# Exercise Part 4

1. Using the meteorite data from the Meteorite_Landings.csv file, create a pivot table that shows both the number of meteorites and the 95th percentile of meteorite mass for those that were found versus observed falling per year from 2005 through 2009 (inclusive). Hint: Be sure to convert the year column to a number as we did in the previous exercise.

In [85]:

```
import pandas as pd

meteorites = pd.read_csv("Meteorite_Landings.csv")
meteorites.head()
```

Out[85]:

| | 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 [86]:

```
# Update the year column to only contain the year.
meteorites['year'] = meteorites.year.str.slice(6,11)
meteorites['year'] = pd.to_numeric(meteorites.year)
meteorites.head()
```

Out[86]:

| | name | id | nametype | recclass | mass (g) | fall | year | reclat | reclong | Gec |
|---|---|---|---|---|---|---|---|---|---|---|
| **0** | Aachen | 1 | Valid | L5 | 21.0 | Fell | 1880.0 | 50.77500 | 6.08333 | |
| **1** | Aarhus | 2 | Valid | H6 | 720.0 | Fell | 1951.0 | 56.18333 | 10.23333 | ( |
| **2** | Abee | 6 | Valid | EH4 | 107000.0 | Fell | 1952.0 | 54.21667 | -113.00000 | ( |
| **3** | Acapulco | 10 | Valid | Acapulcoite | 1914.0 | Fell | 1976.0 | 16.88333 | -99.90000 | ( |
| **4** | Achiras | 370 | Valid | L6 | 780.0 | Fell | 1902.0 | -33.16667 | -64.95000 | (- |

In [87]:

```
# Filter the year ranging from 2005 - 2009
meteorites_pivot = meteorites[(meteorites.year >= 2005) & (meteorites.year <= 2009)
meteorites_pivot.head()
```

Out[87]:

| | name | id | nametype | recclass | mass (g) | fall | year | reclat | reclong | G |
|---|---|---|---|---|---|---|---|---|---|---|
| **30** | Almahata Sitta | 48915 | Valid | Ureilite-an | 3950.0 | Fell | 2008.0 | 20.74575 | 32.41275 | |
| **49** | Ash Creek | 48954 | Valid | L6 | 9500.0 | Fell | 2009.0 | 31.80500 | -97.01000 | |
| **82** | Bassikounou | 44876 | Valid | H5 | 29560.0 | Fell | 2006.0 | 15.78333 | -5.90000 | |
| **101** | Berduc | 48975 | Valid | L6 | 270.0 | Fell | 2008.0 | -31.91000 | -58.32833 | |
| **148** | Bunburra Rockhole | 48653 | Valid | Eucrite | 324.0 | Fell | 2007.0 | -31.35000 | 129.19000 | |

In [88]:

```
# Make pivot table of year as index and fall as column with 95th percentile of mass
meteorites_pivot = meteorites_pivot.pivot_table(
    index = 'year', columns = 'fall', sort = True,
    values = 'mass (g)', aggfunc = lambda x: x.quantile(.95)
)
meteorites_pivot
```

Out[88]:

| fall | Fell | Found |
|---|---|---|
| **year** | | |
| **2005.0** | NaN | 4500.00 |
| **2006.0** | 25008.0 | 1600.50 |
| **2007.0** | 89675.0 | 1126.90 |
| **2008.0** | 106000.0 | 2274.80 |
| **2009.0** | 8333.4 | 1397.25 |

In [ ]:

```
# Make another pivot table but with counted meteorites
```

2. Using the meteorite data from the Meteorite_Landings.csv file, compare summary statistics of the mass column for the meteorites that were found versus observed falling.

In [82]:

```
meteorites.groupby('fall')['mass (g)'].describe()
```

Out[82]:

| | count | mean | std | min | 25% | 50% | 75% | max |
|---|---|---|---|---|---|---|---|---|
| **fall** | | | | | | | | |
| **Fell** | 1075.0 | 47070.715023 | 717067.125826 | 0.1 | 686.00 | 2800.0 | 10450.0 | 23000000.0 |
| **Found** | 44510.0 | 12461.922983 | 571105.752311 | 0.0 | 6.94 | 30.5 | 178.0 | 60000000.0 |

# Exercise Part 5

1. Using the taxi trip data in the 2019_Yellow_Taxi_Trip_Data.csv file, resample the data to an hourly frequency based on the dropoff time. Calculate the total trip_distance, fare_amount, tolls_amount, and tip_amount, then find the 5 hours with the most tips.

In [98]:

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

Out[98]:

| | 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 [99]:

```
# Change dropoff datatype to datetime
taxis['tpep_dropoff_time'] = pd.to_datetime(taxis.tpep_dropoff_datetime)
taxis.dtypes
```

Out[99]:

```
vendorid                          int64
tpep_pickup_datetime             object
tpep_dropoff_datetime            object
passenger_count                   int64
trip_distance                   float64
ratecodeid                        int64
store_and_fwd_flag               object
pulocationid                      int64
dolocationid                      int64
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
tpep_dropoff_time        datetime64[ns]
dtype: object
```

In [100…

```
# Set index
taxis = taxis.set_index('tpep_dropoff_time').sort_index()
taxis
```

Out[100…

| | vendorid | tpep_pickup_datetime | tpep_dropoff_datetime | passenger_coun |
|---|---|---|---|---|
| **tpep_dropoff_time** | | | | |
| **2019-10-23 07:52:09** | 2 | 2019-10-23T07:48:58.000 | 2019-10-23T07:52:09.000 | |
| **2019-10-23 08:03:16** | 2 | 2019-10-23T07:05:34.000 | 2019-10-23T08:03:16.000 | |
| **2019-10-23 08:36:05** | 2 | 2019-10-23T08:18:47.000 | 2019-10-23T08:36:05.000 | |
| **2019-10-23 09:33:13** | 2 | 2019-10-23T09:27:16.000 | 2019-10-23T09:33:13.000 | |
| **2019-10-23 09:49:31** | 2 | 2019-10-23T09:47:25.000 | 2019-10-23T09:49:31.000 | |
| **...** | ... | ... | ... | . |
| **2019-10-24 16:46:42** | 2 | 2019-10-23T16:49:40.000 | 2019-10-24T16:46:42.000 | |
| **2019-10-24 16:47:40** | 2 | 2019-10-23T16:49:36.000 | 2019-10-24T16:47:40.000 | |
| **2019-10-24 16:50:22** | 2 | 2019-10-23T16:51:42.000 | 2019-10-24T16:50:22.000 | |
| **2019-10-24 16:51:44** | 2 | 2019-10-23T16:52:51.000 | 2019-10-24T16:51:44.000 | |
| **2019-10-24 17:15:47** | 2 | 2019-10-23T17:19:31.000 | 2019-10-24T17:15:47.000 | |

10000 rows × 18 columns

In [101…

```
# Filter the dataframe with only the necessary things
taxis = taxis[['trip_distance','fare_amount','tolls_amount','tip_amount']]
taxis.head()
```

Out[101…

| | trip_distance | fare_amount | tolls_amount | tip_amount |
|---|---|---|---|---|
| **tpep_dropoff_time** | | | | |
| **2019-10-23 07:52:09** | 0.67 | 4.5 | 0.0 | 0.0 |
| **2019-10-23 08:03:16** | 14.68 | 50.0 | 0.0 | 4.0 |
| **2019-10-23 08:36:05** | 2.39 | 12.5 | 0.0 | 0.0 |
| **2019-10-23 09:33:13** | 1.11 | 6.0 | 0.0 | 0.0 |
| **2019-10-23 09:49:31** | 0.47 | 52.0 | 0.0 | 0.0 |

In [114…

```python
# Calculate the total trip distance, fare_amount, tolls_amount, and tip_amount by r
test = taxis.select_dtypes(include = 'number').resample('300min').agg(['sum']) # 30
test
```

Out[114…

| | trip_distance | fare_amount | tolls_amount | tip_amount |
|---|---|---|---|---|
| | sum | sum | sum | sum |
| tpep_dropoff_time | | | | |
| **2019-10-23 05:00:00** | 19.32 | 125.00 | 0.00 | 4.00 |
| **2019-10-23 10:00:00** | 30.24 | 139.00 | 0.00 | 18.29 |
| **2019-10-23 15:00:00** | 29947.27 | 149976.73 | 6222.23 | 26257.80 |
| **2019-10-23 20:00:00** | 8.62 | 41.50 | 0.00 | 7.14 |
| **2019-10-24 01:00:00** | 0.00 | 0.00 | 0.00 | 0.00 |
| **2019-10-24 06:00:00** | 27.11 | 210.40 | 0.00 | 5.50 |
| **2019-10-24 11:00:00** | 74.02 | 325.50 | 12.24 | 31.53 |
| **2019-10-24 16:00:00** | 45.92 | 245.00 | 0.00 | 20.68 |