

Welcome to New York City, one of the most-visited cities in the world. There are many [Airbnb](#) listings in New York City to meet the high demand for temporary lodging for travelers, which can be anywhere between a few nights to many months. In this notebook, we will take a closer look at the New York Airbnb market by combining data from multiple file types like `.csv`, `.tsv`, and `.xlsx`.

Recall that **CSV**, **TSV**, and **Excel** files are three common formats for storing data. Three files containing data on 2019 Airbnb listings are available to you:

`data/airbnb_price.csv`

- `listing_id`: unique identifier of listing
- `price`: nightly listing price in USD
- `nbhood_full`: name of borough and neighborhood where listing is located

`data/airbnb_room_type.xlsx` This is an Excel file containing data on Airbnb listing descriptions and room types.

- `listing_id`: unique identifier of listing
- `description`: listing description
- `room_type`: Airbnb has three types of rooms: shared rooms, private rooms, and entire homes/apartments

`data/airbnb_last_review.tsv` This is a TSV file containing data on Airbnb host names and review dates.

- `listing_id`: unique identifier of listing
- `host_name`: name of listing host
- `last_review`: date when the listing was last reviewed

Our goals are to convert untidy data into appropriate formats to analyze, and answer key questions including:

- What is the average price, per night, of an Airbnb listing in NYC?
- How does the average price of an Airbnb listing, per month, compare to the private rental market?
- How many adverts are for private rooms?
- How do Airbnb listing prices compare across the five NYC boroughs?

We've loaded your first package for you! You can add as many cells as you need.

```
import numpy as np
import pandas as pd
```

```
prices = pd.read_csv("data/airbnb_price.csv")
print(prices.head())
```

	listing_id	price	nbhood_full
0	2595	225 dollars	Manhattan, Midtown
1	3831	89 dollars	Brooklyn, Clinton Hill
2	5099	200 dollars	Manhattan, Murray Hill
3	5178	79 dollars	Manhattan, Hell's Kitchen
4	5238	150 dollars	Manhattan, Chinatown

```
np.random.seed(0)
print(prices.sample(5))
print(prices.shape)
```

	listing_id	price	nbhood_full
18173	29854048	225 dollars	Manhattan, Greenwich Village
4890	8308797	45 dollars	Brooklyn, Sheepshead Bay
15197	26441935	55 dollars	Manhattan, Hell's Kitchen
8813	16783144	75 dollars	Manhattan, Roosevelt Island
20807	32206003	30 dollars	Staten Island, Grant City

(25209, 3)

```
xls = pd.read_excel("data/airbnb_room_type.xlsx")
print(xls.head())
```

	listing_id		description	room_type
0	2595		Skylit Midtown Castle	Entire home/apt
1	3831		Cozy Entire Floor of Brownstone	Entire home/apt
2	5099	Large Cozy 1 BR Apartment In Midtown East		Entire home/apt
3	5178		Large Furnished Room Near B'way	private room
4	5238		Cute & Cozy Lower East Side 1 bdrm	Entire home/apt

```
xls = pd.ExcelFile("data/airbnb_room_type.xlsx")
#sheet_name = list(xls.keys())[0]
# Assuming the first sheet is the one we want
room_types = xls.parse(0)
```

```
print(room_types)
```

	listing_id		description	room_type
0	2595		Skylit Midtown Castle	Entire home/apt
1	3831		Cozy Entire Floor of Brownstone	Entire home/apt
2	5099	Large Cozy 1 BR Apartment In Midtown East		Entire home/apt
3	5178		Large Furnished Room Near B'way	private room
4	5238		Cute & Cozy Lower East Side 1 bdrm	Entire home/apt
...
25204	36425863	Lovely Privet Bedroom with Privet Restroom		PRIVATE ROOM
25205	36427429		No.2 with queen size bed	PRIVATE ROOM
25206	36438336		Seas The Moment	Private room
25207	36442252		1B-1B apartment near by Metro	Entire home/apt
25208	36455809	Cozy Private Room in Bushwick, Brooklyn		Private room

[25209 rows x 3 columns]

```
reviews = pd.read_csv("data/airbnb_last_review.tsv", sep = '\t')
print(reviews)
```

	listing_id	host_name	last_review
0	2595	Jennifer	May 21 2019
1	3831	LisaRoxanne	July 05 2019
2	5099	Chris	June 22 2019
3	5178	Shunichi	June 24 2019
4	5238	Ben	June 09 2019
...
25204	36425863	Rusaa	July 07 2019
25205	36427429	HAi	July 07 2019
25206	36438336	Ben	July 07 2019
25207	36442252	Blaine	July 07 2019
25208	36455809	Christine	July 08 2019

[25209 rows x 3 columns]

```
prices["price"] = prices["price"].str.replace(" dollars", " ")
```

```
prices["price"] = pd.to_numeric(prices["price"])
```

```
prices.describe()
```

	listing_id	price
count		25209
mean		20689218.907215677
std		11029278.151984254
min		2595
25%		12022728
50%		22343909
75%		30376690
max		36455809

8 rows

```
#Subsetting prices for listings costing $0, free_listing
```

```
free_listing = prices["price"] == 0
```

```
max_listing = prices["price"] >= 7500
```

```
#Update prices by removing all free listings
```

```
prices = prices.loc[~free_listing]
```

```
prices = prices.loc[~max_listing]
```

```
#Average the price column in the prices dataframe
```

```
avg_price = round(prices["price"].mean(),2)
```

```
print(avg_price)
```

141.53

```
#Add a new column, price_per_month, to the prices DataFrame
```

```
prices["price_per_month"] = prices["price"]* 365/12  
print(prices)
```

	listing_id	price	nbhood_full	price_per_month
0	2595	225	Manhattan, Midtown	6843.750000
1	3831	89	Brooklyn, Clinton Hill	2707.083333
2	5099	200	Manhattan, Murray Hill	6083.333333
3	5178	79	Manhattan, Hell's Kitchen	2402.916667
4	5238	150	Manhattan, Chinatown	4562.500000
...
25204	36425863	129	Manhattan, Upper East Side	3923.750000
25205	36427429	45	Queens, Flushing	1368.750000
25206	36438336	235	Staten Island, Great Kills	7147.916667
25207	36442252	100	Bronx, Mott Haven	3041.666667
25208	36455809	30	Brooklyn, Bushwick	912.500000

```
[25201 rows x 4 columns]
```

```
average_price_per_month = round(prices["price_per_month"].mean(),2)  
print(average_price_per_month)  
difference = round(average_price_per_month - 3100, 2)  
print(difference)
```

```
4304.73
```

```
1204.73
```

```
#Change all values in the room_type column to lowercase.
```

```
room_types["room_type"] = room_types["room_type"].str.lower()  
print(room_types)
```

```
#Convert the room_type column to a dtype.
```

```
room_types["room_type"] = room_types["room_type"].astype("category")
```

```
#Store the count of values for room_type as room_frequencies.
```

```
room_frequencies = room_types["room_type"].value_counts()  
print(room_frequencies)
```

	listing_id		description	room_type
0	2595		Skylit Midtown Castle	entire home/apt
1	3831		Cozy Entire Floor of Brownstone	entire home/apt
2	5099	Large Cozy 1 BR Apartment In Midtown East		entire home/apt
3	5178		Large Furnished Room Near B'way	private room
4	5238		Cute & Cozy Lower East Side 1 bdrm	entire home/apt
...
25204	36425863	Lovely Privet Bedroom with Privet Restroom		private room
25205	36427429		No.2 with queen size bed	private room
25206	36438336		Seas The Moment	private room
25207	36442252		1B-1B apartment near by Metro	entire home/apt
25208	36455809	Cozy Private Room in Bushwick, Brooklyn		private room

```
[25209 rows x 3 columns]
entire home/apt    13266
private room       11356
shared room        587
Name: room_type, dtype: int64
```

```
import datetime as dt
# Change the data type of the last_review column to datetime
reviews["last_review"] = pd.to_datetime(reviews["last_review"])

# Create first_reviewed, the earliest review date
first_reviewed = reviews["last_review"].dt.date.min()

# Create last_reviewed, the earliest review date
last_reviewed = reviews["last_review"].dt.date.max()
print(first_reviewed)
print(last_reviewed)
```

```
2019-01-01
2019-07-09
```

```
#Joining the DataFrames

# Merge prices and room_types to create rooms_and_prices
rooms_and_prices = prices.merge(room_types, how="outer", on="listing_id")

# Merge rooms_and_prices with the reviews DataFrame to create airbnb_merged
airbnb_merged = rooms_and_prices.merge(reviews, how="outer", on="listing_id")
```

```
#Remove missing observations from airbnb_merged
airbnb_merged.dropna(inplace=True)

airbnb_merged.duplicated().sum()
```

```
0
```

```
#Create a new column in airbnb_merged called borough by using the str.partition() method
on airbnb_merged["nbhood_full"] and indexing the first value using [0]
```

```
airbnb_merged['borough'] = airbnb_merged['nbhood_full'].str.partition(',')[0]
```

```
# Group by borough and calculate summary statistics
boroughs = airbnb_merged.groupby('borough')['price'].agg(['sum', 'mean', 'median', 'count'])
# Round boroughs to 2 decimal places, and sort by mean in descending order
boroughs = boroughs.round(2).sort_values("mean", ascending=False)
# Create labels for the price range, label_names
label_names = ["Budget", "Average", "Expensive", "Extravagant"]

ranges = [0, 69, 175, 350, np.inf]

# Insert new column, price_range, into DataFrame
airbnb_merged["price_range"] = pd.cut(airbnb_merged["price"], bins=ranges,
labels=label_names)
```

```
# Calculate occurrence frequencies for each label, prices_by_borough
prices_by_borough = airbnb_merged.groupby(["borough", "price_range"])
["price_range"].count()
```

```
# Step 10. Storing the final result
```

```
solution = {'avg_price': avg_price,
            'average_price_per_month': average_price_per_month,
            'difference': difference,
            'first_reviewed': first_reviewed,
            'last_reviewed': last_reviewed,
            'prices_by_borough': prices_by_borough}
print(solution)
```

```
{'avg_price': 141.53, 'average_price_per_month': 4304.73, 'difference': 1204.73,
'first_reviewed': datetime.date(2019, 1, 1), 'last_reviewed': datetime.date(2019, 7, 9),
'prices_by_borough': borough      price_range
Bronx      Budget      381
            Average     285
            Expensive    25
            Extravagant   5
Brooklyn    Budget    3194
            Average    5532
```

