Import Libraries & Dataset

In [ ]:
```
import pandas as pd
import zipfile
import kaggle
```

In [ ]:
```
# download datatest from kaggle api
!kaggle datasets download -d hmavrodiev/london-bike-sharing-dataset
```

In [ ]:
```
# extract file from the downloaded zip folder
zipfile_name = 'london-bike-sharing-dataset.zip'
with zipfile.ZipFile(zipfile_name, 'r') as file:
    file.extractall()
```

London Bikes Metadata:

- "timestamp" - timestamp field for grouping the data
- "cnt" - the count of a new bike shares
- "t1" - real temperature in C
- "t2" - temperature in C "feels like"
- "hum" - humidity in percentage
- "wind_speed" - wind speed in km/h
- "weather_code" - category of the weather
- "is_holiday" - boolean field - 1 holiday / 0 non holiday
- "is_weekend" - boolean field - 1 if the day is weekend
- "season" - category field meteorological seasons: 0-spring ; 1-summer; 2-fall; 3-winter.

weathe_code" category description:

- 1 = Clear ; mostly clear but have some values with haze/fog/patches of fog/ fog in vicinity
- 2 = scattered clouds / * few clouds
- 3 = Broken clouds
- 4 = Cloudy
- 7 = Rain/ light Rain shower/ Light rain
- 10 = rain with thunderstorm
- 26 = snowfall
- 94 = Freezing Fog

View Dataset

In [ ]:
```
# read csv as a pandas dataframe and display basic summary info
bike_df = pd.read_csv("london_merged.csv")
bike_df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 17414 entries, 0 to 17413
Data columns (total 10 columns):
 #   Column        Non-Null Count  Dtype  
---  ------        --------------  -----  
 0   timestamp     17414 non-null  object 
 1   cnt           17414 non-null  int64  
 2   t1            17414 non-null  float64
 3   t2            17414 non-null  float64
 4   hum           17414 non-null  float64
 5   wind_speed    17414 non-null  float64
 6   weather_code  17414 non-null  float64
 7   is_holiday    17414 non-null  float64
 8   is_weekend    17414 non-null  float64
 9   season        17414 non-null  float64
dtypes: float64(8), int64(1), object(1)
memory usage: 1.3+ MB
```

In [ ]:
```
print('Rows: {} | Cols: {}'.format(bike_df.shape[0],bike_df.shape[1]))
```

```
Rows: 17414 | Cols: 10
```

In [ ]:
```
bike_df.head(n=10)
```

Out[ ]:

	timestamp	cnt	t1	t2	hum	wind_speed	weather_code	is_holiday	is_weekend	season
0	2015-01-04 00:00:00	182	3.0	2.0	93.0	6.0	3.0	0.0	1.0	3.0
1	2015-01-04 01:00:00	138	3.0	2.5	93.0	5.0	1.0	0.0	1.0	3.0
2	2015-01-04 02:00:00	134	2.5	2.5	96.5	0.0	1.0	0.0	1.0	3.0
3	2015-01-04 03:00:00	72	2.0	2.0	100.0	0.0	1.0	0.0	1.0	3.0
4	2015-01-04 04:00:00	47	2.0	0.0	93.0	6.5	1.0	0.0	1.0	3.0
5	2015-01-04 05:00:00	46	2.0	2.0	93.0	4.0	1.0	0.0	1.0	3.0
6	2015-01-04 06:00:00	51	1.0	-1.0	100.0	7.0	4.0	0.0	1.0	3.0
7	2015-01-04 07:00:00	75	1.0	-1.0	100.0	7.0	4.0	0.0	1.0	3.0
8	2015-01-04 08:00:00	131	1.5	-1.0	96.5	8.0	4.0	0.0	1.0	3.0
9	2015-01-04 09:00:00	301	2.0	-0.5	100.0	9.0	3.0	0.0	1.0	3.0

In [ ]:
```
pd.DataFrame({"Column List": bike_df.columns})
```

Out[ ]:

	Column List
0	timestamp
1	cnt
2	t1
3	t2
4	hum
5	wind_speed
6	weather_code
7	is_holiday
8	is_weekend
9	season

In [ ]:
```
bike_df.isnull().sum()
```

Out[ ]:
```
timestamp       0
cnt             0
t1              0
t2              0
hum             0
wind_speed      0
weather_code    0
is_holiday      0
is_weekend      0
season          0
dtype: int64
```

In [ ]:
```
bike_df.weather_code.value_counts()
```

Out[ ]:
```
weather_code
1.0     6150
2.0     4034
3.0     3551
7.0     2141
4.0     1464
26.0      60
10.0      14
Name: count, dtype: int64
```

In [ ]:
```
bike_df.season.value_counts()
```

Out[ ]:
```
season
0.0    4394
1.0    4387
3.0    4330
2.0    4303
Name: count, dtype: int64
```

Cleaning / Editing

- "season" - category field meteorological seasons: 0-spring ; 1-summer; 2-fall; 3-winter.

In [ ]:
```
#rename colums in the Bike dataframe using a dictionary name 'cols_dict'
#create dictiory for renaming columns
cols_dict ={
    'timestamp':'time',
    'cnt':'count',
    't1':'temp_real_C',
    't2':'temp_feels_like_C',
    'hum':'humidity_percent',
    'wind_speed':'wind_speed_kph',
    'weather_code':'weather',
    'is_holiday':'is_holiday',
    'is_weekend':'is_weekend',
    'season':'season'
}

#rename the columns
bike_df.rename(cols_dict, axis=1, inplace=True)
```

- 1 = Clear ; mostly clear but have some values with haze/fog/patches of fog/ fog in vicinity
- 2 = scattered clouds / * few clouds
- 3 = Broken clouds
- 4 = Cloudy
- 7 = Rain/ light Rain shower/ Light rain
- 10 = rain with thunderstorm
- 26 = snowfall
- 94 = Freezing Fog

In [ ]:
```
#create a season dictionary for mapping the integers to their text values
season_dict={
    '0.0' : 'Spring',
    '1.0' : 'Summer',
    '2.0' : 'Fall',
    '3.0' : 'Winter'
}

#create a weather dictionary for mapping the integers to their text values
weather_dict={
    '1.0':'Clear',
    '2.0':'Scatter Clouds',
    '3.0':'Broken Clouds',
    '4.0':'Cloudy',
    '7.0':'Light Rain',
    '10.0':'Thunderstorm',
    '26.0':'Snowfall',
    '94.0':'Freezing Fog'
}

# changing the seasons column data type to string
bike_df.season = bike_df.season.astype('str')

# mapping the values 0-3 to the actual written seasons
bike_df.season = bike_df.season.map(season_dict)

# changing the weather column data type to string
bike_df.weather = bike_df.weather.astype('str')

# mapping the values to the actual written weathers
bike_df.weather = bike_df.weather.map(weather_dict)
```

In [ ]:
```
#make humidy expressed as an actual %
bike_df.humidity_percent = bike_df.humidity_percent / 100
```

In [ ]:
```
bike_df.humidity_percent
```

Out[ ]:
```
0        0.930
1        0.930
2        0.965
3        1.000
4        0.930
         ...
17409    0.810
17410    0.810
17411    0.785
17412    0.760
17413    0.760
Name: humidity_percent, Length: 17414, dtype: float64
```

In [ ]:
```
bike_df.sample(n=5)
```

Out[ ]:

	time	count	temp_real_C	temp_feels_like_C	humidity_percent	wind_speed_kph	weather	is_holiday	is_weekend	season
5191	2015-08-08 22:00:00	1083	20.0	20.0	0.600	5.5	Clear	0.0	1.0	Summer
11215	2016-04-18 04:00:00	41	5.5	2.5	0.735	13.0	Clear	0.0	0.0	Spring
9438	2016-02-03 03:00:00	29	5.0	1.0	0.700	22.0	Clear	0.0	0.0	Winter
4094	2015-06-24 01:00:00	228	15.0	15.0	0.770	9.0	Clear	0.0	0.0	Summer
13131	2016-07-07 15:00:00	1954	23.0	23.0	0.485	18.0	Clear	0.0	0.0	Summer

Save Dataframe to Excel Sheet

In [ ]:
```
#write dataframe to an Excel File
excel_file_path = 'london_bike_data_final.xlsx'
bike_df.to_excel(excel_file_path, sheet_name ='Bike Data', index = False)

print(f"DataFrame saved to {excel_file_path}")
```

```
DataFrame saved to london_bike_data_final.xlsx
```