London Bike Sharing Dataset - Data Cleaning for Tableau Visualisation

In [1]:
```
# import necessary libraries
import pandas as pd
import zipfile
import kaggle
```

```
Warning: Your Kaggle API key is readable by other users on this system! To
fix this, you can run 'chmod 600 /Users/paulemmerich/.kaggle/kaggle.json'
```

In [3]:
```
# download dataset from kaggle using Kaggle API
!kaggle datasets download -d hmavrodiev/london-bike-sharing-dataset
```

```
Warning: Your Kaggle API key is readable by other users on this system! To
fix this, you can run 'chmod 600 /Users/paulemmerich/.kaggle/kaggle.json'
Downloading london-bike-sharing-dataset.zip to /Users/paulemmerich/Dropbox/
1_Portfolio_github/paulpythonberlin/Dashboard_2-London_Bike_Sharing
100%|████████████████████████████████████████| 165k/165k [00:00<00:00, 528
kB/s]
100%|████████████████████████████████████████| 165k/165k [00:00<00:00, 527
kB/s]
```

In [2]:
```
# extract the file from the downloaded zip file
zipfile_name = "london-bike-sharing-dataset.zip"
with zipfile.ZipFile(zipfile_name, "r") as file:
    file.extractall()
```

In [3]:
```
# read in the csv file as a pandas dataframe
bikes = pd.read_csv("london_merged.csv")
```

In [4]:
```
# explore the data
bikes.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 [5]:
```
bikes
```

Out[5]:

	timestamp	cnt	t1	t2	hum	wind_speed	weather_code	is_holiday	is_weekend
0	2015-01-04 00:00:00	182	3.0	2.0	93.0	6.0	3.0	0.0	1.0
1	2015-01-04 01:00:00	138	3.0	2.5	93.0	5.0	1.0	0.0	1.0
2	2015-01-04 02:00:00	134	2.5	2.5	96.5	0.0	1.0	0.0	1.0
3	2015-01-04 03:00:00	72	2.0	2.0	100.0	0.0	1.0	0.0	1.0
4	2015-01-04 04:00:00	47	2.0	0.0	93.0	6.5	1.0	0.0	1.0
...	...	...	...	...	...	...	...	...	...
17409	2017-01-03 19:00:00	1042	5.0	1.0	81.0	19.0	3.0	0.0	0.0
17410	2017-01-03 20:00:00	541	5.0	1.0	81.0	21.0	4.0	0.0	0.0
17411	2017-01-03 21:00:00	337	5.5	1.5	78.5	24.0	4.0	0.0	0.0
17412	2017-01-03 22:00:00	224	5.5	1.5	76.0	23.0	4.0	0.0	0.0
17413	2017-01-03 23:00:00	139	5.0	1.0	76.0	22.0	2.0	0.0	0.0

17414 rows × 10 columns

In [6]:
```
# count unique values in the weather_code column
bikes.weather_code.value_counts()
```

Out[6]:
```
1.0     6150
2.0     4034
3.0     3551
7.0     2141
4.0     1464
26.0      60
10.0      14
Name: weather_code, dtype: int64
```

In [7]:
```
# count unique values in season column
bikes.season.value_counts()
```

Out[7]:
```
0.0    4394
1.0    4387
3.0    4330
2.0    4303
Name: season, dtype: int64
```

In [8]:
```
# specifying the column names that I want to use
new_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"
}
# renaming the columns to the specified column names
bikes.rename(new_cols_dict, axis=1, inplace=True)
```

In [9]:
```
# changing the humidity values to percentage (a value between 0 and 1)
bikes.humidity_percent = bikes.humidity_percent/100
```

In [10]:
```
# creating a season dictionary so that we can map the integers 0-3 to the ac
season_dict = {
    "0.0": "spring",
    "1.0": "summer",
    "2.0": "autumn",
    "3.0": "winter"
}
# creating a weather dictionary so that we can map the integers to the acual
weather_dict = {
    "1.0": "Clear",
    "2.0": "Scattered clouds",
    "3.0": "Broken clouds",
    "4.0": "Cloudy",
    "7.0": "Rain",
    "10.0": "Rain with thunderstorm",
    "26.0": "Snowfall"
}

# changing the seasons column data type to string
bikes.season = bikes.season.astype("str")
# mapping the values 0-3 to the actual written seasons
bikes.season = bikes.season.map(season_dict)

# changing the weather column data type to string
bikes.weather = bikes.weather.astype("str")
# mapping the values to the actual written weathers
bikes.weather = bikes.weather.map(weather_dict)
```

In [12]:
```
# checking if mapping worked
bikes.head()
```

Out[12]:

	time	count	temp_real_C	temp_feels_like_C	humidity_percent	wind_speed_kph	we
0	2015-01-04 00:00:00	182	3.0	2.0	0.930	6.0	
1	2015-01-04 01:00:00	138	3.0	2.5	0.930	5.0	
2	2015-01-04 02:00:00	134	2.5	2.5	0.965	0.0	
3	2015-01-04 03:00:00	72	2.0	2.0	1.000	0.0	
4	2015-01-04 04:00:00	47	2.0	0.0	0.930	6.5	

In [14]:

```
# writing final dataframe to an excell file that we will use in our Tableau
bikes.to_excel("london_bikes_final.xlsx", sheet_name="Data")
```