SF Salaries Exercise

Welcome to a quick exercise for you to practice your pandas skills! We will be using the SF Salaries Dataset from Kaggle! Just follow along and complete the tasks outlined in bold below. The tasks will get harder and harder as you go along.

Import pandas as pd.

In [2]:
```
import pandas as pd
```

Read Salaries.csv as a dataframe called sal.

In [3]:
```
df=pd.read_csv("C:/Users/STUDENT/Downloads/Salaries.csv")
```

```
C:\Users\STUDENT\anaconda3\lib\site-packages\IPython\core\interactiveshell.py:3063: DtypeWarning: Columns (3,4,5,6,12) have mixed types.Specify dtype option on import or set low_memory=False.
  interactivity=interactivity, compiler=compiler, result=result)
```

Check the head of the DataFrame.

In [4]:
```
df.head()
```

Out[4]:

	Id	EmployeeName	JobTitle	BasePay	OvertimePay	OtherPay	Benefits	TotalPay	TotalPayBenefits	Year	Notes
0	1	NATHANIEL FORD	GENERAL MANAGER-METROPOLITAN TRANSIT AUTHORITY	167411	0	400184	NaN	567595.43	567595.43	2011	NaN
1	2	GARY JIMENEZ	CAPTAIN III (POLICE DEPARTMENT)	155966	245132	137811	NaN	538909.28	538909.28	2011	NaN
2	3	ALBERT PARDINI	CAPTAIN III (POLICE DEPARTMENT)	212739	106088	16452.6	NaN	335279.91	335279.91	2011	NaN
3	4	CHRISTOPHER CHONG	WIRE ROPE CABLE MAINTENANCE	77916	56120.7	198307	NaN	332343.61	332343.61	2011	NaN

Use the .info() method to find out how many entries there are.

In [5]:
```
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 148654 entries, 0 to 148653
Data columns (total 13 columns):
 #   Column            Non-Null Count   Dtype  
---  ------            --------------   -----  
 0   Id                148654 non-null  int64  
 1   EmployeeName      148654 non-null  object 
 2   JobTitle          148654 non-null  object 
 3   BasePay           148049 non-null  object 
 4   OvertimePay       148654 non-null  object 
 5   OtherPay          148654 non-null  object 
 6   Benefits          112495 non-null  object 
 7   TotalPay          148654 non-null  float64
 8   TotalPayBenefits  148654 non-null  float64
 9   Year              148654 non-null  int64  
 10  Notes             0 non-null       float64
 11  Agency            148654 non-null  object 
 12  Status            38119 non-null   object 
dtypes: float64(3), int64(2), object(8)
memory usage: 14.7+ MB
```

What is the average BasePay ?

In [6]:
```
df["BasePay"].mean()
```

```
---------------------------------------------------------------------------
TypeError                                 Traceback (most recent call last)
<ipython-input-6-f7743e53fccf> in <module>
----> 1 df["BasePay"].mean()

~\anaconda3\lib\site-packages\pandas\core\generic.py in stat_func(self, axis, skipna, level, numeric_only, **kwargs)
  11215                 return self._agg_by_level(name, axis=axis, level=level, skipna=skipna)
  11216             return self._reduce(
> 11217                 f, name, axis=axis, skipna=skipna, numeric_only=numeric_only
  11218             )
  11219 

~\anaconda3\lib\site-packages\pandas\core\series.py in _reduce(self, op, name, axis, skipna, numeric_only, filter_type, **kwds)
   3889                 )
   3890             with np.errstate(all="ignore"):
-> 3891                 return op(delegate, skipna=skipna, **kwds)
   3892 
   3893         # TODO(EA) dispatch to Index

~\anaconda3\lib\site-packages\pandas\core\nanops.py in _f(*args, **kwargs)
     67             try:
     68                 with np.errstate(invalid="ignore"):
---> 69                     return f(*args, **kwargs)
     70             except ValueError as e:
     71                 # we want to transform an object array

~\anaconda3\lib\site-packages\pandas\core\nanops.py in f(values, axis, skipna, **kwds)
    123                     result = alt(values, axis=axis, skipna=skipna, **kwds)
    124                 else:
--> 125                     result = alt(values, axis=axis, skipna=skipna, **kwds)
    126 
    127             return result

~\anaconda3\lib\site-packages\pandas\core\nanops.py in nanmean(values, axis, skipna, mask)
    540             dtype_count = dtype
    541         count = _get_counts(values.shape, mask, axis, dtype=dtype_count)
--> 542         the_sum = _ensure_numeric(values.sum(axis, dtype=dtype_sum))
    543 
    544         if axis is not None and getattr(the_sum, "ndim", False):

~\anaconda3\lib\site-packages\numpy\core\_methods.py in _sum(a, axis, dtype, out, keepdims, initial, where)
     36 def _sum(a, axis=None, dtype=None, out=None, keepdims=False,
     37          initial=_NoValue, where=True):
---> 38     return umr_sum(a, axis, dtype, out, keepdims, initial, where)
     39 
     40 def _prod(a, axis=None, dtype=None, out=None, keepdims=False,
```

What is the highest amount of OvertimePay in the dataset ?

In [11]:
```

```

Out[11]:
```
245131.88
```

What is the job title of JOSEPH DRISCOLL ? Note: Use all caps, otherwise you may get an answer that doesn't match up (there is also a lowercase Joseph Driscoll).

In [7]:
```
df[df["EmployeeName"]=="JOSEPH DRISCOLL"]["TotalPayBenefits"]
```

Out[7]:
```
24    270324.91
Name: TotalPayBenefits, dtype: float64
```

How much does JOSEPH DRISCOLL make (including benefits)?

In [8]:
```
df[df["TotalPayBenefits"]==df["TotalPayBenefits"].max()]
```

Out[8]:

	Id	EmployeeName	JobTitle	BasePay	OvertimePay	OtherPay	Benefits	TotalPay	TotalPayBenefits	Year	Notes
0	1	NATHANIEL FORD	GENERAL MANAGER-METROPOLITAN TRANSIT	167411	0	400184	NaN	567595.43	567595.43	2011	NaN

What is the name of highest paid person (including benefits)?

In [9]:
```
df[df["TotalPayBenefits"]==df["TotalPayBenefits"].max()]
```

Out[9]:

	Id	EmployeeName	JobTitle	BasePay	OvertimePay	OtherPay	Benefits	TotalPay	TotalPayBenefits	Year	Notes
0	1	NATHANIEL FORD	GENERAL MANAGER-METROPOLITAN TRANSIT	167411	0	400184	NaN	567595.43	567595.43	2011	NaN

What is the name of lowest paid person (including benefits)? Do you notice something strange about how much he

In [10]:
```
df[df["TotalPayBenefits"]==df["TotalPayBenefits"].min()]
```

Out[10]:

	Id	EmployeeName	JobTitle	BasePay	OvertimePay	OtherPay	Benefits	TotalPay	TotalPayBenefits	Year	Not
148653	148654	Joe Lopez	Counselor, Log Cabin Ranch	0.00	0.00	-618.13	0.00	-618.13	-618.13	2014	N

What was the average (mean) BasePay of all employees per year? (2011-2014) ?

In [12]:
```
df.groupby("Year").mean()
```

Out[12]:

Year	Id	TotalPay	TotalPayBenefits	Notes
2011	18080.0	71744.103871	71744.103871	NaN
2012	54542.5	74113.262265	100553.229232	NaN
2013	91728.5	77611.443142	101440.519714	NaN
2014	129593.0	75463.918140	100250.918884	NaN

How many unique job titles are there?

In [13]:
```
df["JobTitle"].nunique()
```

Out[13]:
```
2159
```

What are the top 5 most common jobs?

In [15]:
```
group=df.groupby("JobTitle").count()
top5=group.sort_values(by="Id",ascending=False)[:5]
top5["Id"]
```

Out[15]:
```
JobTitle
Transit Operator                7036
Special Nurse                   4389
Registered Nurse                3736
Public Svc Aide-Public Works    2518
Police Officer 3                2421
Name: Id, dtype: int64
```

How many Job Titles were represented by only one person in 2013? (e.g. Job Titles with only one occurence in

In [17]:
```
copy_sal=df[df["Year"]==2013]
group=copy_sal.groupby("JobTitle").count()
count=group[group["Id"]==1]
count.count()["Id"]
```

Out[17]:
```
202
```

How many people have the word Chief in their job title? (This is pretty tricky)

In [21]:
```
def find_chief(job_title):
    if 'chief' in job_title.lower().split():
        return True
    else:
        return False
```

In [22]:
```
df=pd.read_csv('Salaries.csv')
sum(df['JobTitle'].apply(lambda x:find_chief(x)))
```

Out[22]:
```
477
```

Bonus: Is there a correlation between length of the Job Title string and Salary?

In [23]:
```
df["title_len"]=df["JobTitle"].apply(len)
```

In [24]:
```
df[["title_len","TotalPayBenefits"]].corr()
```

Out[24]:

	title_len	TotalPayBenefits
title_len	1.000000	-0.036878
TotalPayBenefits	-0.036878	1.000000

Great Job!