```
In [28]:   import pandas as pd

           meteorites = pd.read_csv('Meteorite_Landings.csv', nrows=5)
           meteorites
```

Out[28]:

	name	id	nametype	recclass	mass (g)	fall	year	reclat	reclong	
0	Aachen	1	Valid	L5	21	Fell	01/01/1880 12:00:00 AM	50.77500	6.08333	
1	Aarhus	2	Valid	H6	720	Fell	01/01/1951 12:00:00 AM	56.18333	10.23333	
2	Abee	6	Valid	EH4	107000	Fell	01/01/1952 12:00:00 AM	54.21667	-113.00000	
3	Acapulco	10	Valid	Acapulcoite	1914	Fell	01/01/1976 12:00:00 AM	16.88333	-99.90000	
4	Achiras	370	Valid	L6	780	Fell	01/01/1902 12:00:00 AM	-33.16667	-64.95000	

```
In [3]:   meteorites.name
```

Out[3]:   0   Aachen
          1   Aarhus
          2   Abee
          3   Acapulco
          4   Achiras
          Name: name, dtype: object

```
In [4]:   meteorites.columns
```

Out[4]:   Index(['name', 'id', 'nametype', 'recclass', 'mass (g)', 'fall', 'year',
                 'reclat', 'reclong', 'GeoLocation'],
                dtype='object')

```
In [5]:   meteorites.index
```

Out[5]:   RangeIndex(start=0, stop=5, step=1)

```
In [21]:   import requests

           response = requests.get(
               'https://data.nasa.gov/resource/gh4g-9sfh.json',
                params={'$limit':50_000}
           )
```

```python
if response.ok:
    payload = response.json()
else:
    print(f'Request was not succesful and returned code: {response.status_code}')
    payload = None
```

In [ ]:
```python
payload
```

In [24]:
```python
df = pd.DataFrame(payload)
df.head(3)
```

Out[24]:

	name	id	nametype	recclass	mass	fall	year	reclat	reclong	g
0	Aachen	1	Valid	L5	21	Fell	1880-01-01T00:00:00.000	50.775000	6.083330	
1	Aarhus	2	Valid	H6	720	Fell	1951-01-01T00:00:00.000	56.183330	10.233330	
2	Abee	6	Valid	EH4	107000	Fell	1952-01-01T00:00:00.000	54.216670	-113.000000	

In [36]:
```python
meteorites = pd.read_csv('Meteorite_Landings.csv')
meteorites
```

Out[36]:

	name	id	nametype	recclass	mass (g)	fall	year	reclat
0	Aachen	1	Valid	L5	21.0	Fell	01/01/1880 12:00:00 AM	50.77500
1	Aarhus	2	Valid	H6	720.0	Fell	01/01/1951 12:00:00 AM	56.18333
2	Abee	6	Valid	EH4	107000.0	Fell	01/01/1952 12:00:00 AM	54.21667
3	Acapulco	10	Valid	Acapulcoite	1914.0	Fell	01/01/1976 12:00:00 AM	16.88333
4	Achiras	370	Valid	L6	780.0	Fell	01/01/1902 12:00:00 AM	-33.16667
...	...	...	...	...	...	...	...	...
45711	Zillah 002	31356	Valid	Eucrite	172.0	Found	01/01/1990 12:00:00 AM	29.03700
45712	Zinder	30409	Valid	Pallasite, ungrouped	46.0	Found	01/01/1999 12:00:00 AM	13.78333
45713	Zlin	30410	Valid	H4	3.3	Found	01/01/1939 12:00:00 AM	49.25000
45714	Zubkovsky	31357	Valid	L6	2167.0	Found	01/01/2003 12:00:00 AM	49.78917
45715	Zulu Queen	30414	Valid	L3.7	200.0	Found	01/01/1976 12:00:00 AM	33.98333

45716 rows × 10 columns

◀ ▬▬▬▬▬▬▬▬▬▬▬▬▬▬ ▶

In [37]: `meteorites.shape`

Out[37]: (45716, 10)

In [26]: `meteorites.columns`

Out[26]: Index(['name', 'id', 'nametype', 'recclass', 'mass (g)', 'fall', 'year',
       'reclat', 'reclong', 'GeoLocation'],
      dtype='object')

```
In [27]: meteorites.dtypes
```

```
Out[27]: name            object
         id               int64
         nametype        object
         recclass        object
         mass (g)         int64
         fall            object
         year            object
         reclat         float64
         reclong        float64
         GeoLocation     object
         dtype: object
```

```
In [40]: meteorites.head(10)
```

Out[40]:

	name	id	nametype	recclass	mass (g)	fall	year	reclat	reclong
0	Aachen	1	Valid	L5	21.0	Fell	01/01/1880 12:00:00 AM	50.77500	6.08333
1	Aarhus	2	Valid	H6	720.0	Fell	01/01/1951 12:00:00 AM	56.18333	10.23333
2	Abee	6	Valid	EH4	107000.0	Fell	01/01/1952 12:00:00 AM	54.21667	-113.00000
3	Acapulco	10	Valid	Acapulcoite	1914.0	Fell	01/01/1976 12:00:00 AM	16.88333	-99.90000
4	Achiras	370	Valid	L6	780.0	Fell	01/01/1902 12:00:00 AM	-33.16667	-64.95000
5	Adhi Kot	379	Valid	EH4	4239.0	Fell	01/01/1919 12:00:00 AM	32.10000	71.80000
6	Adzhi-Bogdo (stone)	390	Valid	LL3-6	910.0	Fell	01/01/1949 12:00:00 AM	44.83333	95.16667
7	Agen	392	Valid	H5	30000.0	Fell	01/01/1814 12:00:00 AM	44.21667	0.61667
8	Aguada	398	Valid	L6	1620.0	Fell	01/01/1930 12:00:00 AM	-31.60000	-65.23333
9	Aguila Blanca	417	Valid	L	1440.0	Fell	01/01/1920 12:00:00 AM	-30.86667	-64.55000

In [39]: `meteorites.tail()`

	name	id	nametype	recclass	mass (g)	fall	year	reclat	r
45711	Zillah 002	31356	Valid	Eucrite	172.0	Found	01/01/1990 12:00:00 AM	29.03700	17
45712	Zinder	30409	Valid	Pallasite, ungrouped	46.0	Found	01/01/1999 12:00:00 AM	13.78333	8
45713	Zlin	30410	Valid	H4	3.3	Found	01/01/1939 12:00:00 AM	49.25000	17
45714	Zubkovsky	31357	Valid	L6	2167.0	Found	01/01/2003 12:00:00 AM	49.78917	41
45715	Zulu Queen	30414	Valid	L3.7	200.0	Found	01/01/1976 12:00:00 AM	33.98333	-115

In [41]:

```python
meteorites.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 45716 entries, 0 to 45715
Data columns (total 10 columns):
 #   Column       Non-Null Count  Dtype
---  ------       --------------  -----
 0   name         45716 non-null  object
 1   id           45716 non-null  int64
 2   nametype     45716 non-null  object
 3   recclass     45716 non-null  object
 4   mass (g)     45585 non-null  float64
 5   fall         45716 non-null  object
 6   year         45425 non-null  object
 7   reclat       38401 non-null  float64
 8   reclong      38401 non-null  float64
 9   GeoLocation  38401 non-null  object
dtypes: float64(3), int64(1), object(6)
memory usage: 3.5+ MB
```

In [46]:

```python
meteorites[['name','GeoLocation']]
```

	name	GeoLocation
0	Aachen	(50.775, 6.08333)
1	Aarhus	(56.18333, 10.23333)
2	Abee	(54.21667, -113.0)
3	Acapulco	(16.88333, -99.9)
4	Achiras	(-33.16667, -64.95)
...	...	...
45711	Zillah 002	(29.037, 17.0185)
45712	Zinder	(13.78333, 8.96667)
45713	Zlin	(49.25, 17.66667)
45714	Zubkovsky	(49.78917, 41.5046)
45715	Zulu Queen	(33.98333, -115.68333)

45716 rows × 2 columns

In [47]: `meteorites[100:104]`

	name	id	nametype	recclass	mass (g)	fall	year	reclat	reclon
100	Benton	5026	Valid	LL6	2840.0	Fell	01/01/1949 12:00:00 AM	45.95000	-67.5500
101	Berduc	48975	Valid	L6	270.0	Fell	01/01/2008 12:00:00 AM	-31.91000	-58.3283
102	Béréba	5028	Valid	Eucrite-mmict	18000.0	Fell	01/01/1924 12:00:00 AM	11.65000	-3.6500
103	Berlanguillas	5029	Valid	L6	1440.0	Fell	01/01/1811 12:00:00 AM	41.68333	-3.8000

◀ ━━━━━━━━━━━━━━━ ▶

In [51]: `meteorites.iloc[100:104, [0,3,4,6]]`

Out[51]:

	name	recclass	mass (g)	year
100	Benton	LL6	2840.0	01/01/1949 12:00:00 AM
101	Berduc	L6	270.0	01/01/2008 12:00:00 AM
102	Béréba	Eucrite-mmict	18000.0	01/01/1924 12:00:00 AM
103	Berlanguillas	L6	1440.0	01/01/1811 12:00:00 AM

In [50]:
```python
meteorites.loc[100:104, 'mass (g)':'year']
```

Out[50]:

	mass (g)	fall	year
100	2840.0	Fell	01/01/1949 12:00:00 AM
101	270.0	Fell	01/01/2008 12:00:00 AM
102	18000.0	Fell	01/01/1924 12:00:00 AM
103	1440.0	Fell	01/01/1811 12:00:00 AM
104	960.0	Fell	01/01/2004 12:00:00 AM

In [53]:
```python
meteorites.loc[100:104, 'mass (g)']
```

Out[53]:
```
100     2840.0
101      270.0
102    18000.0
103     1440.0
104      960.0
Name: mass (g), dtype: float64
```

In [55]:
```python
meteorites.iloc[-1, [9]]
```

Out[55]:
```
GeoLocation    (33.98333, -115.68333)
Name: 45715, dtype: object
```

In [56]:
```python
(meteorites['mass (g)'] >50) & (meteorites.fall == 'Found')
```

Out[56]:
```
0        False
1        False
2        False
3        False
4        False
         ...
45711     True
45712    False
45713    False
45714     True
45715     True
Length: 45716, dtype: bool
```

In [57]:
```python
meteorites[(meteorites['mass (g)'] > 1e6) & (meteorites.fall =='Fell')]
```

`Out[57]:`

	name	id	nametype	recclass	mass (g)	fall	year	reclat	reclong
29	Allende	2278	Valid	CV3	2000000.0	Fell	01/01/1969 12:00:00 AM	26.96667	-105.3166
419	Jilin	12171	Valid	H5	4000000.0	Fell	01/01/1976 12:00:00 AM	44.05000	126.1666
506	Kunya-Urgench	12379	Valid	H5	1100000.0	Fell	01/01/1998 12:00:00 AM	42.25000	59.2000
707	Norton County	17922	Valid	Aubrite	1100000.0	Fell	01/01/1948 12:00:00 AM	39.68333	-99.8666
920	Sikhote-Alin	23593	Valid	Iron, IIAB	23000000.0	Fell	01/01/1947 12:00:00 AM	46.16000	134.6533

◀ ▭▭▭▭▭▭▭▭▭▭▭▭▭ ▶

`In [59]: meteorites.query("`mass (g)` > 1e6 and fall =='Fell'")`

`Out[59]:`

	name	id	nametype	recclass	mass (g)	fall	year	reclat	reclong
29	Allende	2278	Valid	CV3	2000000.0	Fell	01/01/1969 12:00:00 AM	26.96667	-105.3166
419	Jilin	12171	Valid	H5	4000000.0	Fell	01/01/1976 12:00:00 AM	44.05000	126.1666
506	Kunya-Urgench	12379	Valid	H5	1100000.0	Fell	01/01/1998 12:00:00 AM	42.25000	59.2000
707	Norton County	17922	Valid	Aubrite	1100000.0	Fell	01/01/1948 12:00:00 AM	39.68333	-99.8666
920	Sikhote-Alin	23593	Valid	Iron, IIAB	23000000.0	Fell	01/01/1947 12:00:00 AM	46.16000	134.6533

◀ ▭▭▭▭▭▭▭▭▭▭▭▭▭ ▶

`In [60]: meteorites.fall.value_counts()`

```
Out[60]: fall
         Found    44609
         Fell      1107
         Name: count, dtype: int64
```

`In [61]: meteorites.value_counts(subset=['nametype', 'fall'],normalize=True)`

```
Out[61]:  nametype  fall
          Valid     Found     0.974145
                    Fell      0.024215
          Relict    Found     0.001641
          Name: proportion, dtype: float64
```

```
In [62]:  meteorites.value_counts(subset=['nametype', 'fall'],normalize=False)
```

```
Out[62]:  nametype  fall
          Valid     Found     44534
                    Fell       1107
          Relict    Found        75
          Name: count, dtype: int64
```

```
In [66]:  flot = meteorites['mass (g)'].mean()

          print(float(flot))
```

```
          13278.078548601512
```

```
In [67]:  meteorites['mass (g)'].quantile([0.01,0.05,0.5,0.95,0.99])
```

```
Out[67]:  0.01         0.44
          0.05         1.10
          0.50        32.60
          0.95      4000.00
          0.99     50600.00
          Name: mass (g), dtype: float64
```

```
In [68]:  meteorites['mass (g)'].median()
```

```
Out[68]:  32.6
```

```
In [69]:  meteorites['mass (g)'].max()
```

```
Out[69]:  60000000.0
```

```
In [72]:  meteorites.loc[meteorites['mass (g)'].idxmax()]
```

```
Out[72]:  name                           Hoba
          id                            11890
          nametype                      Valid
          recclass                   Iron, IVB
          mass (g)                 60000000.0
          fall                          Found
          year          01/01/1920 12:00:00 AM
          reclat                     -19.58333
          reclong                     17.91667
          GeoLocation     (-19.58333, 17.91667)
          Name: 16392, dtype: object
```

```
In [73]:  meteorites.recclass.nunique()
```

```
Out[73]:  466
```

```
In [75]: meteorites.name.nunique()
```

```
Out[75]: 45716
```

```
In [74]: meteorites.recclass.unique()[:14]
```

```
Out[74]: array(['L5', 'H6', 'EH4', 'Acapulcoite', 'L6', 'LL3-6', 'H5', 'L',
               'Diogenite-pm', 'Unknown', 'H4', 'H', 'Iron, IVA', 'CR2-an'],
              dtype=object)
```

```
In [77]: meteorites.describe()
```

Out[77]:

	id	mass (g)	reclat	reclong
count	45716.000000	4.558500e+04	38401.000000	38401.000000
mean	26889.735104	1.327808e+04	-39.122580	61.074319
std	16860.683030	5.749889e+05	46.378511	80.647298
min	1.000000	0.000000e+00	-87.366670	-165.433330
25%	12688.750000	7.200000e+00	-76.714240	0.000000
50%	24261.500000	3.260000e+01	-71.500000	35.666670
75%	40656.750000	2.026000e+02	0.000000	157.166670
max	57458.000000	6.000000e+07	81.166670	354.473330

```
In [78]: meteorites.describe(include='all')
```

	name	id	nametype	recclass	mass (g)	fall	year	
count	45716	45716.000000	45716	45716	4.558500e+04	45716	45425	3840
unique	45716	NaN	2	466	NaN	2	266	
top	Aachen	NaN	Valid	L6	NaN	Found	01/01/2003 12:00:00 AM	
freq	1	NaN	45641	8285	NaN	44609	3323	
mean	NaN	26889.735104	NaN	NaN	1.327808e+04	NaN	NaN	-39
std	NaN	16860.683030	NaN	NaN	5.749889e+05	NaN	NaN	46
min	NaN	1.000000	NaN	NaN	0.000000e+00	NaN	NaN	-87
25%	NaN	12688.750000	NaN	NaN	7.200000e+00	NaN	NaN	-76
50%	NaN	24261.500000	NaN	NaN	3.260000e+01	NaN	NaN	-71
75%	NaN	40656.750000	NaN	NaN	2.026000e+02	NaN	NaN	0
max	NaN	57458.000000	NaN	NaN	6.000000e+07	NaN	NaN	81

Exercise (Part 1)

Using the 2019_Yellow_Taxi_Trip_Data.csv dataset, accomplish the following items and submit a PDF of the notebook:

1.

Create a DataFrame by reading in the 2019_Yellow_Taxi_Trip_Data.csv file. Examine the first 5 rows2.. Find the dimensions (number of rows and number of columns) in the dat3.a. Using the data in the 2019_Yellow_Taxi_Trip_Data.csv file, calculate summary statistics for the fare_amount, tip_amount, tolls_amount, and total_amount colum4.ns. Isolate the fare_amount, tip_amount, tolls_amount, and total_amount for the longest trip by distance (trip_distance).

In [3]:
```python
import pandas as pd

YellowTaxi = pd.read_csv('2019_Yellow_Taxi_Trip_Data.csv')
YellowTaxi.head()
```

Out[3]:

	vendorid	tpep_pickup_datetime	tpep_dropoff_datetime	passenger_count	trip_distance
0	2	2019-10-23T16:39:42.000	2019-10-23T17:14:10.000	1	7.93
1	1	2019-10-23T16:32:08.000	2019-10-23T16:45:26.000	1	2.00
2	2	2019-10-23T16:08:44.000	2019-10-23T16:21:11.000	1	1.36
3	2	2019-10-23T16:22:44.000	2019-10-23T16:43:26.000	1	1.00
4	2	2019-10-23T16:45:11.000	2019-10-23T16:58:49.000	1	1.96

In [83]: `YellowTaxi.shape`

Out[83]: `(10000, 18)`

In [97]: `YellowTaxi[['fare_amount', 'tip_amount', 'tolls_amount', 'total_amount']].describe(`

Out[97]:

	fare_amount	tip_amount	tolls_amount	total_amount
count	10000.000000	10000.000000	10000.000000	10000.000000
mean	15.106313	2.634494	0.623447	22.564659
std	13.954762	3.409800	6.437507	19.209255
min	-52.000000	0.000000	-6.120000	-65.920000
25%	7.000000	0.000000	0.000000	12.375000
50%	10.000000	2.000000	0.000000	16.300000
75%	16.000000	3.250000	0.000000	22.880000
max	176.000000	43.000000	612.000000	671.800000

In [98]: `YellowTaxi[['fare_amount', 'tip_amount', 'tolls_amount', 'total_amount']].mean()`

Out[98]:
```
fare_amount      15.106313
tip_amount        2.634494
tolls_amount      0.623447
total_amount     22.564659
dtype: float64
```

In [99]: `YellowTaxi[['fare_amount', 'tip_amount', 'tolls_amount', 'total_amount']].median()`

```
fare_amount      10.0
tip_amount        2.0
tolls_amount      0.0
total_amount     16.3
dtype: float64
```

```
YellowTaxi[['fare_amount', 'tip_amount', 'tolls_amount', 'total_amount']].quantile(
```

	fare_amount	tip_amount	tolls_amount	total_amount
0.01	3.000	0.000	0.00	6.3000
0.05	4.500	0.000	0.00	9.3000
0.50	10.000	2.000	0.00	16.3000
0.95	52.000	10.361	6.12	67.1075
0.99	62.005	15.860	6.12	82.4000

```
YellowTaxi[['fare_amount', 'tip_amount', 'tolls_amount', 'total_amount']].max()
```

```
fare_amount      176.0
tip_amount        43.0
tolls_amount     612.0
total_amount     671.8
dtype: float64
```

```
YellowTaxi.trip_distance
```

```
0        7.93
1        2.00
2        1.36
3        1.00
4        1.96
         ...
9995     1.30
9996     1.40
9997     0.70
9998     2.50
9999     3.00
Name: trip_distance, Length: 10000, dtype: float64
```

```
Distance = YellowTaxi.iloc[:,[10,13,14,16]]
Distance.describe()
```

	fare_amount	tip_amount	tolls_amount	total_amount
count	10000.000000	10000.000000	10000.000000	10000.000000
mean	15.106313	2.634494	0.623447	22.564659
std	13.954762	3.409800	6.437507	19.209255
min	-52.000000	0.000000	-6.120000	-65.920000
25%	7.000000	0.000000	0.000000	12.375000
50%	10.000000	2.000000	0.000000	16.300000
75%	16.000000	3.250000	0.000000	22.880000
max	176.000000	43.000000	612.000000	671.800000

In [109...
```python
Distance.loc[YellowTaxi["trip_distance"].idxmax()]
```

Out[109...
```
fare_amount      176.00
tip_amount        18.29
tolls_amount       6.12
total_amount     201.21
Name: 8338, dtype: float64
```

Introducing Pandas has refreshed my memory of my last sub VDA and this has taught me more and has helped me access data and how to put a play on it, even though there were hard parts like the question 4 in exercise 1, but did it!

- _____

  *

◀ ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ ▶

START OF SECOND SESSION !!!

In [23]:
```python
taxis = pd.read_csv('2019_Yellow_Taxi_Trip_Data.csv')
taxis.head()
```

Out[23]:

	vendorid	tpep_pickup_datetime	tpep_dropoff_datetime	passenger_count	trip_distance
0	2	2019-10-23T16:39:42.000	2019-10-23T17:14:10.000	1	7.93
1	1	2019-10-23T16:32:08.000	2019-10-23T16:45:26.000	1	2.00
2	2	2019-10-23T16:08:44.000	2019-10-23T16:21:11.000	1	1.36
3	2	2019-10-23T16:22:44.000	2019-10-23T16:43:26.000	1	1.00
4	2	2019-10-23T16:45:11.000	2019-10-23T16:58:49.000	1	1.96

```python
In [25]: masks = taxis.columns.str.contains('id$|store_and_fwd_flag', regex=True)
         columns_to_drop = taxis.columns[masks]
         columns_to_drop
```

Out[25]: Index(['vendorid', 'ratecodeid', 'store_and_fwd_flag', 'pulocationid',
               'dolocationid'],
              dtype='object')

```python
In [27]: taxis = YellowTaxi.drop(columns = columns_to_drop)
         taxis.head()
```

Out[27]:

	tpep_pickup_datetime	tpep_dropoff_datetime	passenger_count	trip_distance	payment_t
0	2019-10-23T16:39:42.000	2019-10-23T17:14:10.000	1	7.93	
1	2019-10-23T16:32:08.000	2019-10-23T16:45:26.000	1	2.00	
2	2019-10-23T16:08:44.000	2019-10-23T16:21:11.000	1	1.36	
3	2019-10-23T16:22:44.000	2019-10-23T16:43:26.000	1	1.00	
4	2019-10-23T16:45:11.000	2019-10-23T16:58:49.000	1	1.96	

```python
In [34]: taxis = taxis.rename(
             columns={
                 'tpep_pickup_datetime': 'pickup',
                 'tpep_dropoff_datetime': 'dropoff'
             }
         )
         taxis.columns
```

```
Out[34]: Index(['pickup', 'dropoff', 'passenger_count', 'trip_distance', 'payment_type',
                'fare_amount', 'extra', 'mta_tax', 'tip_amount', 'tolls_amount',
                'improvement_surcharge', 'total_amount', 'congestion_surcharge'],
               dtype='object')
```

In [33]: `taxis.dtypes`

```
Out[33]: pickup                    object
         dropoff                   object
         passenger_count            int64
         trip_distance            float64
         payment_type               int64
         fare_amount              float64
         extra                    float64
         mta_tax                  float64
         tip_amount               float64
         tolls_amount             float64
         improvement_surcharge    float64
         total_amount             float64
         congestion_surcharge     float64
         dtype: object
```

In [37]:
```python
taxis[['pickup', 'dropoff']] = taxis[['pickup','dropoff']].apply(pd.to_datetime)
taxis.dtypes
```

```
Out[37]: pickup                   datetime64[ns]
         dropoff                  datetime64[ns]
         passenger_count                   int64
         trip_distance                   float64
         payment_type                      int64
         fare_amount                     float64
         extra                           float64
         mta_tax                         float64
         tip_amount                      float64
         tolls_amount                    float64
         improvement_surcharge           float64
         total_amount                    float64
         congestion_surcharge            float64
         dtype: object
```

CREATING COLUMNS

In [43]:
```python
taxis = taxis.assign(
    elapsed_time=lambda x: x.dropoff - x.pickup, # 1
    cost_before_tip=lambda x: x.total_amount - x.tip_amount,
    tip_pct=lambda x: x.tip_amount / x.cost_before_tip, #2
    fees=lambda x: x.cost_before_tip - x.fare_amount, #3
    avg_speed=lambda x: x.trip_distance.div(
        x.elapsed_time.dt.total_seconds() / 60 / 60
    )
)
taxis.dtypes
```

```
Out[43]: pickup                    datetime64[ns]
         dropoff                   datetime64[ns]
         passenger_count                    int64
         trip_distance                    float64
         payment_type                       int64
         fare_amount                      float64
         extra                            float64
         mta_tax                          float64
         tip_amount                       float64
         tolls_amount                     float64
         improvement_surcharge            float64
         total_amount                     float64
         congestion_surcharge             float64
         elapsed_time              timedelta64[ns]
         cost_before_tip                  float64
         tip_pct                          float64
         fees                             float64
         avg_speed                        float64
         dtype: object
```

In [44]: `taxis.head(2)`

Out[44]:

	pickup	dropoff	passenger_count	trip_distance	payment_type	fare_amount	extra	m
0	2019-10-23 16:39:42	2019-10-23 17:14:10	1	7.93	1	29.5	1.0	
1	2019-10-23 16:32:08	2019-10-23 16:45:26	1	2.00	1	10.5	1.0	

In [47]: `taxis.sort_values(['dropoff'], ascending=[False]).head(2)`

Out[47]:

	pickup	dropoff	passenger_count	trip_distance	payment_type	fare_amount	extra
9183	2019-10-23 17:19:31	2019-10-24 17:15:47	2	7.48	2	29.0	1.0
7576	2019-10-23 16:52:51	2019-10-24 16:51:44	1	3.75	1	17.5	1.0

In [52]: `taxis.sort_values(['dropoff','payment_type'], ascending=[False,True]).head(5)`

Out[52]:

	pickup	dropoff	passenger_count	trip_distance	payment_type	fare_amount	extra
9183	2019-10-23 17:19:31	2019-10-24 17:15:47	2	7.48	2	29.0	1.0
7576	2019-10-23 16:52:51	2019-10-24 16:51:44	1	3.75	1	17.5	1.0
6902	2019-10-23 16:51:42	2019-10-24 16:50:22	1	11.19	2	39.5	1.0
6550	2019-10-23 16:49:36	2019-10-24 16:47:40	1	2.54	1	11.0	1.0
5907	2019-10-23 16:49:40	2019-10-24 16:46:42	1	3.55	2	15.5	1.0

◄ ▬▬▬▬▬▬▬ ►

In [56]:
```python
taxis.nlargest(3, 'trip_distance')
```

Out[56]:

	pickup	dropoff	passenger_count	trip_distance	payment_type	fare_amount	extra
8338	2019-10-23 16:50:53	2019-10-24 15:32:55	1	38.11	1	176.0	0.0
9965	2019-10-23 17:34:29	2019-10-23 18:48:00	1	37.86	2	52.0	4.5
1656	2019-10-23 16:04:45	2019-10-23 19:11:40	3	37.57	1	52.0	4.5

◄ ▬▬▬▬▬▬▬ ►

EXERCISE (PART 2)

Read in the meteorite data from the meteorite)landing.csv file, rename the mass (g) column to mass, and drop all the latitude and longitude columns. Sort the result by mass in descending order

In [99]:
```python
meteorites = pd.read_csv('Meteorite_Landings.csv', nrows=5)
meteorites.head()
```

	name	id	nametype	recclass	mass (g)	fall	year	reclat	reclong
0	Aachen	1	Valid	L5	21	Fell	01/01/1880 12:00:00 AM	50.77500	6.08333
1	Aarhus	2	Valid	H6	720	Fell	01/01/1951 12:00:00 AM	56.18333	10.23333
2	Abee	6	Valid	EH4	107000	Fell	01/01/1952 12:00:00 AM	54.21667	-113.00000
3	Acapulco	10	Valid	Acapulcoite	1914	Fell	01/01/1976 12:00:00 AM	16.88333	-99.90000
4	Achiras	370	Valid	L6	780	Fell	01/01/1902 12:00:00 AM	-33.16667	-64.95000

In [100…
```python
meteorites = pd.read_csv('Meteorite_Landings.csv', nrows=5)

meteorites = meteorites.rename(
    columns={ 'mass (g)': 'mass'} )
meteorites.columns
```

Out[100…
```
Index(['name', 'id', 'nametype', 'recclass', 'mass', 'fall', 'year', 'reclat',
       'reclong', 'GeoLocation'],
      dtype='object')
```

In [101…
```python
meteor = meteorites.columns.str.contains('reclat|reclong', regex=[True,True])
columns_to_drop = meteorites.columns[meteor]
columns_to_drop

meteorites = meteorites.drop(columns = columns_to_drop)
meteorites.head()
```

	name	id	nametype	recclass	mass	fall	year	GeoLocation
0	Aachen	1	Valid	L5	21	Fell	01/01/1880 12:00:00 AM	(50.775, 6.08333)
1	Aarhus	2	Valid	H6	720	Fell	01/01/1951 12:00:00 AM	(56.18333, 10.23333)
2	Abee	6	Valid	EH4	107000	Fell	01/01/1952 12:00:00 AM	(54.21667, -113.0)
3	Acapulco	10	Valid	Acapulcoite	1914	Fell	01/01/1976 12:00:00 AM	(16.88333, -99.9)
4	Achiras	370	Valid	L6	780	Fell	01/01/1902 12:00:00 AM	(-33.16667, -64.95)

```python
meteorites.sort_values(['mass'], ascending=[False]).head(2)
```

	name	id	nametype	recclass	mass	fall	year	GeoLocation
2	Abee	6	Valid	EH4	107000	Fell	01/01/1952 12:00:00 AM	(54.21667, -113.0)
3	Acapulco	10	Valid	Acapulcoite	1914	Fell	01/01/1976 12:00:00 AM	(16.88333, -99.9)