Part_2_Working_with_CSV_Files

October 31, 2022

0.1 1. Reading the csv file:

```
[1]: import pandas as pd
     import numpy as np
```

```
[2]: df = pd.read_csv('/content/sample_data/tips.csv')
```

a) To check name of columns in dataframe: ITS NOT A METHOD BUT ATTRIBUTE

```
[3]: df.columns
```

```
[3]: Index(['total_bill', 'tip', 'sex', 'smoker', 'day', 'time', 'size',
            'price_per_person', 'Payer Name', 'CC Number', 'Payment ID'],
           dtype='object')
```

b) To check index range of data in dataframe: ITS NOT A METHOD BUT ATTRIBUTE

```
[4]: df.index
```

```
[4]: RangeIndex(start=0, stop=244, step=1)
```

So, Number of rows = stop - start = 244 - 0 = 244

c) To understand the general structure of dataframe: use head() function

```
[5]: df.head()
```

```
[5]:    total_bill   tip     sex smoker  day    time  size  price_per_person  \
     0       16.99  1.01  Female     No  Sun  Dinner     2              8.49
     1       10.34  1.66    Male     No  Sun  Dinner     3              3.45
     2       21.01  3.50    Male     No  Sun  Dinner     3              7.00
     3       23.68  3.31    Male     No  Sun  Dinner     2             11.84
     4       24.59  3.61  Female     No  Sun  Dinner     4              6.15

                Payer Name         CC Number Payment ID
     0  Christy Cunningham  3560325168603410    Sun2959
     1      Douglas Tucker  4478071379779230    Sun4608
     2      Travis Walters  6011812112971322    Sun4458
```

```
3    Nathaniel Harris  4676137647685994    Sun5260
4        Tonya Carter  4832732618637221    Sun2251
```

To check 10 top rows, use ***head(10)****. To check last 5 rows, use* ***tail()****. To check last 10 rows, use* ***tail(10)****.*

d) To get statistical information of the dataframe:

[6]:
```
df.describe()
```

[6]:
```
       total_bill         tip        size  price_per_person     CC Number
count  244.000000  244.000000  244.000000        244.000000  2.440000e+02
mean    19.785943    2.998279    2.569672          7.888197  2.563496e+15
std      8.902412    1.383638    0.951100          2.914234  2.369340e+15
min      3.070000    1.000000    1.000000          2.880000  6.040679e+10
25%     13.347500    2.000000    2.000000          5.800000  3.040731e+13
50%     17.795000    2.900000    2.000000          7.255000  3.525318e+15
75%     24.127500    3.562500    3.000000          9.390000  4.553675e+15
max     50.810000   10.000000    6.000000         20.270000  6.596454e+15
```

[7]:
```
# you can transpose to see the other side of view

(df.describe()).transpose()
```

[7]:
```
                  count          mean           std           min  \
total_bill        244.0  1.978594e+01  8.902412e+00  3.070000e+00
tip               244.0  2.998279e+00  1.383638e+00  1.000000e+00
size              244.0  2.569672e+00  9.510998e-01  1.000000e+00
price_per_person  244.0  7.888197e+00  2.914234e+00  2.880000e+00
CC Number         244.0  2.563496e+15  2.369340e+15  6.040679e+10

                           25%           50%           75%           max
total_bill        1.334750e+01  1.779500e+01  2.412750e+01  5.081000e+01
tip               2.000000e+00  2.900000e+00  3.562500e+00  1.000000e+01
size              2.000000e+00  2.000000e+00  3.000000e+00  6.000000e+00
price_per_person  5.800000e+00  7.255000e+00  9.390000e+00  2.027000e+01
CC Number         3.040731e+13  3.525318e+15  4.553675e+15  6.596454e+15
```

0.2 2. Working with Columns:

[8]:
```
type(df['total_bill'])
```

[8]:
```
pandas.core.series.Series
```

a) Grabbing multiple columns:

```
[9]: # METHOD - 1

     columnstograb = ['total_bill', 'tip']
     df[columnstograb].head()
```

```
[9]:    total_bill   tip
     0       16.99  1.01
     1       10.34  1.66
     2       21.01  3.50
     3       23.68  3.31
     4       24.59  3.61
```

```
[10]: # METHOD - 2

      df[['total_bill', 'tip']].head()
```

```
[10]:    total_bill   tip
      0       16.99  1.01
      1       10.34  1.66
      2       21.01  3.50
      3       23.68  3.31
      4       24.59  3.61
```

b) Creating a new column:

```
[11]: # This new column will be created at end
      # If such name column already exists, that will be overwritten

      df['tip_prec'] = 100*df['total_bill'] / df['tip']
      df.head()
```

```
[11]:    total_bill   tip     sex smoker  day    time  size  price_per_person  \
      0       16.99  1.01  Female     No  Sun  Dinner     2              8.49
      1       10.34  1.66    Male     No  Sun  Dinner     3              3.45
      2       21.01  3.50    Male     No  Sun  Dinner     3              7.00
      3       23.68  3.31    Male     No  Sun  Dinner     2             11.84
      4       24.59  3.61  Female     No  Sun  Dinner     4              6.15

                 Payer Name         CC Number Payment ID     tip_prec
      0  Christy Cunningham  3560325168603410    Sun2959  1682.178218
      1      Douglas Tucker  4478071379779230    Sun4608   622.891566
      2      Travis Walters  6011812112971322    Sun4458   600.285714
      3    Nathaniel Harris  4676137647685994    Sun5260   715.407855
      4        Tonya Carter  4832732618637221    Sun2251   681.163435
```

c) Controlling precision of any column:

```
[12]: df['tip_prec'] = df['tip_prec'].round(2)
      df.head()
```

```
[12]:    total_bill   tip     sex smoker  day    time  size  price_per_person  \
      0       16.99  1.01  Female     No  Sun  Dinner     2              8.49
      1       10.34  1.66    Male     No  Sun  Dinner     3              3.45
      2       21.01  3.50    Male     No  Sun  Dinner     3              7.00
      3       23.68  3.31    Male     No  Sun  Dinner     2             11.84
      4       24.59  3.61  Female     No  Sun  Dinner     4              6.15

                 Payer Name         CC Number Payment ID  tip_prec
      0  Christy Cunningham  3560325168603410    Sun2959   1682.18
      1      Douglas Tucker  4478071379779230    Sun4608    622.89
      2      Travis Walters  6011812112971322    Sun4458    600.29
      3    Nathaniel Harris  4676137647685994    Sun5260    715.41
      4        Tonya Carter  4832732618637221    Sun2251    681.16
```

d) Removing columns:

- To drop a column, must write axis = 1.
- To drop a row, no need to write axis = 0 as it is set as default.
- To make changes permanent, either use **df =**, or use **inplace = True**

```
[13]: df = df.drop('tip_prec', axis=1)
```

0.3 3. Working with Rows:

The index in dataframe is considered as **Primary key** which has the values as unique.

If we want to set some column of data as index, make sure that it is having unique value for each row.

```
[14]: df.set_index('Payment ID', inplace = True)
```

```
[15]: df.head()
```

```
[15]:             total_bill   tip     sex smoker  day    time  size  \
      Payment ID
      Sun2959          16.99  1.01  Female     No  Sun  Dinner     2
      Sun4608          10.34  1.66    Male     No  Sun  Dinner     3
      Sun4458          21.01  3.50    Male     No  Sun  Dinner     3
      Sun5260          23.68  3.31    Male     No  Sun  Dinner     2
      Sun2251          24.59  3.61  Female     No  Sun  Dinner     4

                  price_per_person          Payer Name         CC Number
      Payment ID
      Sun2959                 8.49  Christy Cunningham  3560325168603410
      Sun4608                 3.45      Douglas Tucker  4478071379779230
```

```
Sun4458                  7.00      Travis Walters  6011812112971322
Sun5260                 11.84    Nathaniel Harris  4676137647685994
Sun2251                  6.15        Tonya Carter  4832732618637221
```

[34]:
```
df.reset_index(inplace = True)
```

a) To grab a row using integer index: use iloc[ ]

Here **i** in **iloc** means integer

[16]:
```
df.iloc[0]
```

[16]:
```
total_bill                          16.99
tip                                  1.01
sex                                Female
smoker                                 No
day                                   Sun
time                               Dinner
size                                    2
price_per_person                     8.49
Payer Name            Christy Cunningham
CC Number               3560325168603410
Name: Sun2959, dtype: object
```

[17]:
```
# to grap multiple rows

df.iloc[0:4]
```

[17]:
```
            total_bill   tip     sex smoker  day    time  size  \
Payment ID
Sun2959          16.99  1.01  Female     No  Sun  Dinner     2
Sun4608          10.34  1.66    Male     No  Sun  Dinner     3
Sun4458          21.01  3.50    Male     No  Sun  Dinner     3
Sun5260          23.68  3.31    Male     No  Sun  Dinner     2

            price_per_person          Payer Name         CC Number
Payment ID
Sun2959                 8.49  Christy Cunningham  3560325168603410
Sun4608                 3.45      Douglas Tucker  4478071379779230
Sun4458                 7.00      Travis Walters  6011812112971322
Sun5260                11.84    Nathaniel Harris  4676137647685994
```

b) To grab a row using real named index: use loc[ ]

[18]:
```
# we are assuming here we have named index of Payment ID

df.loc['Sun5260']
```

[18]:
```
total_bill                       23.68
tip                               3.31
sex                               Male
smoker                              No
day                                Sun
time                            Dinner
size                                 2
price_per_person                 11.84
Payer Name          Nathaniel Harris
CC Number           4676137647685994
Name: Sun5260, dtype: object
```

[20]:
```
# to grap multiple rows for named index: same like multiple columns grabbing

# Method - 1
rowstograb = ['Sun5260', 'Sun4608']
df.loc[rowstograb]
```

[20]:
```
            total_bill   tip   sex smoker  day    time  size  \
Payment ID
Sun5260          23.68  3.31  Male     No  Sun  Dinner     2
Sun4608          10.34  1.66  Male     No  Sun  Dinner     3

            price_per_person        Payer Name         CC Number
Payment ID
Sun5260                11.84  Nathaniel Harris  4676137647685994
Sun4608                 3.45    Douglas Tucker  4478071379779230
```

[21]:
```
# Method - 2

df.loc[['Sun5260', 'Sun4608']]
```

[21]:
```
            total_bill   tip   sex smoker  day    time  size  \
Payment ID
Sun5260          23.68  3.31  Male     No  Sun  Dinner     2
Sun4608          10.34  1.66  Male     No  Sun  Dinner     3

            price_per_person        Payer Name         CC Number
Payment ID
Sun5260                11.84  Nathaniel Harris  4676137647685994
Sun4608                 3.45    Douglas Tucker  4478071379779230
```

c) To drop a row:

[22]:
```
# deleting using named index

df.drop('Sun4608', inplace = True)
```

```
[23]: # to delete using numeric index, must use slicing with iloc

      df = df.iloc[1:]
```

d) Inserting a new row:

```
[24]: # copy the first row and append it. It will be appended at last automatically.␣
       ↪It will create duplication but pandas is Ok with that!

      new_row = df.iloc[0]
      df = df.append(new_row)
```

```
[25]: df
```

```
[25]:             total_bill   tip     sex smoker   day    time  size  \
      Payment ID
      Sun4458          21.01  3.50    Male     No   Sun  Dinner     3
      Sun5260          23.68  3.31    Male     No   Sun  Dinner     2
      Sun2251          24.59  3.61  Female     No   Sun  Dinner     4
      Sun9679          25.29  4.71    Male     No   Sun  Dinner     4
      Sun5985           8.77  2.00    Male     No   Sun  Dinner     2
      …                  …    …      …     …    …     …     …
      Sat1766          27.18  2.00  Female    Yes   Sat  Dinner     2
      Sat3880          22.67  2.00    Male    Yes   Sat  Dinner     2
      Sat17            17.82  1.75    Male     No   Sat  Dinner     2
      Thur672          18.78  3.00  Female     No  Thur  Dinner     2
      Sun4458          21.01  3.50    Male     No   Sun  Dinner     3

                  price_per_person          Payer Name         CC Number
      Payment ID
      Sun4458                 7.00      Travis Walters  6011812112971322
      Sun5260                11.84   Nathaniel Harris  4676137647685994
      Sun2251                 6.15        Tonya Carter  4832732618637221
      Sun9679                 6.32          Erik Smith   213140353657882
      Sun5985                 4.38  Kristopher Johnson  2223727524230344
      …                        …                 …                …
      Sat1766                13.59      Monica Sanders  3506806155565404
      Sat3880                11.34          Keith Wong  6011891618747196
      Sat17                   8.91        Dennis Dixon     4375220550950
      Thur672                 9.39    Michelle Hardin  3511451626698139
      Sun4458                 7.00      Travis Walters  6011812112971322

      [243 rows x 10 columns]
```

0.4 3. Conditional Filtering:

[26]:
```
df['size'] > 4
```

[26]:
```
Payment ID
Sun4458     False
Sun5260     False
Sun2251     False
Sun9679     False
Sun5985     False
            …
Sat1766     False
Sat3880     False
Sat17       False
Thur672     False
Sun4458     False
Name: size, Length: 243, dtype: bool
```

[27]:
```
df[df['size'] > 4]
```

[27]:
```
            total_bill   tip     sex smoker   day    time  size  \
Payment ID
Thur3948         29.80  4.20  Female     No  Thur   Lunch     6
Thur1025         34.30  6.70    Male     No  Thur   Lunch     6
Thur3621         41.19  5.00    Male     No  Thur   Lunch     5
Thur6179         27.05  5.00  Female     No  Thur   Lunch     6
Sun9176          29.85  5.14  Female     No   Sun  Dinner     5
Sun7518          48.17  5.00    Male     No   Sun  Dinner     6
Sun5842          20.69  5.00    Male     No   Sun  Dinner     5
Sun9987          30.46  2.00    Male    Yes   Sun  Dinner     5
Sat7320          28.15  3.00    Male    Yes   Sat  Dinner     5

            price_per_person         Payer Name          CC Number
Payment ID
Thur3948                4.97     Angela Sanchez       503857080488
Thur1025                5.72     Steven Carlson   3526515703718508
Thur3621                8.24       Eric Andrews   4356531761046453
Thur6179                4.51       Regina Jones      4311048695487
Sun9176                 5.97     Madison Wilson   4210875236164664
Sun7518                 8.03      Ryan Gonzales   3523151482063321
Sun5842                 4.14     Joseph Howell     30362407455623
Sun9987                 6.09      David Barrett   4792882899700988
Sat7320                 5.63  Shawn Barnett PhD      4590982568244
```

To apply multiple conditions, must use & and |. Never use |AND, OR because they apply condition as a whole but & and | apply condition by reading one by one row.

Syntax: * **df[(condition2) & (condition2)]** * **df[(condition2) | (condition2)]**

```
[28]: df[(df['size'] > 4) & (df['tip'] > 2.00)]
```

```
[28]:             total_bill   tip     sex smoker   day    time  size  \
      Payment ID
      Thur3948         29.80  4.20  Female     No  Thur   Lunch     6
      Thur1025         34.30  6.70    Male     No  Thur   Lunch     6
      Thur3621         41.19  5.00    Male     No  Thur   Lunch     5
      Thur6179         27.05  5.00  Female     No  Thur   Lunch     6
      Sun9176          29.85  5.14  Female     No   Sun  Dinner     5
      Sun7518          48.17  5.00    Male     No   Sun  Dinner     6
      Sun5842          20.69  5.00    Male     No   Sun  Dinner     5
      Sat7320          28.15  3.00    Male    Yes   Sat  Dinner     5

                  price_per_person         Payer Name         CC Number
      Payment ID
      Thur3948                4.97     Angela Sanchez      503857080488
      Thur1025                5.72     Steven Carlson  3526515703718508
      Thur3621                8.24       Eric Andrews  4356531761046453
      Thur6179                4.51       Regina Jones     4311048695487
      Sun9176                 5.97     Madison Wilson  4210875236164664
      Sun7518                 8.03      Ryan Gonzales  3523151482063321
      Sun5842                 4.14      Joseph Howell    30362407455623
      Sat7320                 5.63  Shawn Barnett PhD     4590982568244
```

```
[29]: df[(df['day'] == 'Sat') | (df['day'] == 'Sun')]
```

```
[29]:             total_bill   tip     sex smoker  day    time  size  \
      Payment ID
      Sun4458          21.01  3.50    Male     No  Sun  Dinner     3
      Sun5260          23.68  3.31    Male     No  Sun  Dinner     2
      Sun2251          24.59  3.61  Female     No  Sun  Dinner     4
      Sun9679          25.29  4.71    Male     No  Sun  Dinner     4
      Sun5985           8.77  2.00    Male     No  Sun  Dinner     2
      …                  …     …       …      …   …       …     …
      Sat2657          29.03  5.92    Male     No  Sat  Dinner     3
      Sat1766          27.18  2.00  Female    Yes  Sat  Dinner     2
      Sat3880          22.67  2.00    Male    Yes  Sat  Dinner     2
      Sat17            17.82  1.75    Male     No  Sat  Dinner     2
      Sun4458          21.01  3.50    Male     No  Sun  Dinner     3

                  price_per_person           Payer Name         CC Number
      Payment ID
      Sun4458                 7.00       Travis Walters  6011812112971322
      Sun5260                11.84    Nathaniel Harris  4676137647685994
      Sun2251                 6.15         Tonya Carter  4832732618637221
      Sun9679                 6.32           Erik Smith   213140353657882
      Sun5985                 4.38  Kristopher Johnson  2223727524230344
```

```
…                         …                 …                 …
Sat2657                9.68      Michael Avila  5296068606052842
Sat1766               13.59    Monica Sanders   3506806155565404
Sat3880               11.34         Keith Wong  6011891618747196
Sat17                  8.91       Dennis Dixon     4375220550950
Sun4458                7.00     Travis Walters  6011812112971322

[162 rows x 10 columns]
```

[30]:
```
# if we want to apply multiple conditions on same column
options = ['Sat', 'Sun']
df[df['day'].isin(options)]
```

[30]:
```
            total_bill   tip     sex smoker  day    time  size  \
Payment ID
Sun4458          21.01  3.50    Male     No  Sun  Dinner     3
Sun5260          23.68  3.31    Male     No  Sun  Dinner     2
Sun2251          24.59  3.61  Female     No  Sun  Dinner     4
Sun9679          25.29  4.71    Male     No  Sun  Dinner     4
Sun5985           8.77  2.00    Male     No  Sun  Dinner     2
…                    …     …       …    …    …       …
Sat2657          29.03  5.92    Male     No  Sat  Dinner     3
Sat1766          27.18  2.00  Female    Yes  Sat  Dinner     2
Sat3880          22.67  2.00    Male    Yes  Sat  Dinner     2
Sat17            17.82  1.75    Male     No  Sat  Dinner     2
Sun4458          21.01  3.50    Male     No  Sun  Dinner     3

            price_per_person          Payer Name         CC Number
Payment ID
Sun4458                 7.00      Travis Walters  6011812112971322
Sun5260                11.84   Nathaniel Harris   4676137647685994
Sun2251                 6.15        Tonya Carter  4832732618637221
Sun9679                 6.32          Erik Smith   213140353657882
Sun5985                 4.38  Kristopher Johnson  2223727524230344
…                         …                 …                 …
Sat2657                 9.68       Michael Avila  5296068606052842
Sat1766                13.59      Monica Sanders  3506806155565404
Sat3880                11.34          Keith Wong  6011891618747196
Sat17                   8.91        Dennis Dixon     4375220550950
Sun4458                 7.00      Travis Walters  6011812112971322

[162 rows x 10 columns]
```

0.5 4. Applying Customized Functions on Columns:

The customize function should return only single value in each calling. Otherwise it will give error.

[31]:
```
def takelast4digits(num):
  return int(str(num)[:4])

# we cannot do slicing with int, so must convert int to string then do slicing␣
↪and then convert back to int
```

[32]:
```
df['CC Number'].apply(takelast4digits)  # do not write the parameter brackets␣
↪in it. Pandas will do it automatically by considering one row each time as␣
↪data for the function
```

[32]:
```
Payment ID
Sun4458     6011
Sun5260     4676
Sun2251     4832
Sun9679     2131
Sun5985     2223
            …
Sat1766     3506
Sat3880     6011
Sat17       4375
Thur672     3511
Sun4458     6011
Name: CC Number, Length: 243, dtype: int64
```

[33]:
```
# Method 2
df['CC Number'].apply(lambda num:int(str(num)[:4]))
```

[33]:
```
Payment ID
Sun4458     6011
Sun5260     4676
Sun2251     4832
Sun9679     2131
Sun5985     2223
            …
Sat1766     3506
Sat3880     6011
Sat17       4375
Thur672     3511
Sun4458     6011
Name: CC Number, Length: 243, dtype: int64
```

If we want to apply customize function of multiple columns:

[34]:
```
def quality(total_bill, tip):
  if tip/total_bill > 0.25:
    return 'Generous'
  else:
    return 'Other'
```

[35]:
```
df['Quality'] = np.vectorize(quality)(df['total_bill'], df['tip'])   # np.
 ↪vectorize(functiona_name)(columns_names)
df.head()
```

[35]:
```
            total_bill   tip     sex smoker  day    time  size  \
Payment ID
Sun4458          21.01  3.50    Male     No  Sun  Dinner     3
Sun5260          23.68  3.31    Male     No  Sun  Dinner     2
Sun2251          24.59  3.61  Female     No  Sun  Dinner     4
Sun9679          25.29  4.71    Male     No  Sun  Dinner     4
Sun5985           8.77  2.00    Male     No  Sun  Dinner     2

            price_per_person          Payer Name         CC Number Quality
Payment ID
Sun4458                 7.00      Travis Walters  6011812112971322   Other
Sun5260                11.84   Nathaniel Harris  4676137647685994   Other
Sun2251                 6.15        Tonya Carter  4832732618637221   Other
Sun9679                 6.32          Erik Smith   213140353657882   Other
Sun5985                 4.38  Kristopher Johnson  2223727524230344   Other
```

0.6 4. Statistical Analysis of the data

[36]:
```
# to sort the data according to a specific column

df.sort_values('tip')
```

[36]:
```
            total_bill    tip     sex smoker   day    time  size  \
Payment ID
Sat3455           3.07   1.00  Female    Yes   Sat  Dinner     1
Fri3780           5.75   1.00  Female    Yes   Fri  Dinner     2
Sat5032          12.60   1.00    Male    Yes   Sat  Dinner     2
Sat4801           7.25   1.00  Female     No   Sat  Dinner     1
Sat6983          12.90   1.10  Female    Yes   Sat  Dinner     2
…                  …    …      …    …    …      …    …
Thur1025         34.30   6.70    Male     No  Thur   Lunch     6
Sat8139          48.27   6.73    Male     No   Sat  Dinner     4
Sat239           39.42   7.58    Male     No   Sat  Dinner     4
Sat4590          48.33   9.00    Male     No   Sat  Dinner     4
Sat1954          50.81  10.00    Male    Yes   Sat  Dinner     3

            price_per_person       Payer Name         CC Number   Quality
Payment ID
Sat3455                 3.07    Tiffany Brock  4359488526995267  Generous
Fri3780                 2.88     Leah Ramirez  3508911676966392     Other
```

```
Sat5032                      6.30    Matthew Myers  3543676378973965     Other
Sat4801                      7.25      Terri Jones  3559221007826887     Other
Sat6983                      6.45     Jessica Owen     4726904879471     Other
…                           …                …                 …        …
Thur1025                     5.72   Steven Carlson  3526515703718508     Other
Sat8139                     12.07      Brian Ortiz  6596453823950595     Other
Sat239                       9.86   Lance Peterson  3542584061609808     Other
Sat4590                     12.08  Alex Williamson      676218815212     Other
Sat1954                     16.94    Gregory Clark  5473850968388236     Other

[243 rows x 11 columns]
```

By default, the sorting order is ascending. If we want to sort it in descending way, we need to write **ascending = False**.

[37]:
```
df.sort_values('tip', ascending=False)
```

[37]:
```
            total_bill    tip     sex smoker   day    time  size  \
Payment ID
Sat1954          50.81  10.00    Male    Yes   Sat  Dinner     3
Sat4590          48.33   9.00    Male     No   Sat  Dinner     4
Sat239           39.42   7.58    Male     No   Sat  Dinner     4
Sat8139          48.27   6.73    Male     No   Sat  Dinner     4
Thur1025         34.30   6.70    Male     No  Thur   Lunch     6
…                  …    …      …    …      …    …
Sat6983          12.90   1.10  Female    Yes   Sat  Dinner     2
Sat4801           7.25   1.00  Female     No   Sat  Dinner     1
Sat5032          12.60   1.00    Male    Yes   Sat  Dinner     2
Fri3780           5.75   1.00  Female    Yes   Fri  Dinner     2
Sat3455           3.07   1.00  Female    Yes   Sat  Dinner     1

            price_per_person       Payer Name         CC Number   Quality
Payment ID
Sat1954                16.94    Gregory Clark  5473850968388236     Other
Sat4590                12.08  Alex Williamson      676218815212     Other
Sat239                  9.86   Lance Peterson  3542584061609808     Other
Sat8139                12.07      Brian Ortiz  6596453823950595     Other
Thur1025                5.72   Steven Carlson  3526515703718508     Other
…                           …                …                 …        …
Sat6983                 6.45     Jessica Owen     4726904879471     Other
Sat4801                 7.25      Terri Jones  3559221007826887     Other
Sat5032                 6.30    Matthew Myers  3543676378973965     Other
Fri3780                 2.88     Leah Ramirez  3508911676966392     Other
Sat3455                 3.07    Tiffany Brock  4359488526995267  Generous

[243 rows x 11 columns]
```

```
[38]: # to sort according to multiple columns, firstly values will be sorted␣
      ↪according to first column and then will be sorted according to second column

      df.sort_values(['tip', 'size'])
```

```
[38]:             total_bill    tip     sex smoker   day    time  size  \
      Payment ID
      Sat3455           3.07   1.00  Female    Yes   Sat  Dinner     1
      Sat4801           7.25   1.00  Female     No   Sat  Dinner     1
      Fri3780           5.75   1.00  Female    Yes   Fri  Dinner     2
      Sat5032          12.60   1.00    Male    Yes   Sat  Dinner     2
      Sat6983          12.90   1.10  Female    Yes   Sat  Dinner     2
      …                  …  …     …     …  …      …     …
      Thur1025         34.30   6.70    Male     No  Thur   Lunch     6
      Sat8139          48.27   6.73    Male     No   Sat  Dinner     4
      Sat239           39.42   7.58    Male     No   Sat  Dinner     4
      Sat4590          48.33   9.00    Male     No   Sat  Dinner     4
      Sat1954          50.81  10.00    Male    Yes   Sat  Dinner     3

                  price_per_person       Payer Name         CC Number   Quality
      Payment ID
      Sat3455                 3.07    Tiffany Brock  4359488526995267  Generous
      Sat4801                 7.25      Terri Jones  3559221007826887     Other
      Fri3780                 2.88     Leah Ramirez  3508911676966392     Other
      Sat5032                 6.30   Matthew Myers  3543676378973965     Other
      Sat6983                 6.45     Jessica Owen     4726904879471     Other
      …                          …              …               …        …
      Thur1025                5.72   Steven Carlson  3526515703718508     Other
      Sat8139                12.07      Brian Ortiz  6596453823950595     Other
      Sat239                  9.86   Lance Peterson  3542584061609808     Other
      Sat4590                12.08  Alex Williamson      676218815212     Other
      Sat1954                16.94    Gregory Clark  5473850968388236     Other

      [243 rows x 11 columns]
```

```
[39]: # to find the max of a column
      df['tip'].max()
```

```
[39]: 10.0
```

```
[40]: df['tip'].min()
```

```
[40]: 1.0
```

```
[42]: # to get index of the maximum value in a column
      df['tip'].idxmax()
```

```
[42]: 'Sat1954'
```

```
[43]: df['tip'].idxmin()
```

```
[43]: 'Sat3455'
```

```
[44]: # to get row with highest tip value
      df.loc[df['tip'].idxmax()]
```

```
[44]: total_bill                     50.81
      tip                             10.0
      sex                             Male
      smoker                           Yes
      day                              Sat
      time                          Dinner
      size                               3
      price_per_person               16.94
      Payer Name             Gregory Clark
      CC Number           5473850968388236
      Quality                        Other
      Name: Sat1954, dtype: object
```

```
[45]: # to get how much the columns are correlated to each other
      df.corr()
```

```
[45]:                   total_bill       tip      size  price_per_person  CC Number
      total_bill          1.000000  0.675512  0.601708          0.645608   0.109451
      tip                 0.675512  1.000000  0.491509          0.345743   0.119122
      size                0.601708  0.491509  1.000000         -0.173534  -0.027946
      price_per_person    0.645608  0.345743 -0.173534          1.000000   0.138402
      CC Number           0.109451  0.119122 -0.027946          0.138402   1.000000
```

```
[47]: df[['total_bill', 'size']].corr()
```

```
[47]:             total_bill      size
      total_bill    1.000000  0.601708
      size          0.601708  1.000000
```

```
[48]: df['sex'].value_counts()
```

```
[48]: Male      157
      Female     86
      Name: sex, dtype: int64
```

```
[49]: df['sex'].unique()    # All names of unique enteries in column
```

```
[49]: array(['Male', 'Female'], dtype=object)
```

```
[51]: df['tip'].nunique()    # total number of unique enteries in column
```

```
[51]: 121
```

```
[52]: # Alternative of df['tip'].nunique()
      len(df['tip'].unique())
```

```
[52]: 121
```

0.6.1 Replacing values:

```
[53]: # Method 1
      df['sex'].replace('Male', 'M')
```

```
[53]: Payment ID
      Sun4458         M
      Sun5260         M
      Sun2251    Female
      Sun9679         M
      Sun5985         M
                  …
      Sat1766    Female
      Sat3880         M
      Sat17           M
      Thur672    Female
      Sun4458         M
      Name: sex, Length: 243, dtype: object
```

```
[55]: # Method 2

      replace_vals = {'Female': 'F', 'Male': 'M'}
      df['sex'].replace(replace_vals)
```

```
[55]: Payment ID
      Sun4458    M
      Sun5260    M
      Sun2251    F
      Sun9679    M
      Sun5985    M
                ..
      Sat1766    F
      Sat3880    M
      Sat17      M
      Thur672    F
      Sun4458    M
      Name: sex, Length: 243, dtype: object
```

0.6.2 Dealing with duplicated rows:

It return True for all duplicated row and False for unique rows.

```
[56]: df.duplicated()
```

```
[56]: Payment ID
      Sun4458     False
      Sun5260     False
      Sun2251     False
      Sun9679     False
      Sun5985     False
                  …
      Sat1766     False
      Sat3880     False
      Sat17       False
      Thur672     False
      Sun4458      True
      Length: 243, dtype: bool
```

```
[71]: # to drop duplicate rows
      df = df.drop_duplicates()
```

```
[72]: df.duplicated()
```

```
[72]: Payment ID
      Sun4458     False
      Sun5260     False
      Sun2251     False
      Sun9679     False
      Sun5985     False
                  …
      Sat2657     False
      Sat1766     False
      Sat3880     False
      Sat17       False
      Thur672     False
      Length: 242, dtype: bool
```

Filter: To check values in a range, we can use **between method.**

```
[74]: df['total_bill'].between(10,20)
```

```
[74]: Payment ID
      Sun4458     False
      Sun5260     False
      Sun2251     False
      Sun9679     False
```

```
Sun5985    False
           …
Sat2657    False
Sat1766    False
Sat3880    False
Sat17       True
Thur672     True
Name: total_bill, Length: 242, dtype: bool
```

[75]:
```
# to see actual rows with this filter

df[df['total_bill'].between(10,20)]
```

[75]:
```
            total_bill   tip     sex smoker   day    time  size  \
Payment ID
Sun6820          15.04  1.96    Male     No   Sun  Dinner     2
Sun3775          14.78  3.23    Male     No   Sun  Dinner     2
Sun2546          10.27  1.71    Male     No   Sun  Dinner     2
Sun1300          15.42  1.57    Male     No   Sun  Dinner     2
Sun2971          18.43  3.00    Male     No   Sun  Dinner     4
…                    …     …       …    …     …
Sat7220          15.53  3.00    Male    Yes   Sat  Dinner     2
Sat4615          10.07  1.25    Male     No   Sat  Dinner     2
Sat5032          12.60  1.00    Male    Yes   Sat  Dinner     2
Sat17            17.82  1.75    Male     No   Sat  Dinner     2
Thur672          18.78  3.00  Female     No  Thur  Dinner     2

            price_per_person       Payer Name         CC Number Quality
Payment ID
Sun6820                 7.52  Joseph Mcdonald  3522866365840377   Other
Sun3775                 7.39    Jerome Abbott  3532124519049786   Other
Sun2546                 5.14    William Riley      566287581219   Other
Sun1300                 7.71  Chad Harrington      577040572932   Other
Sun2971                 4.61     Joshua Jones  6011163105616890   Other
…                          …              …                 …       …
Sat7220                 7.76    Tracy Douglas  4097938155941930   Other
Sat4615                 5.04    Sean Gonzalez  3534021246117605   Other
Sat5032                 6.30    Matthew Myers  3543676378973965   Other
Sat17                   8.91     Dennis Dixon     4375220550950   Other
Thur672                 9.39  Michelle Hardin  3511451626698139   Other

[128 rows x 11 columns]
```

To check largest 10 rows, use **nlargest()** method. It has **descending = True**.

[78]:
```
df.nlargest(10, 'tip')
```

```
[78]:             total_bill    tip     sex smoker   day    time  size  \
      Payment ID
      Sat1954          50.81  10.00    Male    Yes   Sat  Dinner     3
      Sat4590          48.33   9.00    Male     No   Sat  Dinner     4
      Sat239           39.42   7.58    Male     No   Sat  Dinner     4
      Sat8139          48.27   6.73    Male     No   Sat  Dinner     4
      Thur1025         34.30   6.70    Male     No  Thur   Lunch     6
      Sun6059          23.17   6.50    Male    Yes   Sun  Dinner     4
      Sat3374          28.17   6.50  Female    Yes   Sat  Dinner     3
      Sun9677          32.40   6.00    Male     No   Sun  Dinner     4
      Sat2657          29.03   5.92    Male     No   Sat  Dinner     3
      Thur9003         24.71   5.85    Male     No  Thur   Lunch     2

                  price_per_person         Payer Name         CC Number   Quality
      Payment ID
      Sat1954                16.94      Gregory Clark  5473850968388236     Other
      Sat4590                12.08    Alex Williamson      676218815212     Other
      Sat239                  9.86     Lance Peterson  3542584061609808     Other
      Sat8139                12.07        Brian Ortiz  6596453823950595     Other
      Thur1025                5.72    Steven Carlson  3526515703718508     Other
      Sun6059                 5.79  Dr. Michael James     4718501859162  Generous
      Sat3374                 9.39    Marissa Jackson  4922302538691962     Other
      Sun9677                 8.10       James Barnes  3552002592874186     Other
      Sat2657                 9.68      Michael Avila  5296068606052842     Other
      Thur9003               12.36       Roger Taylor     4410248629955     Other
```

```
[79]: df.nsmallest(10, 'tip')
```

```
[79]:             total_bill   tip     sex smoker   day    time  size  \
      Payment ID
      Sat3455           3.07  1.00  Female    Yes   Sat  Dinner     1
      Fri3780           5.75  1.00  Female    Yes   Fri  Dinner     2
      Sat4801           7.25  1.00  Female     No   Sat  Dinner     1
      Sat5032          12.60  1.00    Male    Yes   Sat  Dinner     2
      Sat6983          12.90  1.10  Female    Yes   Sat  Dinner     2
      Sat2929          32.83  1.17    Male    Yes   Sat  Dinner     2
      Sat5056          10.51  1.25    Male     No   Sat  Dinner     2
      Thur6600          8.51  1.25  Female     No  Thur   Lunch     2
      Sat4615          10.07  1.25    Male     No   Sat  Dinner     2
      Sun3279           9.68  1.32    Male     No   Sun  Dinner     2

                  price_per_person      Payer Name         CC Number   Quality
      Payment ID
      Sat3455                 3.07  Tiffany Brock  4359488526995267  Generous
      Fri3780                 2.88   Leah Ramirez  3508911676966392     Other
      Sat4801                 7.25    Terri Jones  3559221007826887     Other
      Sat5032                 6.30  Matthew Myers  3543676378973965     Other
```

```
Sat6983                   6.45        Jessica Owen     4726904879471     Other
Sat2929                  16.42        Thomas Brown  4284722681265508     Other
Sat5056                   5.26       Kenneth Hayes   213142079731108     Other
Thur6600                  4.26      Rebecca Harris  4320272020376174     Other
Sat4615                   5.04       Sean Gonzalez  3534021246117605     Other
Sun3279                   4.84  Christopher Spears  4387671121369212     Other
```

[80]:
```
# to get random 5 rows

df.sample(5)
```

[80]:
```
            total_bill   tip   sex smoker  day    time  size  \
Payment ID
Sat3697          21.70  4.30  Male     No  Sat  Dinner     2
Sun9209           7.25  5.15  Male    Yes  Sun  Dinner     2
Sun9774          18.04  3.00  Male     No  Sun  Dinner     2
Fri5959          13.42  1.58  Male    Yes  Fri   Lunch     2
Sun5205          17.26  2.74  Male     No  Sun  Dinner     3

            price_per_person         Payer Name         CC Number   Quality
Payment ID
Sat3697                10.85       David Collier  5529694315416009     Other
Sun9209                 3.62         Larry White    30432617123103  Generous
Sun9774                 9.02        William Roth  6573923967142503     Other
Fri5959                 6.71  Ronald Vaughn DVM   341503466406403     Other
Sun5205                 5.75       Gregory Smith     4292362333741     Other
```

[81]:
```
# to get 10% rows of dataframe

df.sample(frac=0.1)
```

[81]:
```
            total_bill   tip     sex smoker  day    time  size  \
Payment ID
Sun2127          13.13  2.00    Male     No  Sun  Dinner     2
Sun5260          23.68  3.31    Male     No  Sun  Dinner     2
Sat5056          10.51  1.25    Male     No  Sat  Dinner     2
Sat6240          44.30  2.50  Female    Yes  Sat  Dinner     3
Fri5700          22.49  3.50    Male     No  Fri  Dinner     2
Sun444           17.51  3.00  Female    Yes  Sun  Dinner     2
Sat8124          17.78  3.27    Male     No  Sat  Dinner     2
Sat6983          12.90  1.10  Female    Yes  Sat  Dinner     2
Sun5814          19.77  2.00    Male     No  Sun  Dinner     4
Sat9213          20.65  3.35    Male     No  Sat  Dinner     3
Sat4615          10.07  1.25    Male     No  Sat  Dinner     2
Sun1878          21.58  3.92    Male     No  Sun  Dinner     2
Sat4319          20.45  3.00    Male     No  Sat  Dinner     4
Sat8980          26.41  1.50  Female     No  Sat  Dinner     2
```

```
Thur6321          7.51  2.00    Male     No  Thur   Lunch     2
Sat6406          16.93  3.07  Female     No   Sat  Dinner     3
Sat1967          26.86  3.14  Female    Yes   Sat  Dinner     2
Sun4598           9.60  4.00  Female    Yes   Sun  Dinner     2
Sun9470          25.56  4.34    Male     No   Sun  Dinner     4
Sun2971          18.43  3.00    Male     No   Sun  Dinner     4
Thur6048         18.71  4.00    Male    Yes  Thur   Lunch     3
Sun4561           9.94  1.56    Male     No   Sun  Dinner     2
Fri3780           5.75  1.00  Female    Yes   Fri  Dinner     2
Sat6801          14.00  3.00    Male     No   Sat  Dinner     2

            price_per_person          Payer Name         CC Number   Quality
Payment ID
Sun2127                 6.56        Jason Arnold  3571825125296106     Other
Sun5260                11.84    Nathaniel Harris  4676137647685994     Other
Sat5056                 5.26       Kenneth Hayes   213142079731108     Other
Sat6240                14.77       Heather Cohen   379771118886604     Other
Fri5700                11.24           Earl Horn  6011849326227398     Other
Sun444                  8.76      Audrey Griffin  3500853929693258     Other
Sat8124                 8.89      Jacob Castillo  3551492000704805     Other
Sat6983                 6.45        Jessica Owen     4726904879471     Other
Sun5814                 4.94         James Smith   213169731428229     Other
Sat9213                 6.88       Timothy Oneal  6568069240986485     Other
Sat4615                 5.04       Sean Gonzalez  3534021246117605     Other
Sun1878                10.79      Matthew Reilly   180073029785069     Other
Sat4319                 5.11      Robert Bradley   213141668145910     Other
Sat8980                13.20        Melody Simon  4745394421258160     Other
Thur6321                3.76      Daniel Robbins  4823139288341889  Generous
Sat6406                 5.64          Erin Lewis  5161695527390786     Other
Sat1967                13.43  Victoria Obrien MD     4216245673726     Other
Sun4598                 4.80        Melanie Gray     4211808859168  Generous
Sun9470                 6.39        Ronald Owens  6569607991983380     Other
Sun2971                 4.61        Joshua Jones  6011163105616890     Other
Thur6048                6.24        Jason Conrad     4581233003487     Other
Sun4561                 4.97        Curtis Morgan  4628628020417301     Other
Fri3780                 2.88        Leah Ramirez  3508911676966392     Other
Sat6801                 7.00       James Sanchez   345243048851323     Other
```