In [2]:
```
import pandas as pd
taxis = pd.read_csv('Module 6 Data sets/2019_Yellow_Taxi_Trip_Data.csv')
taxis.head()
```

Out[2]:

	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 [3]:
```
taxis.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 [17]:
```
mask = taxis.columns.str.contains('id$|store_and_fwd_flag', regex=True)
columns_to_drop = taxis.columns[mask]
columns_to_drop
```

Out[17]:
```
Index(['vendorid', 'ratecodeid', 'store_and_fwd_flag', 'pulocationid',
       'dolocationid'],
      dtype='object')
```

In [21]:
```
taxis.drop(columns=columns_to_drop)
taxis.head()
```

Out[21]:

	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 [29]:
```
taxis = taxis.drop(columns=columns_to_drop)
taxis.head()
```

Out[29]:

	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	

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 [35]:
```
taxis.dtypes
```

Out[35]:
```
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 [36]:
```
#change column data type taxis [['pickup', "dropoff"]]
taxis [['pickup', 'dropoff']] =taxis [['pickup', 'dropoff']] .apply(pd.to_datetime)
taxis.dtypes
```

Out[36]:
```
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
```

In [39]:
```
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
) #4
)
```

In [40]:
```
taxis.dtypes
```

Out[40]:
```
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 [41]:
```
taxis.head()
```

Out[41]:

	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	
2	2019-10-23 16:08:44	2019-10-23 16:21:11	1	1.36	1	9.5	1.0	
3	2019-10-23 16:22:44	2019-10-23 16:43:26	1	1.00	1	13.0	1.0	
4	2019-10-23 16:45:11	2019-10-23 16:58:49	1	1.96	1	10.5	1.0	

In [43]:
```
taxis.sort_values('trip_distance', ascending = True).head()
#sorting
```

Out[43]:

	pickup	dropoff	passenger_count	trip_distance	payment_type	fare_amount	extra
518	2019-10-23 16:00:09	2019-10-23 16:00:20	3	0.0	2	2.5	0.0
3334	2019-10-23 16:54:54	2019-10-23 16:55:51	1	0.0	1	6.0	0.0
6407	2019-10-23 16:28:13	2019-10-23 16:28:59	1	0.0	2	19.7	0.0
9483	2019-10-23 17:38:15	2019-10-23 17:39:04	1	0.0	2	2.5	1.0
8885	2019-10-23 16:42:51	2019-10-23 16:42:56	1	0.0	1	78.5	0.0

In [49]:

```
taxis.sort_values(['trip_distance', 'extra'], ascending = [False, True]).head
#sorting two columns.
```

Out[49]:
```
<bound method NDFrame.head of                  pickup             dropoff  passen
ger_count  trip_distance  \
8338 2019-10-23 16:50:53 2019-10-24 15:32:55               1          38.11
9965 2019-10-23 17:34:29 2019-10-23 18:48:00               1          37.86
1656 2019-10-23 16:04:45 2019-10-23 19:11:40               3          37.57
2237 2019-10-23 16:09:02 2019-10-23 17:40:37               1          28.41
436  2019-10-23 16:43:22 2019-10-23 17:56:45               4          28.06
...                  ...                 ...             ...           ...
8223 2019-10-23 16:43:30 2019-10-23 16:43:59               2           0.00
8235 2019-10-23 16:21:22 2019-10-23 16:21:36               1           0.00
8736 2019-10-23 16:59:28 2019-10-23 16:59:32               1           0.00
9210 2019-10-23 17:05:49 2019-10-23 17:06:05               2           0.00
3054 2019-10-23 16:46:26 2019-10-23 16:46:58               1           0.00

      payment_type  fare_amount  extra  mta_tax  tip_amount  tolls_amount  \
8338             1        176.0    0.0      0.5       18.29          6.12
9965             2         52.0    4.5      0.5        0.00          6.12
1656             1         52.0    4.5      0.5       13.18          6.12
2237             1         87.5    1.0      0.5        0.00          6.12
436              1         52.0    4.5      0.5       13.18          6.12
...            ...          ...    ...      ...         ...           ...
8223             1         52.0    4.5      0.5       11.45          0.00
8235             1         52.0    4.5      0.5       12.68          6.12
8736             2         52.0    4.5      0.5        0.00          0.00
9210             1         52.0    4.5      0.5       15.86          6.12
3054             1         52.0    7.0      0.5       13.15          6.12

      improvement_surcharge  total_amount  congestion_surcharge  \
8338                    0.3        201.21                   0.0
9965                    0.3         65.92                   2.5
1656                    0.3         79.10                   2.5
2237                    0.3         95.42                   0.0
436                     0.3         79.10                   2.5
...                     ...           ...                   ...
8223                    0.3         68.75                   0.0
8235                    0.3         76.10                   0.0
8736                    0.3         59.80                   2.5
9210                    0.3         79.28                   0.0
3054                    0.3         79.07                   2.5

        elapsed_time  cost_before_tip   tip_pct   fees  avg_speed
8338 0 days 22:42:02           182.92  0.099989   6.92   1.678814
9965 0 days 01:13:31            65.92  0.000000  13.92  30.899116
1656 0 days 03:06:55            65.92  0.199939  13.92  12.059920
2237 0 days 01:31:35            95.42  0.000000   7.92  18.612557
436  0 days 01:13:23            65.92  0.199939  13.92  22.942539
...              ...              ...       ...    ...        ...
8223 0 days 00:00:29            57.30  0.199825   5.30   0.000000
8235 0 days 00:00:14            63.42  0.199937  11.42   0.000000
8736 0 days 00:00:04            59.80  0.000000   7.80   0.000000
9210 0 days 00:00:16            63.42  0.250079  11.42   0.000000
3054 0 days 00:00:32            65.92  0.199484  13.92   0.000000

[10000 rows x 18 columns]>
```

In [46]:
```
taxis.nlargest(4,'passenger_count')

#if maraming tied, it will display them all
```

Out[46]:

	pickup	dropoff	passenger_count	trip_distance	payment_type	fare_amount	extra
41	2019-10-23 16:12:20	2019-10-23 16:38:36	6	3.27	1	17.5	1.0
42	2019-10-23 16:50:46	2019-10-23 16:57:37	6	0.80	1	6.0	1.0
246	2019-10-23 16:41:32	2019-10-23 18:03:31	6	10.46	1	53.0	1.0
326	2019-10-23 16:05:22	2019-10-23 16:20:51	6	1.96	1	11.0	1.0

In [50]:
```
taxis.nsmallest(4,'tip_amount')
```

Out[50]:

	pickup	dropoff	passenger_count	trip_distance	payment_type	fare_amount	extra
1	2019-10-23 16:32:08	2019-10-23 16:45:26	1	2.00	1	10.5	1.0
12	2019-10-23 16:35:45	2019-10-23 16:39:14	1	0.70	2	4.5	3.5
16	2019-10-23 16:07:34	2019-10-23 16:12:48	2	1.13	2	-5.5	-1.0
17	2019-10-23 16:07:34	2019-10-23 16:12:48	2	1.13	2	5.5	1.0

In [56]:
```
import pandas as pd
meteorites = pd.read_csv('Module 6 Data sets/Meteorite_Landings.csv')
meteorites.head()
```

Out[56]:

	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

In [60]:
```
meteorites = meteorites.rename(
    columns = {
        'mass (g)':'mass'
}
)
meteorites.columns
```

Out[60]:
```
Index(['name', 'id', 'nametype', 'recclass', 'mass', 'fall', 'year', 'reclat',
       'reclong', 'GeoLocation'],
      dtype='object')
```

In [65]:
```
meteorites = meteorites.drop(columns='GeoLocation')
meteorites.columns
```

Out[65]:
```
Index(['name', 'id', 'nametype', 'recclass', 'mass', 'fall', 'year', 'reclat',
       'reclong'],
      dtype='object')
```

In [68]:
```
meteorites.sort_values(by =  'mass', ascending=False)
```

Out[68]:

	name	id	nametype	recclass	mass	fall	year	reclat
16392	Hoba	11890	Valid	Iron, IVB	60000000.0	Found	01/01/1920 12:00:00 AM	-19.58333
5373	Cape York	5262	Valid	Iron, IIIAB	58200000.0	Found	01/01/1818 12:00:00 AM	76.13333
5365	Campo del Cielo	5247	Valid	Iron, IAB-MG	50000000.0	Found	12/22/1575 12:00:00 AM	-27.46667
5370	Canyon Diablo	5257	Valid	Iron, IAB-MG	30000000.0	Found	01/01/1891 12:00:00 AM	35.05000
3455	Armanty	2335	Valid	Iron, IIIE	28000000.0	Found	01/01/1898 12:00:00 AM	47.00000
...	...	...	...	...	...	...	...	...
38282	Wei-hui-fu (a)	24231	Valid	Iron	NaN	Found	01/01/1931 12:00:00 AM	NaN
38283	Wei-hui-fu (b)	24232	Valid	Iron	NaN	Found	01/01/1931 12:00:00 AM	NaN
38285	Weiyuan	24233	Valid	Mesosiderite	NaN	Found	01/01/1978 12:00:00 AM	35.26667
41472	Yamato 792768	28117	Valid	CM2	NaN	Found	01/01/1979 12:00:00 AM	-71.50000
45698	Zapata County	30393	Valid	Iron	NaN	Found	01/01/1930 12:00:00 AM	27.00000

45716 rows × 9 columns

In [ ]:
```
Discuss?
```

In [ ]:
```
Discuss?
```

In [ ]:
```
Discuss?
```

In [69]:
```
import pandas as pd
meteorites = pd.read_csv('Module 6 Data sets/Meteorite_Landings.csv')
meteorites.tail()
```

Out[69]:

	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 [70]:
```
import pandas as pd
meteorites = pd.read_csv('Module 6 Data sets/Meteorite_Landings.csv')
meteorites.head()
```

Out[70]:

	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

In [94]:
```
 #Extract only the year from the 'year' column and convert it to a numeric type
meteorites["year"] = pd.to_numeric(meteorites["year"].astype(str).str.slice(0, 4),
```

In [95]:
```
# Update the 'year' column to only contain the year part and convert it to a numeri
```

```
df["year"] = pd.to_numeric(df["year"].astype(str).str.slice(0, 4), errors="coerce")
```

In [96]:
```
# Create a new column indicating if the meteorite was observed falling before 1970
meteorites["before_1970"] = meteorites["year"] < 1970
```

In [97]:
```
# Set the index to the 'id' column
meteorites.set_index("id", inplace=True)
```

In [98]:
```
# Sort the index before using loc[]
meteorites = meteorites.sort_index()
```

In [99]:
```
# Extract rows with IDs between 10036 and 10040 (inclusive)
filtered_meteorites = meteorites.loc[10036:10040]
```

In [100…
```
# Display the result
print(filtered_meteorites)
```

```
           name nametype         recclass  mass (g)   fall  year    reclat  \
id
10036     Enigma    Valid               H4      94.0  Found   NaN  31.33333
10037       Enon    Valid  Iron, ungrouped     763.0  Found   NaN  39.86667
10038      Enshi    Valid               H5    8000.0   Fell   NaN  30.30000
10039  Ensisheim    Valid             LL6  127000.0   Fell   NaN  47.86667

         reclong            GeoLocation  before_1970
id
10036  -82.31667  (31.33333, -82.31667)        False
10037  -83.95000     (39.86667, -83.95)        False
10038  109.50000          (30.3, 109.5)        False
10039    7.35000       (47.86667, 7.35)        False
```

In [ ]:
```

```