you to repeatedly see*

*# the complete rows with the missing values*

*# in the particular column...*

while True:

print("\n(Enter a hyphen '-' to exit this loop)")

colname = input("Input column name: ")

if colname == "-": break

try: printMissingValueRows(colname)

except: print("Invalid column name!")

*#================================================*

*# ROWS WITH SALARY ABOVE A CERTAIN AMOUNT*

**def** report\_rowsAboveSalaryAmount(amt):

*# --VITAL CODE--*

isaboveamt\_indicator = data['Salary'] > amt

isaboveamt\_elements = data['Salary'][isaboveamt\_indicator]

isaboveamt\_indices = isaboveamt\_elements.keys()

if len(isaboveamt\_indices) == 0:

print("No rows with salary above this amount.")

else:

print("Rows with salary greater than this amount...")

printListWithSeparator(isaboveamt\_indices, ", ")

*# --OPTIONAL CODE (within else block)--*

print("\n" + "="\*24)

if(input("View complete rows? (y/n) ") == "y"):

print("\n" + "="\*24)

printRowsByIndices(isaboveamt\_indices)

*#================================================*

*# ROWS WITH GIVEN TEAM NAMES*

**def** report\_rowsWithGivenTeamNames(teamnames):

hasteamnames\_indicator = (data['Team'] == teamnames[0])

for t in teamnames[1:]:

hasteamnames\_indicator = (hasteamnames\_indicator | (data['Team'] == t))

"""

LOGIC DEMONSTRATION - REPEATING OR OPERATIONS

\_\_\_

To demonstrate the above code's logic more clearly,

run the following code...

a = np.array([1, 2, 3, 4, 5, 6, 7, 8, 9])

indicator = (a == 1) # Initial value

for i in [2, 4, 7]:

indicator = indicator | (a == i)

print(indicator)

\_\_\_

This is a substitute for multiple 'or' operations as in...

indicator = (a == 1 | a == 2 | a == 4 | a == 7)

"""

hasteamnames\_elements = data['Team'][hasteamnames\_indicator]

hasteamnames\_indices = hasteamnames\_elements.keys()

if len(hasteamnames\_indices) == 0:

print("No rows with team names 'Legal' and 'Finance'.")

else:

print("Rows with team names 'Legal' and 'Finance'...")

printListWithSeparator(hasteamnames\_indices, ", ")

*# --OPTIONAL CODE (within else block)--*

print("\n" + "="\*24)

if(input("View complete rows? (y/n) ") == "y"):

printRowsByIndices(hasteamnames\_indices)

*#================================================*

*# REPLACE MISSING VALUES IN COLUMN*

**def** report\_replaceMissingValues(colname, newvalue):

*# Getting the indices with missing values in the column...*

*# (This is only for displaying purposes)*

isnull\_indices = data[colname][data[colname].isnull()].keys()

*# Filling the NaN values in the column...*

modified = data[colname].fillna(newvalue)

*# Equating this Series object*

*# to the Series object of a duplicate dataset*

*# (to preserve original data)...*

tmp = data.copy(deep = True)

"""

NOTES ON COPY

\_\_\_

deep = True makes an entirely new data frame

with duplicate values, and changes in original

don't affect it.

\_\_\_

deep = False makes a new identifier for the data frame,

and changes in original will be reflected too,

since it is the same object, but with a different identifier.

"""

tmp[colname] = modified.copy(deep =True)

*# Printing the modified data frame...*

printRowsByIndicesAndDataframe(isnull\_indices, tmp)

*#================================================*

*# GETTING BONUS % ABOVE A THRESHOLD*

*# (AND REPLACING IT WITH GIVEN VALUE)*

**def** report\_reduceBonus(threshold, reduceby):

abovethreshold\_elements = data['Bonus %'][data['Bonus %'] > threshold]

abovethreshold\_values = abovethreshold\_elements.values

abovethreshold\_indices = abovethreshold\_elements.keys()

*# (The indices are used for display purposes only)*

*# Showing initial values (that are to be replaced)...*

print("INITIAL VALUES")

printRowsByIndices(abovethreshold\_indices)

*# Creating a new column where the desired values are replaced...*

modified = data['Bonus %'].replace(abovethreshold\_values, abovethreshold\_values - reduceby)

*# Equating this Series object*

*# to the Series object of a duplicate dataset*

*# (to preserve original data)...*

tmp = data.copy(deep = True) *#*

tmp['Bonus %'] = modified.copy(deep = True)

*# Showing final values (that have been replaced)...*

print("-"\*12)

print("AFTER REPLACEMENT")

printRowsByIndicesAndDataframe(abovethreshold\_indices, tmp)

## Main program

from functionDefinitions import \*

questions = """

QUESTIONS

1. Find the missing values and give a report of it.

2. Fetch all the rows having Salary >80000

3. Fetch all the rows having Team = Legal and Finance

4. Find the missing values in Senior Management and replace it with NONE

5. Find the employees of having Bonus % more than 12 and reduce it with 2."""

map = {

*# Format: Question number: (Function name, Arguments)*

'?': (print, [questions]),

'1': (report\_missingValues, []),

'2': (report\_rowsAboveSalaryAmount, [8000]),

'3': (report\_rowsWithGivenTeamNames, [['Legal', 'Finance']]),

'4': (report\_replaceMissingValues, ["Senior Management", "NONE"]),

'5': (report\_reduceBonus, [12, 2]),

}

print("\nASSIGNMENT 9 - PYTHON LAB")

while True:

print("\n" + "="\*24)

print("MAIN LOOP")

print("(Enter a hyphen '-' to exit)")

print("(Enter a question mark '?' to see questions)")

print("\_"\*24)

option = input("Input question number: ")

print("\n" + "="\*24)

try:

if option == '-':

print("\nGoodbye!\n")

break

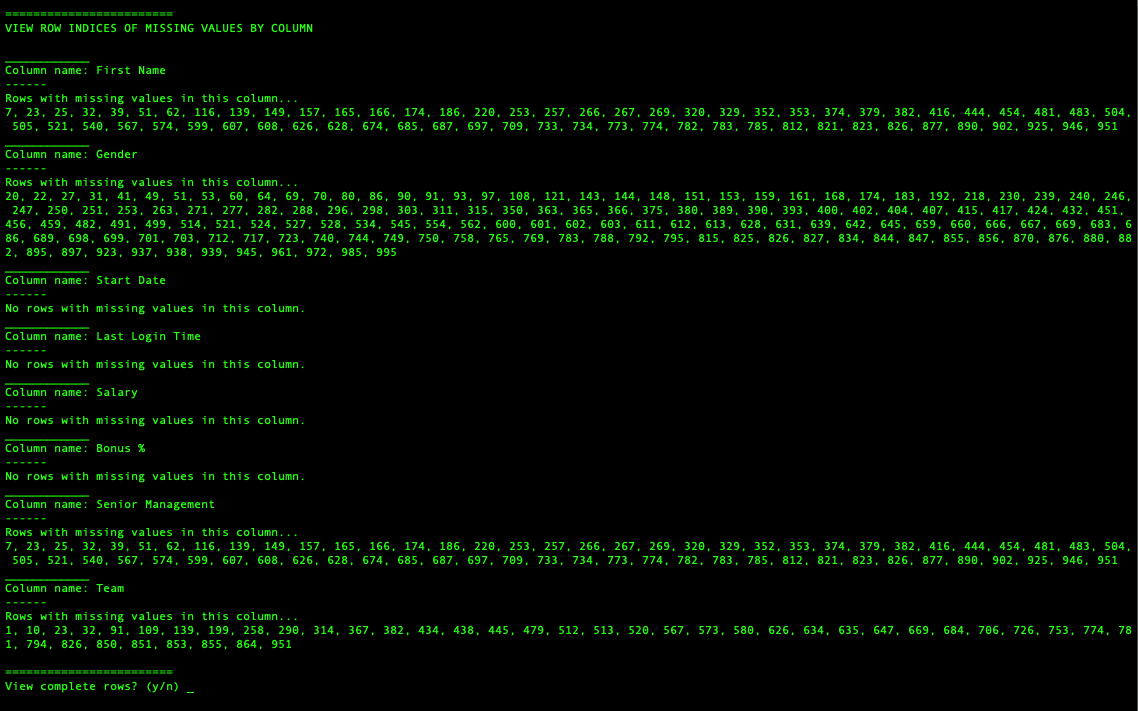
map[option][0](\*map[option][1])

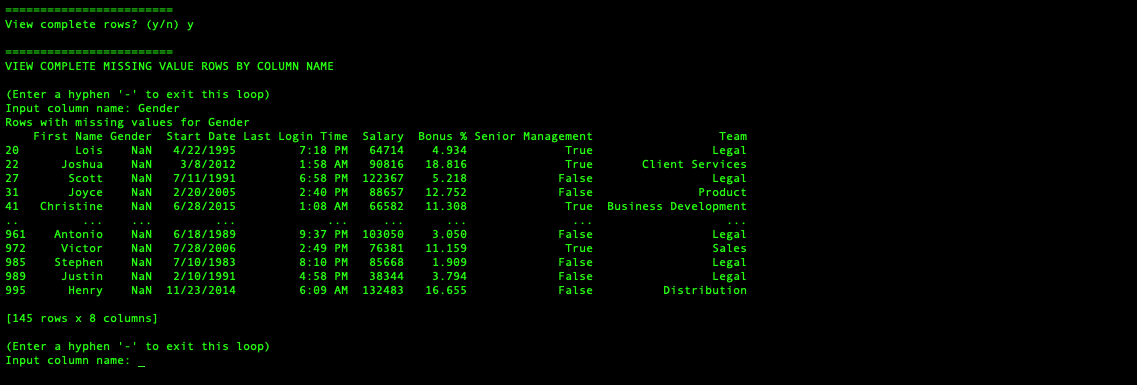
except: print("Invalid option!")

# Outputs

## Main loop

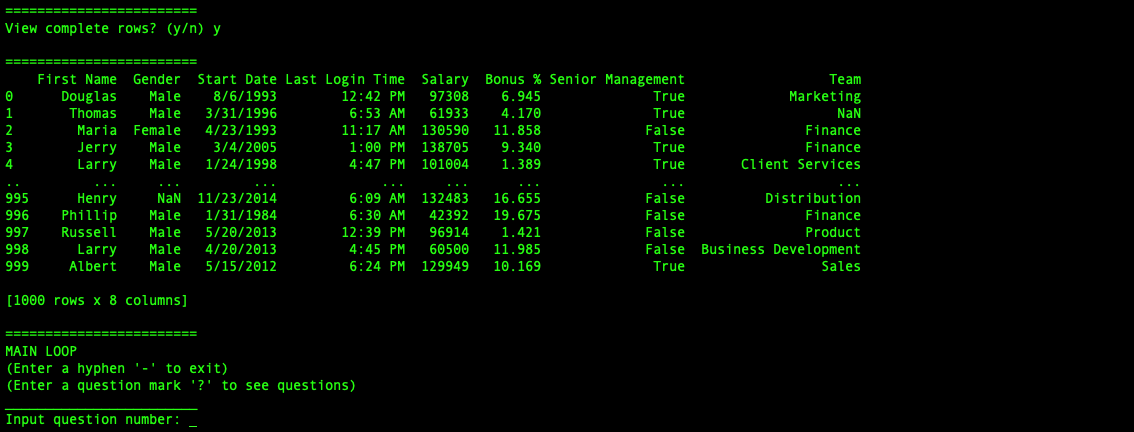
## Question 1



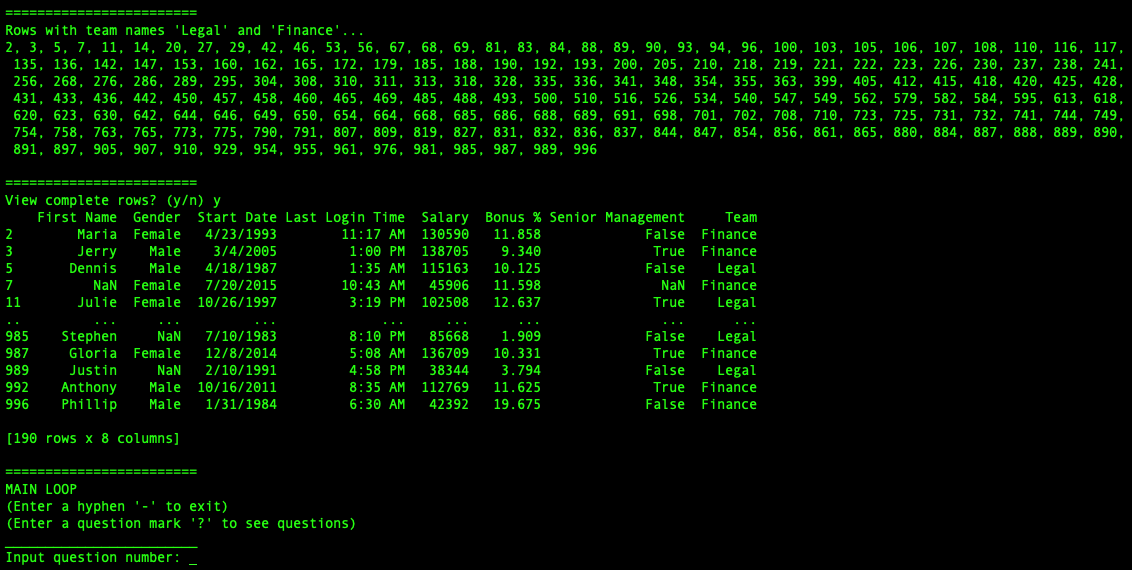


## Question 2

## 

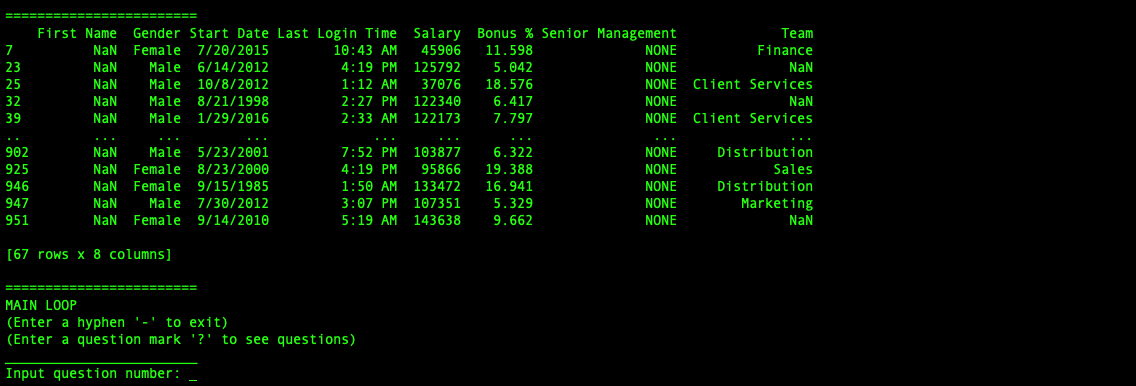


## Question 3



## Question 4

(Replacing NaN values with 'NONE' in the 'Senior Management column)



## Question 5

