DATA 690 Homework 3 (50 points - Due on Sunday, March 5, 2023 by 11:59 pm ET)

The output of this assignment for submission should be in PDF format **AND** .py or .ipynb. The name of the file should be as follows: Lastname_Firstname_Homework3.pdf (example: Thomas_Sunela_Homework3.pdf) **AND** Lastname_Firstname_Homework3.ipynb (example: Thomas_Sunela_Assignment3.ipynb. In short, you are submitting the python notebook as well as the pdf of that notebook. Do **NOT** submit .html file, the system will give you an error.

Incorrect file name will cost you points!

Instructions for converting a Jupyter Python notebook to PDF: Go to the menu and choose, File --> Download As --> html. Open that html file and print it to PDF. Submit the PDF file **NOT** the html file.

If you are using Google Colab, remember to review the PDF before submitting to ensure that all cells and answers are displayed in the PDF.

Things to note:

- Each cell should display an output
- Use both Markdown and code comments in the Jupyter Notebook as needed

IF YOU ARE MAKING ANY ASSUMPTIONS, WRITE THAT IN A MARKDOWN CELL OR COMMENT

Answer the questions asked as well, not just code

We will be using the SF Salaries dataset from Kaggle! The dataset is provided to you in Blackboard.

#1 Import pandas as pd

In [1]:
```
import pandas as pd
```

#2 Read Salaries.csv as a dataframe called sal

In [2]:
```
sal = pd.read_csv('https://raw.githubusercontent.com/SravaniRVS/DATA-690/main/Assignments%
sal
```

```
/Users/sravaniravulaparthi/opt/anaconda3/lib/python3.9/site-packages/IPython/core/interact
iveshell.py:3444: DtypeWarning: Columns (12) have mixed types.Specify dtype option on impo
rt or set low_memory=False.
  exec(code_obj, self.user_global_ns, self.user_ns)
```

Out[2]:

	Id	EmployeeName	JobTitle	BasePay	OvertimePay	OtherPay	Benefits	TotalPay
0	1	NATHANIEL FORD	GENERAL MANAGER-METROPOLITAN TRANSIT AUTHORITY	167411.18	0.00	400184.25	NaN	567595.43
1	2	GARY JIMENEZ	CAPTAIN III (POLICE DEPARTMENT)	155966.02	245131.88	137811.38	NaN	538909.28

	Id	EmployeeName	JobTitle	BasePay	OvertimePay	OtherPay	Benefits	TotalPay
2	3	ALBERT PARDINI	CAPTAIN III (POLICE DEPARTMENT)	212739.13	106088.18	16452.60	NaN	335279.91
3	4	CHRISTOPHER CHONG	WIRE ROPE CABLE MAINTENANCE MECHANIC	77916.00	56120.71	198306.90	NaN	332343.61
4	5	PATRICK GARDNER	DEPUTY CHIEF OF DEPARTMENT, (FIRE DEPARTMENT)	134401.60	9737.00	182234.59	NaN	326373.19
...	...	...	...	...	...	...	...	...
148645	148646	Carolyn A Wilson	Human Services Technician	0.00	0.00	0.00	0.0	0.00
148646	148648	Joann Anderson	Communications Dispatcher 2	0.00	0.00	0.00	0.0	0.00
148647	148649	Leon Walker	Custodian	0.00	0.00	0.00	0.0	0.00
148648	148650	Roy I Tillery	Custodian	0.00	0.00	0.00	0.0	0.00
148649	148654	Joe Lopez	Counselor, Log Cabin Ranch	0.00	0.00	-618.13	0.0	-618.13

148650 rows × 13 columns

#3 Check the head of the DataFrame. Is there anything unique? What can you tell about the data? (5 points)

In [3]:
```
sal.head()
```

Out[3]:

	Id	EmployeeName	JobTitle	BasePay	OvertimePay	OtherPay	Benefits	TotalPay	TotalPayBen
0	1	NATHANIEL FORD	GENERAL MANAGER-METROPOLITAN TRANSIT AUTHORITY	167411.18	0.00	400184.25	NaN	567595.43	5675
1	2	GARY JIMENEZ	CAPTAIN III (POLICE DEPARTMENT)	155966.02	245131.88	137811.38	NaN	538909.28	5389
2	3	ALBERT PARDINI	CAPTAIN III (POLICE DEPARTMENT)	212739.13	106088.18	16452.60	NaN	335279.91	3352
3	4	CHRISTOPHER CHONG	WIRE ROPE CABLE MAINTENANCE MECHANIC	77916.00	56120.71	198306.90	NaN	332343.61	3323
4	5	PATRICK GARDNER	DEPUTY CHIEF OF DEPARTMENT, (FIRE DEPARTMENT)	134401.60	9737.00	182234.59	NaN	326373.19	3263

Written Answer:

- We already know that the dataframe contains information about the salaries of employees in the San Francisco area
- Each row represents a different employee, and the columns provide information about each employee, such as their name, job title, and salary etc...
- As we can see the dataset contains some missing values, indicated by the **`NaN`** values in some columns
- In my opinion the **`Benefits`** column contains information about the benefits provided to each employee, such as healthcare or retirement benefits
- The **`TotalPay`** column represents the sum of an employee's **`BasePay`**, **`OvertimePay`**, and **`OtherPay`**
- The **`TotalPayBenefits`** column represents the sum of an employee's **`TotalPay`** and **`Benefits`**

#4 Use the .info() method to find out how many entries there are. Can you tell anything more about the data? (3 points)

In [4]:
```
sal.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 148650 entries, 0 to 148649
Data columns (total 13 columns):
 #   Column            Non-Null Count   Dtype  
---  ------            --------------   -----  
 0   Id                148650 non-null  int64  
 1   EmployeeName      148650 non-null  object 
 2   JobTitle          148650 non-null  object 
 3   BasePay           148045 non-null  float64
 4   OvertimePay       148650 non-null  float64
 5   OtherPay          148650 non-null  float64
 6   Benefits          112491 non-null  float64
 7   TotalPay          148650 non-null  float64
 8   TotalPayBenefits  148650 non-null  float64
 9   Year              148650 non-null  int64  
 10  Notes             0 non-null       float64
 11  Agency            148650 non-null  object 
 12  Status            38119 non-null   object 
dtypes: float64(7), int64(2), object(4)
memory usage: 14.7+ MB
```

Written Answer:

- As indicated by the RangeIndex, the dataframe contains **148,654** entries
- There are 13 columns in the dataframe, with a mix of numerical and non-numerical data types
- Some columns contain missing values, as indicated by the difference between the Non-Null Count and the total number of entries in the dataframe
  - In particular, the **`Benefits`** and **`Status`** columns contain a relatively large number of missing values.
- The **`Notes`** column appears to contain entirely missing values and I think it can be dropped from the dataframe
- The **`EmployeeName`** and **`JobTitle`** columns contain non-numeric data types
- The **`BasePay`**, **`OvertimePay`**, **`OtherPay`**, **`Benefits`**, **`TotalPay`**, and **`TotalPayBenefits`** columns are all represented as floating-point numbers (float64 data type)
- The **`Status`** column also contains a relatively large number of missing values

#5 What is the average BasePay? (3 points)

In [5]:
```
round(sal['BasePay'].mean(),2)
```

Out[5]:
```
66325.45
```

Explanation:

- Used **`.mean()`** method to calculate the avereage **`BasePay`**
- Also used **`round()`** function to limit the output to specific number of decimal points, **2** in this case

#6 What is the highest amount of OvertimePay in the dataset? (3 points)

In [6]:
```
sal['OvertimePay'].max()
```

Out[6]:
```
245131.88
```

Explanation:

- Used **`.max()`** which is a method that returns the maximum value of a column in a DataFrame or a Series

#7 What is the job title of JOSEPH DRISCOLL? (4 points)

In [7]:
```
sal.loc[sal['EmployeeName'] == 'JOSEPH DRISCOLL', ['JobTitle']]
```

Out[7]:

	JobTitle
24	CAPTAIN, FIRE SUPPRESSION

Explanation:

- The above mentioned code, first locates the row where the **`EmployeeName`** column is equal to **'JOSEPH DRISCOLL'** using the **`.loc[]`** method, and
- then selects the value in the **`JobTitle`** column for that row

#8 How much does JOSEPH DRISCOLL make (including benefits)? (4 points)

In [8]:
```
sal.loc[sal['EmployeeName'] == 'JOSEPH DRISCOLL', ['TotalPayBenefits']]
```

Out[8]:

	TotalPayBenefits
24	270324.91

Explanation:

- The code for this is similar to the above question, first it locates the row where the **`EmployeeName`** column is equal to **'JOSEPH DRISCOLL'** using the **`.loc[]`** method, and
- Then selects the value in the **`TotalPayBenefits`** column for that row

#9 What is the name of highest paid person (including benefits)? (3 points)

In [9]:
```
sal.sort_values(by='TotalPayBenefits', ascending=False).iloc[0].EmployeeName
```

Out[9]:
```
'NATHANIEL FORD'
```

Explanation:

- The code first sorts the **sal** dataframe by the **`TotalPayBenefits`** column in descending order using the **`sort_values()`** method. The **ascending=False** argument sorts the output in descending order.
- Then used **`.ilooc[0]`** to get the row with the highest TotalPayBenefits value from sorted **sal**
- Finally, the code selects the employee name from that row using **`.EmployeeName`**

#10 What is the name of lowest paid person (including benefits)? Do you notice something strange about how much he or she is paid? (5 points)

In [10]:
```
sal.sort_values(by='TotalPayBenefits').iloc[0]
```

Out[10]:
```
Id                                  148654
EmployeeName                     Joe Lopez
JobTitle            Counselor, Log Cabin Ranch
BasePay                                0.0
OvertimePay                            0.0
OtherPay                            -618.13
Benefits                               0.0
TotalPay                            -618.13
TotalPayBenefits                    -618.13
Year                                   2014
Notes                                   NaN
Agency                        San Francisco
Status                                   PT
Name: 148649, dtype: object
```

Written Answer:

- Yes, there is something strange about the amount **Joe Lopez** is paid.
- According to the output, his **`TotalPayBenefits`** is negative, which means he owes the company money instead of receiving a salary.

#11 What was the average (mean) BasePay of all employees in 2011? (5 points)

In [11]:
```
round(sal[sal['Year'] == 2011]['BasePay'].mean(),2)
```

Out[11]:
```
63595.96
```

Explanation:

- This code first selects only the rows from **sal** where the **`Year`** column is equal to 2011
- Next, the code selects only the **`BasePay`** column from the filtered rows using bracket notation
- Finally, the code uses the **`mean()`** method to calculate the average (mean) of the values in the 'BasePay' column

#12 How many unique job titles are there? (5 points)

In [12]:
```
sal['JobTitle'].nunique()
```

Out[12]:
```
2158
```

Explanation:

- This code selects only the **`JobTitle`** column from **sal** using bracket notation.
- Then, the code uses the **`nunique()`** method to calculate the number of unique job titles in the **JobTitle** column

#13 What are the top 5 most common jobs? (5 points)

In [13]:
```
sal['JobTitle'].value_counts().head()
```

Out[13]:
```
Transit Operator                7036
Special Nurse                   4389
Registered Nurse                3736
Public Svc Aide-Public Works    2518
Police Officer 3                2421
Name: JobTitle, dtype: int64
```

Explanation:

- This code selects only the **`JobTitle`** column from **sal** using bracket notation
- Then, the code uses the **`value_counts()`** method to count the number of occurrences of each unique job title in the **JobTitle** column
- Finally, the code uses the **`head()`** method to select only the top 5 most common job titles

#14 How many Job Titles were represented by only one person in 2013? (e.g. Job Titles with only one occurence in 2013?) (5 points)

In [14]:
```
sum(sal[sal['Year']==2013]['JobTitle'].value_counts() == 1)
```

Out[14]:
```
202
```

Explanation:

- This code first selects only the rows in **sal** where the **`Year`** column is equal to **2013**
- Then, it selects only the **`JobTitle`** column using bracket notation
- Next, it uses the **`value_counts()`** method to count the number of occurrences of each unique job title in the **JobTitle** column and used the comparison operator **==** to check which counts are equal to **1**, and
- Then uses the **`sum()`** function to count the number of True values