This project used Python functions to understand U.S. bike-share data and calculated statistics by building an interactive environment where a user chooses the data and filter for a dataset to analyze.

In [1]:
```
import time
import pandas as pd
import numpy as np
CITY_DATA = { 'chicago':r'C:\Users\HP\Downloads\bikeshare-2\chicago.csv',
              'new York': r'C:\Users\HP\Downloads\bikeshare-2\new york city.csv',
              'washington': r'C:\Users\HP\Downloads\bikeshare-2\washington.csv' }
```

In [2]:
```
def get_filters():
    """
    Asks user to specify a city, month, and day to analyze.

    Returns:
        (str) city - name of the city to analyze
        (str) month - name of the month to filter by, or "all" to apply no month filter
        (str) day - name of the day of week to filter by, or "all" to apply no day filter
    """
    print('Hello! Let\'s explore some US bikeshare data!')

    # get user input for city (chicago, new york city, washington). HINT: Use a while loop to handle invalid in
    city = input('Would you like to see data for Chicago, New York, or Washington?').lower()
    while city not in ['chicago','new york','washington']:
        print('you entered a wrong city name')
        city = input('please write the correct city name?').lower()

    # get user input to filter the data by month, day, or not at all.
    filter = input(' Would you like to filter the data by month, day, or not at all?').lower()
    while filter not in ['month', 'day','not at all']:
        print('you entered a wrong filter')
        filter = input('please choose the right filter').lower()

    if filter == 'not at all':
        month = 'all'
        day = 'all'

    if filter == 'month':
        month = input(' Which month - January, February, March, April, May, June, or all?').lower()
        while month not in ['january', 'february', 'march', 'april', 'may', 'june', 'all']:
            print('you entered a wrong month name')
            month= input('please write the month again').lower()
        day = 'all'

    if filter == 'day':
        month = 'all'
        day = input('Which day - Monday, Tuesday, Wednesday, Thursday, Friday, Saturday, Sunday or all?').lowe
        while day not in ['monday', 'tuesday', 'wednesday', 'thursday', 'friday', 'saturday', 'sunday', 'all']:
            print('you entered a wrong day name')
            day = input('please write the day again').lower()

    return city, month, day
```

In [3]:
```
def load_data(city, month, day):
    """
    Loads data for the specified city and filters by month and day if applicable.

    Args:
        (str) city - name of the city to analyze
        (str) month - name of the month to filter by, or "all" to apply no month filter
        (str) day - name of the day of week to filter by, or "all" to apply no day filter
    Returns:
        df - Pandas DataFrame containing city data filtered by month and day
    """
    # load data file into a dataframe
    df = pd.read_csv(CITY_DATA[city])

    # convert the Start Time column to datetime
    df['Start Time'] = pd.to_datetime(df['Start Time'])

    # extract month and day of week from Start Time to create new columns
    df['month'] = df['Start Time'].dt.month
    df['day_of_week'] = df['Start Time'].dt.day_name()

    # filter by month if applicable
    if month != 'all':
        # use the index of the months list to get the corresponding int
        months = ['january', 'february', 'march', 'april', 'may', 'june']
        month = months.index(month) + 1

        # filter by month to create the new dataframe
        df = df[df['month'] == month]

    # filter by day of week if applicable
    if day != 'all':
        # filter by day of week to create the new dataframe
        df = df[df['day_of_week'] == day.title()]

    return df
```

In [4]:
```
def time_stats(df):
    """Displays statistics on the most frequent times of travel."""
    try:
        print('\nCalculating The Most Frequent Times of Travel...\n')
        start_time = time.time()

        # TO DO: display the most common month
        most_common_month = df['month'].mode()[0]

        # TO DO: display the most common day of week
        most_common_day = df['day_of_week'].mode()[0]

        # TO DO: display the most common start hour
        df['hour'] = df['Start Time'].dt.hour
        most_common_start_hour = df['hour'].mode()[0]

        print("\nThis took %s seconds." % (time.time() - start_time))
        print('-'*40)
        print('most common month is :{}'.format(most_common_month), '\nmost common day is :{}'.format(most_comm
    except:
        print('these data is not availble for this city')

def station_stats(df):
    """Displays statistics on the most popular stations and trip."""
    try:
        print('\nCalculating The Most Popular Stations and Trip...\n')
        start_time = time.time()

        # TO DO: display most commonly used start station
        common_start_station = df['Start Station'].mode()[0]

        # TO DO: display most commonly used end station
        common_end_station = df['End Station'].mode()[0]

        # TO DO: display most frequent combination of start station and end station trip
        frequent_combined_stations = df.groupby(['Start Station','End Station']).size().idxmax()

        print("\nThis took %s seconds." % (time.time() - start_time))
        print('-'*40)
        print('\nmost commonly used start station is {}'.format(common_start_station), '\nmost commonly used en
    except:
        print('these data is not availble for this city')

def trip_duration_stats(df):
    """Displays statistics on the total and average trip duration."""
    try:
        print('\nCalculating Trip Duration...\n')
        start_time = time.time()

        # TO DO: display total travel time
        total_travel_time = df['Trip Duration'].sum()

        # TO DO: display mean travel time
        mean_travel_time = df['Trip Duration'].mean()

        print("\nThis took %s seconds." % (time.time() - start_time))
        print('-'*40)
        print('total travel time: {} hours'.format(total_travel_time), '\nmean travel time: {} hours'.format(me
    except:
        print('these data is not availble for this city')

def user_stats(df):
    """Displays statistics on bikeshare users."""

    try:
        print('\nCalculating User Stats...\n')
        start_time = time.time()

        # TO DO: Display counts of user types
        user_types = df['User Type'].value_counts()

        # TO DO: Display counts of gender
        gender_counts = df['Gender'].value_counts()

        # TO DO: Display earliest, most recent, and most common year of birth
        earliest_year_of_birth = df['Birth Year'].min()
        recent_year_of_birth = df['Birth Year'].max()
        most_common_year_of_birth = df['Birth Year'].mode()

        print("\nThis took %s seconds." % (time.time() - start_time))
        print('-'*40)
        print('counts of user types: {}'.format(user_types),'\ncounts of gender: {}'.format(gender_counts),'\ne
    except:
        print('these data is not availble for this city')
```

In [5]:
```
def main():
    while True:
        city, month, day = get_filters()
        df = load_data(city, month, day)

        time_stats(df)
        station_stats(df)
        trip_duration_stats(df)
        user_stats(df)

        i = 0
        restart = input('\nWould you like to restart? Enter yes or no.\n').lower()
        while restart.lower() == 'yes':
            print(df.iloc[i:i+5])
            i+=5
            restart = input('Do you want to display 5 more rows? yes or no: ').lower()
        if restart.lower() != 'yes':
            print('thank you')
            break

if __name__ == "__main__":
    main()
```

```
Hello! Let's explore some US bikeshare data!
Would you like to see data for Chicago, New York, or Washington?Chicago
 Would you like to filter the data by month, day, or not at all?day
Which day - Monday, Tuesday, Wednesday, Thursday, Friday, Saturday, Sunday or all?Thursday

Calculating The Most Frequent Times of Travel...


This took 0.0200347900390625 seconds.
----------------------------------------
most common month is :6
most common day is :Thursday
most common start hour is :17

Calculating The Most Popular Stations and Trip...


This took 0.028415679931640625 seconds.
----------------------------------------

most commonly used start station is Clinton St & Washington Blvd
most commonly used end station is Clinton St & Washington Blvd
most frequent combination of start station and end station trip is ('Lake Shore Dr & Monroe St', 'Streeter Dr &
Grand Ave')

Calculating Trip Duration...


This took 0.010091066360473633 seconds.
----------------------------------------
total travel time: 34345345 hours
mean travel time: 796.9682097691148 hours

Calculating User Stats...


This took 0.011911153793334961 seconds.
----------------------------------------
counts of user types: Subscriber    38134
Customer      4961
Name: User Type, dtype: int64
counts of gender: Male       29514
Female      8627
Name: Gender, dtype: int64
earliest year of birth is: 1899.0
recent year of birth is: 2016.0
most common year of birth is: 0    1989.0
dtype: float64

Would you like to restart? Enter yes or no.
no
thank you
```