Data Skills Lab

Materials:

- Download the January 2023 Yellow Taxi Data PARQUET file https://www.nyc.gov/site/tlc/about/tlc-trip-record-data.page
- Download the Taxi Zone Lookup table CSV file on the same page
- Read the Yellow Taxi data dictionary https://www.nyc.gov/assets/tlc/downloads/pdf/data_dictionary_trip_records_yellow.pdf

Assignment:

Use pandas to read the 2 data files into your Python notebook. Answer the following questions and upload your results here:

Tips: there are 3 airports, JFK, LaGuardia, and Newark (EWR)

1. Answer the following questions:

- How many pickups happened at each airport?
- How many dropoffs happened at each airport?
- What is the total amount of airport fees collected at each NYC airport? (JFK and LaGuardia)
- What borough destination had the most tips?
- What were the top 10 pickup locations by number of passengers?

2. Create a data visualization of your choice

```
import pandas as pd
import seaborn as sns
import matplotlib.pyplot as plt
```

```
taxi_link = (
    "https://d37ci6vzurychx.cloudfront.net/trip-data/yellow_tripdata_2023-01.parquet"
)
zone_link = "https://d37ci6vzurychx.cloudfront.net/misc/taxi_zone_lookup.csv"

trips = pd.read_parquet(taxi_link, engine="pyarrow")
taxi_zones = pd.read_csv(zone_link)
```

```
trips.head()
```

	VendorID	tpep_pickup_datetime	tpep_dropoff_datetime	passenger_count	trip_distance	RatecodeID	store_and_fwd_flag	PULo
0	2	2023-01-01 00:32:10	2023-01-01 00:40:36	1.0	0.97	1.0	N	
1	2	2023-01-01 00:55:08	2023-01-01 01:01:27	1.0	1.10	1.0	N	
2	2	2023-01-01 00:25:04	2023-01-01 00:37:49	1.0	2.51	1.0	N	
3	1	2023-01-01 00:03:48	2023-01-01 00:13:25	0.0	1.90	1.0	N	
4	2	2023-01-01 00:10:29	2023-01-01 00:21:19	1.0	1.43	1.0	N	

```
trips["pickup_day"] = trips["tpep_pickup_datetime"].apply(lambda x: x.day)
trips["pickup_dow"] = trips["tpep_pickup_datetime"].apply(lambda x: x.day_name())
trips["pickup_dow_num"] = trips["tpep_pickup_datetime"].apply(lambda x: x.day_of_week)
```

```
taxi_zones.head()
airport_list = [1, 132, 138]
airport_zones = taxi_zones.query("LocationID in @airport_list")
```

```
# rows before 3066766
trips_merged_pu = trips.merge(
    taxi_zones, left_on=["PULocationID"], right_on=["LocationID"], how="inner"
)
```

```
trips_merged_pu.head()
```

	VendorID	tpep_pickup_datetime	tpep_dropoff_datetime	passenger_count	trip_distance	RatecodeID	store_and_fwd_flag	PULo
0	2	2023-01-01 00:32:10	2023-01-01 00:40:36	1.0	0.97	1.0	N	
1	2	2023-01-01 00:50:34	2023-01-01 01:02:52	1.0	1.84	1.0	N	
2	1	2023-01-01 00:52:06	2023-01-01 01:02:18	2.0	1.70	1.0	N	
3	2	2023-01-01 00:19:12	2023-01-01 00:38:27	1.0	5.70	1.0	N	
4	2	2023-01-01 00:18:08	2023-01-01 00:32:43	1.0	2.17	1.0	N	

5 rows × 26 columns

```
trips_merged_pu.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 3066766 entries, 0 to 3066765
Data columns (total 26 columns):
 #   Column                 Dtype
---  ------                 -----
 0   VendorID               int64
 1   tpep_pickup_datetime   datetime64[us]
 2   tpep_dropoff_datetime  datetime64[us]
 3   passenger_count        float64
 4   trip_distance          float64
 5   RatecodeID             float64
 6   store_and_fwd_flag     object
 7   PULocationID           int64
 8   DOLocationID           int64
 9   payment_type           int64
 10  fare_amount            float64
 11  extra                  float64
 12  mta_tax                float64
 13  tip_amount             float64
 14  tolls_amount           float64
 15  improvement_surcharge  float64
 16  total_amount           float64
 17  congestion_surcharge   float64
 18  airport_fee            float64
 19  pickup_day             int64
 20  pickup_dow             object
 21  pickup_dow_num         int64
 22  LocationID             int64
 23  Borough                object
 24  Zone                   object
 25  service_zone           object
dtypes: datetime64[us](2), float64(12), int64(7), object(5)
memory usage: 608.3+ MB
```

```
# 1 - How many pickups happened at each airport?
result_1 = (
    trips_merged_pu.query("PULocationID in @airport_list")
    .groupby(["Zone"])
    .agg({"Zone": "count", "passenger_count": "sum"})
)
result_1.columns = ["pickup_count", "passenger_count"]
result_1.reset_index(inplace=True)
```

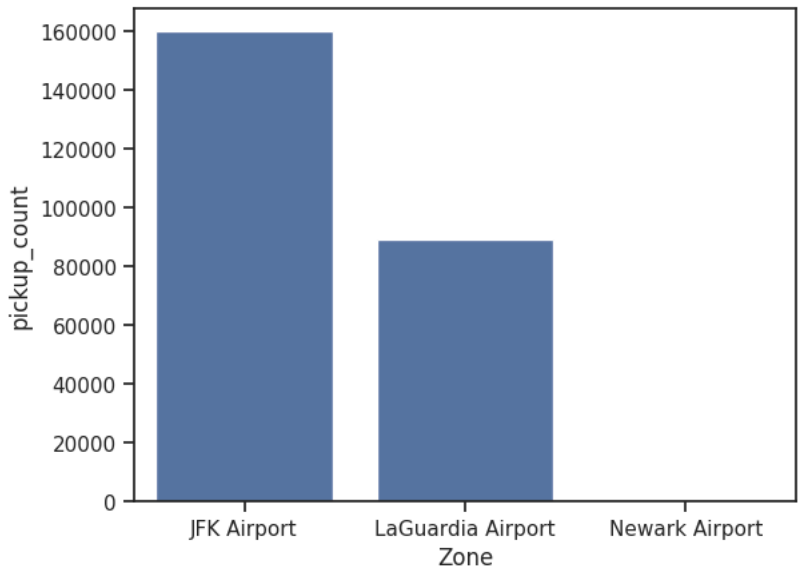
```
result_1
```

	Zone	pickup_count	passenger_count
0	JFK Airport	160030	228407.0
1	LaGuardia Airport	89188	119617.0
2	Newark Airport	410	648.0

Next steps: Generate code with result_1 View recommended plots

```
sns.barplot(result_1, x="Zone", y="pickup_count")
```

```
<Axes: xlabel='Zone', ylabel='pickup_count'>
```




```
# 2 - How many dropoffs happened at each airport?
trips_merged_do = trips.merge(
    taxi_zones.query("LocationID in @airport_list"),
    left_on=["DOLocationID"],
    right_on=["LocationID"],
    how="inner",
)
```

```
trips_merged_do.shape
```

```
(72747, 26)
```

```
result_2 = trips_merged_do.groupby(["Zone"]).agg(
    {"Zone": "count", "passenger_count": "sum"}
)
result_2.columns = ["dropoff_count", "passenger_count"]
result_2.reset_index(inplace=True)
```

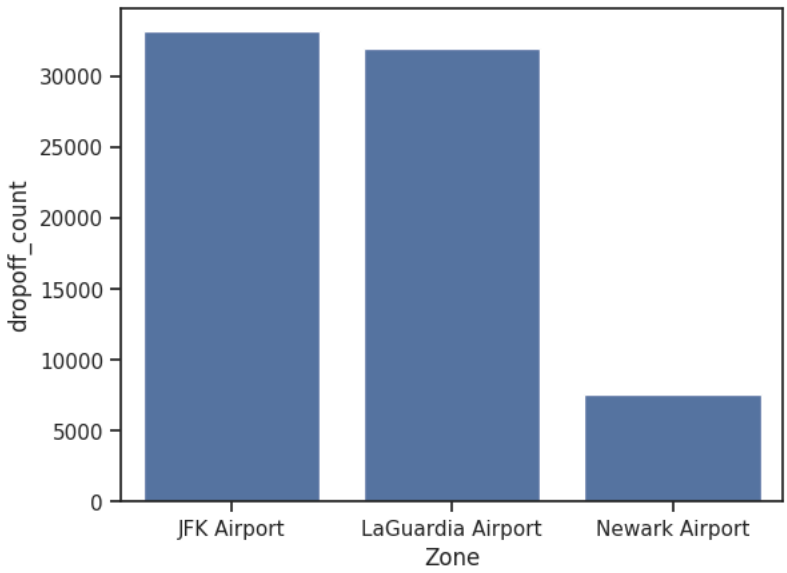
```
result_2
```

	Zone	dropoff_count	passenger_count
0	JFK Airport	33190	49805.0
1	LaGuardia Airport	32031	42552.0
2	Newark Airport	7526	12156.0

Next steps: Generate code with result_2 View recommended plots

```
sns.barplot(result_2, x="Zone", y="dropoff_count")
```

```
<Axes: xlabel='Zone', ylabel='dropoff_count'>
```



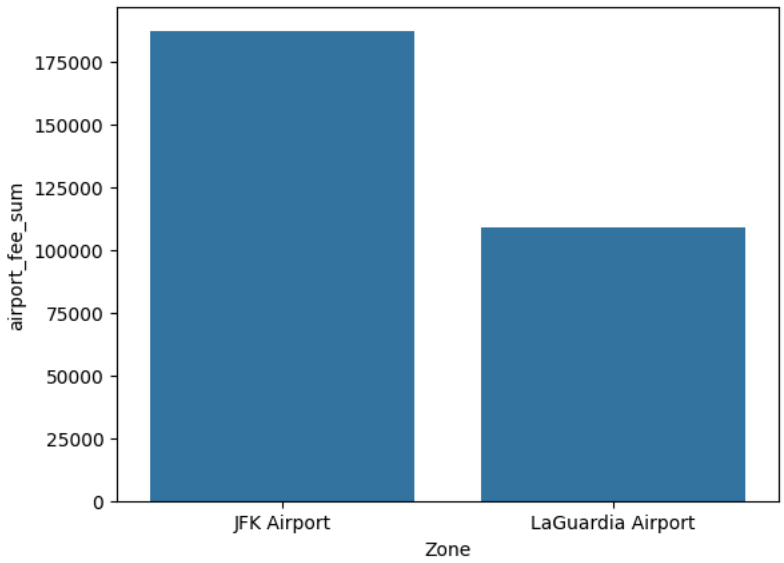

```
# 3 - What is the total amount of airport fees collected at each NYC airport? (JFK and LaGuardia)
result_3 = (
    trips.query("PULocationID in @airport_list")
    .groupby("PULocationID")
    .agg({"airport_fee": "sum", "PULocationID": "count"})
)
result_3.columns = ["airport_fee_sum", "pickup_count"]
result_3.reset_index(inplace=True)


# dropping bad EWR airport row
result_3.drop(0, axis=0, inplace=True)


result_3 = result_3.merge(
    taxi_zones, left_on="PULocationID", right_on="LocationID", how="inner"
)


trips.query("PULocationID == 1 and airport_fee > 0")
```

	VendorID	tpep_pickup_datetime	tpep_dropoff_datetime	passenger_count	trip_distance	RatecodeID	store_and_fwd_flag
261195	2	2023-01-04 14:49:22	2023-01-04 14:49:42	2.0	0.0	5.0	N
2559949	2	2023-01-27 15:15:51	2023-01-27 15:19:06	1.0	0.0	5.0	N

2 rows × 22 columns

```
sns.barplot(result_3, x="Zone", y="airport_fee_sum")
```

```
<Axes: xlabel='Zone', ylabel='airport_fee_sum'>
```




```
# 4 - What borough destination had the most tips?

trips_merged_do_all = trips.merge(
    taxi_zones, left_on=["PULocationID"], right_on=["LocationID"], how="left"
)

borough_metrics = (
    trips_merged_do_all.groupby("Borough")
    .agg(
        {
            "tip_amount": "sum",
            "DOLocationID": "count",
            "trip_distance": "mean",
        }
    )
    .reset_index()
)

borough_metrics.head()
```

	Borough	tip_amount	DOLocationID	trip_distance
0	Bronx	61818.26	18313	10.332796
1	Brooklyn	704746.40	118902	9.061841
2	EWR	108362.21	7526	17.885436
3	Manhattan	8382541.67	2725880	3.189211
4	Queens	873584.81	161624	8.808632

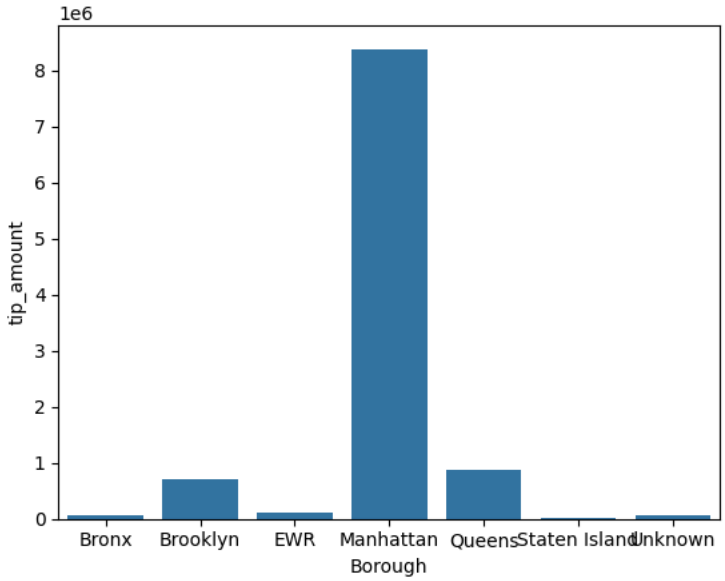
Next steps: Generate code with `borough_metrics` View recommended plots

```
borough_metrics[["Borough", "tip_amount"]]
```

	Borough	tip_amount
0	Bronx	61818.26
1	Brooklyn	704746.40
2	EWR	108362.21
3	Manhattan	8382541.67
4	Queens	873584.81
5	Staten Island	5859.28
6	Unknown	76625.08

```
sns.barplot(borough_metrics, x="Borough", y="tip_amount")
```

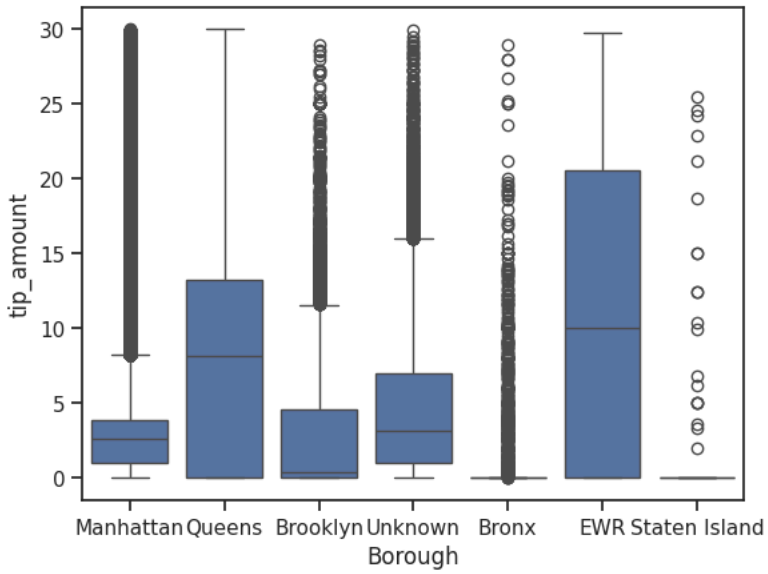
```
<Axes: xlabel='Borough', ylabel='tip_amount'>
```




```
trips_merged_pu.head()
```

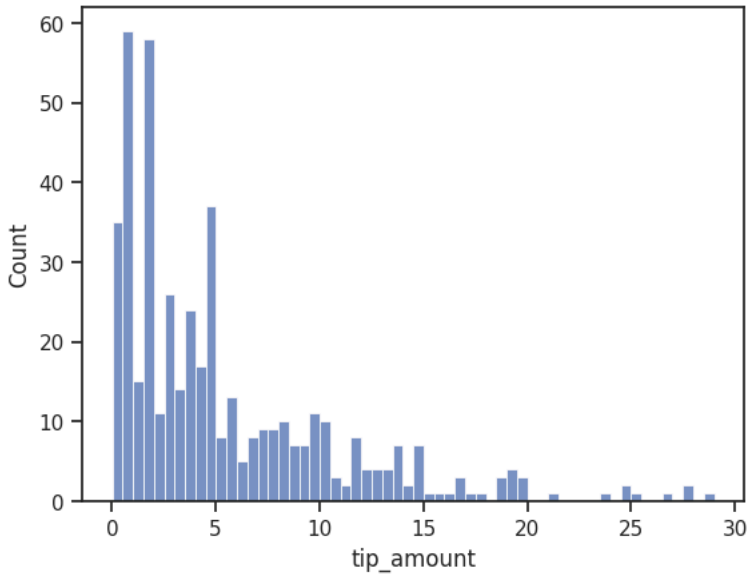
```
sns.boxplot(
    trips_merged_pu.query("tip_amount < 30 and tip_amount >= 0"),
    x="Borough",
    y="tip_amount",
)
```

```
<Axes: xlabel='Borough', ylabel='tip_amount'>
```




```
sns.histplot(
    trips_merged_pu.query("tip_amount < 30 and tip_amount > 0 and Borough == 'Bronx'"),
    x="tip_amount",
    binwidth=0.5,
)
```

```
<Axes: xlabel='tip_amount', ylabel='Count'>
```




```
# 5 - What were the top 10 pickup locations by number of passengers?
result_5 = pd.DataFrame(
    trips_merged_pu.groupby("Zone")["passenger_count"]
    .sum()
    .sort_values(ascending=False)[0:10]
).reset_index()


sns.barplot(result_5, x="Zone", y="passenger_count")
plt.xticks(rotation=45)
```

```
([0, 1, 2, 3, 4, 5, 6, 7, 8, 9],
pickups_day = trips.groupby(["pickup_day"]).agg({"passenger_count": "sum"}).reset_index()
      Text(2, 0, 'Midtown Center'),
pickups_day.head()
```

	pickup_day	passenger_count
0	1	113990.0
1	2	96851.0
2	3	115287.0
3	4	126389.0
4	5	133863.0

Next steps: Generate code with pickups_day View recommended plots

```
sns.barplot(pickups_day, x="pickup_day", y="passenger_count")
```

```
<Axes: xlabel='pickup_day', ylabel='passenger_count'>
```