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.
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.

```
import pandas as pd
```

```
meteorites = pd.read_csv('/Meteorite_Landings.csv')
meteorites.head()
```

	name	id	nametype	recclass	mass (g)	fall	year	reclat	reclong	GeoLocation
0	Aachen	1	Valid	L5	21.0	Fell	01/01/1880 12:00:00 AM	50.77500	6.08333	(50.775, 6.08333)
1	Aarhus	2	Valid	H6	720.0	Fell	01/01/1951 12:00:00 AM	56.18333	10.23333	(56.18333, 10.23333)
2	Abee	6	Valid	EH4	107000.0	Fell	01/01/1952 12:00:00 AM	54.21667	-113.00000	(54.21667, -113.0)
3	Acapulco	10	Valid	Acapulcoite	1914.0	Fell	01/01/1976 12:00:00 AM	16.88333	-99.90000	(16.88333, -99.9)
4	Achiras	370	Valid	L6	780.0	Fell	01/01/1902 12:00:00 AM	-33.16667	-64.95000	(-33.16667, -64.95)

Next steps: View recommended plots New interactive sheet

```
meteorites['year'] = pd.to_datetime(meteorites['year'], errors = 'coerce').dt.year
meteorites
```

```
<ipython-input-50-225db33ce179>:1: UserWarning: Could not infer format, so each element will be parsed individually, falling back to `da
  meteorites['year'] = pd.to_datetime(meteorites['year'], errors = 'coerce').dt.year
```

	name	id	nametype	recclass	mass (g)	fall	year	reclat	reclong	GeoLocation
0	Aachen	1	Valid	L5	21.0	Fell	1880.0	50.77500	6.08333	(50.775, 6.08333)
1	Aarhus	2	Valid	H6	720.0	Fell	1951.0	56.18333	10.23333	(56.18333, 10.23333)
2	Abee	6	Valid	EH4	107000.0	Fell	1952.0	54.21667	-113.00000	(54.21667, -113.0)
3	Acapulco	10	Valid	Acapulcoite	1914.0	Fell	1976.0	16.88333	-99.90000	(16.88333, -99.9)
4	Achiras	370	Valid	L6	780.0	Fell	1902.0	-33.16667	-64.95000	(-33.16667, -64.95)
...	...	...	...	...	...	...	...	...	...	...
45711	Zillah 002	31356	Valid	Eucrite	172.0	Found	1990.0	29.03700	17.01850	(29.037, 17.0185)
45712	Zinder	30409	Valid	Pallasite, ungrouped	46.0	Found	1999.0	13.78333	8.96667	(13.78333, 8.96667)
45713	Zlin	30410	Valid	H4	3.3	Found	1939.0	49.25000	17.66667	(49.25, 17.66667)
45714	Zubkovsky	31357	Valid	L6	2167.0	Found	2003.0	49.78917	41.50460	(49.78917, 41.5046)
45715	Zulu Queen	30414	Valid	L3.7	200.0	Found	1976.0	33.98333	-115.68333	(33.98333, -115.68333)

45716 rows × 10 columns

Next steps: View recommended plots New interactive sheet

```
meteorites = meteorites.set_index('year')
```

```
meteorites2 = meteorites.loc[(meteorites.index >= 2005) & (meteorites.index <= 2009)]
```

```
meteorite95 = meteorites2.groupby(['year','fall'])['mass (g)'].quantile(0.95)
```

```
mets = meteorites2.groupby(meteorites2.index).size()
```

```
meteorites3 = meteorite95.reset_index(name='mass (g)').set_index('year')
```

```
meteorites3['MetsCount'] = mets
```

```
meteorites3
```

year	fall	mass (g)	MetsCount
2005.0	Found	4500.00	875
2006.0	Fell	25008.00	2456
2006.0	Found	1600.50	2456
2007.0	Fell	89675.00	1189
2007.0	Found	1126.90	1189
2008.0	Fell	106000.00	957
2008.0	Found	2274.80	957
2009.0	Fell	8333.40	1497
2009.0	Found	1397.25	1497

Next steps: View recommended plots New interactive sheet

```
meteorite_data = pd.read_csv('/Meteorite_Landings.csv')

summary_stats = meteorite_data.groupby("fall")["mass (g)"].describe()
summary_stats
```

fall	count	mean	std	min	25%	50%	75%	max
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

Next steps: View recommended plots New interactive sheet

Exercise Part 5:

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.

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

	vendorid	tpep_pickup_datetime	tpep_dropoff_datetime	passenger_count	trip_distance	ratecodeid	store_and_fwd_flag	pulocationid
0	2	2019-10-23T16:39:42.000	2019-10-23T17:14:10.000	1	7.93	1	N	138
1	1	2019-10-23T16:32:08.000	2019-10-23T16:45:26.000	1	2.00	1	N	11
2	2	2019-10-23T16:08:44.000	2019-10-23T16:21:11.000	1	1.36	1	N	163
3	2	2019-10-23T16:22:44.000	2019-10-23T16:43:26.000	1	1.00	1	N	170
4	2	2019-10-23T16:45:11.000	2019-10-23T16:58:49.000	1	1.96	1	N	163

Next steps: View recommended plots New interactive sheet

```
taxis["tpep_dropoff_datetime"] = pd.to_datetime(taxis["tpep_dropoff_datetime"])


taxis.set_index("tpep_dropoff_datetime", inplace=True)
taxis.head()
```

tpep_dropoff_datetime	vendorid	tpep_pickup_datetime	passenger_count	trip_distance	ratecodeid	store_and_fwd_flag	pulocationid	dol
2019-10-23 17:14:10	2	2019-10-23T16:39:42.000	1	7.93	1	N	138	
2019-10-23 16:45:26	1	2019-10-23T16:32:08.000	1	2.00	1	N	11	
2019-10-23 16:21:11	2	2019-10-23T16:08:44.000	1	1.36	1	N	163	
2019-10-23 16:43:26	2	2019-10-23T16:22:44.000	1	1.00	1	N	170	
2019-10-23 16:58:49	2	2019-10-23T16:45:11.000	1	1.96	1	N	163	

Next steps: View recommended plots New interactive sheet

```
taxis.index = pd.to_datetime(taxis.index)


taxis['hour'] = taxis.index.hour
taxis2 = taxis.groupby('hour')[['trip_distance', 'fare_amount', 'tolls_amount', 'tip_amount']].sum()


taxis3 = taxis2['tip_amount'].nlargest(5)
taxis3
```

hour	tip_amount
16	12249.32
17	12044.03
18	1907.64
15	75.10
19	25.74

dtype: float64