import time

import pandas as pd

import numpy as np

# Use this data to load in

CITY\_DATA = { 'chicago': 'chicago.csv',

'new york city': 'new\_york\_city.csv',

'washington': 'washington.csv' }

# Add days of the week

DAYS = ['Monday', 'Tuesday', 'Wednesday','Thursday','Friday','Saturday', 'Sunday', 'All']

# Add months of the year

MONTHS = ['January', 'February', 'March', 'April', 'May', 'June', 'All']

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!')

# TO DO: get user input for city (chicago, new york city, washington). HINT: Use a while loop to handle invalid inputs

city = input("Please select city >>> (Chicago, New York or Washington): ").lower()

# Prevent error if incorrect

while city not in CITY\_DATA:

print("Please try again name is not valid")

city = input("Please select city >>> (Chicago, New York or Washington): ").lower()

# TO DO: get user input for month (all, january, february, ... , june)

month = input("Please select monthly option >>> (January, February, March, April, May, June or All): ").lower()

# Prevent error if incorrect

while month not in MONTHS:

print("Please try again month is not valid")

month = input("Please select monthly filter >>> (January, February, March, April, May, June or All): ").lower()

# TO DO: get user input for day of week (all, monday, tuesday, ... sunday)

day = input ("Please select day of the week option >>> (Monday, Tuesday, Wednesday, Thursday, Friday, Saturday, Sunday or ALL): ").title()

# Prevent error if incorrect

while day not in DAYS:

print("Please try again day is not valid")

day = input("Please select day of the week option >>> (Monday, Tuesday, Wednesday, Thursday, Friday, Saturday, Sunday or ALL):").title()

print('-'\*40)

return city, month, day

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

"""

#Read and load in as df to return df / use CITY DATA

df = pd.read\_csv(CITY\_DATA[city])

# Clean data and change data types

# Clean Data

#Columns unnamed = 0 drop

df.drop(columns = 'Unnamed: 0', inplace = True)

#Missing values

df.fillna(method = 'fill', inplace = True)

# Change data types

# Change startime/ endtime to datetime format

df['Start Time'] = pd.to\_datetime(df['Start Time'])

df['End Time'] = pd.to\_datetime(df['End Time'])

#Create day and month column

df['Day'] = df['Start Time'].dt.day\_name()

df['Month'] = df['Start Time'].dt.month()

# filter by day

if day != 'All':

df = df[df['Day'] == day]

# Filter by month

if month !='All':

month\_index = MONTHS.index(month) + 1

df = df[df['Month'] ==month ]

# return data

return df

def time\_stats(df):

"""Displays statistics on the most frequent times of travel."""

print('\nCalculating The Most Frequent Times of Travel...\n')

start\_time = time.time()

# TO DO: display the most common month

if month == 'all':

print('The most common month is: ', MONTHS[df['Month'].mode()[0] - 1].title())

else

print('The most common month is: ', month.title())

# TO DO: display the most common day of week

if day == 'All:

print('The most common day is: ', df['Day'].mode()[0].title()

else

print('The most common day is: ', day)

# TO DO: display the most common start hour

# First need to create columns for start time and start hour to find common start hour

# Start time / start hour column

df['Start Hour'] = df['Start Time'].dt.hour

# TO DO: display the most common start hour

print('The most common start hour is: ', df['Start Hour'].mode()[0])

print("\nThis took %s seconds." % (time.time() - start\_time))

print('-'\*40)

def station\_stats(df):

"""Displays statistics on the most popular stations and trip."""

print('\nCalculating The Most Popular Stations and Trip...\n')

start\_time = time.time()

# TO DO: display most commonly used start station

print('the most common start station is: ', df['Start Station'].mode()[0])

# TO DO: display most commonly used end station

print('The most common end station is: ', df['End Station'].mode()[0])

# TO DO: display most frequent combination of start station and end station trip

df['frequent'] = df['Start Station'] + " to " df['End Station']

print('The most frequent start and end station is: ', df['frequent'].mode()[0])

print("\nThis took %s seconds." % (time.time() - start\_time))

print('-'\*40)

def trip\_duration\_stats(df):

"""Displays statistics on the total and average trip duration."""

print('\nCalculating Trip Duration...\n')

start\_time = time.time()

# TO DO: display total travel time / sum = total

print('The total travel time is: ', df['Trip Duration'].sum())

# TO DO: display mean travel time / mean = average

print('the mean travel time is: ', int(df['Trip Duration'].mean()))

print("\nThis took %s seconds." % (time.time() - start\_time))

print('-'\*40)

def user\_stats(df):

"""Displays statistics on bikeshare users."""

print('\nCalculating User Stats...\n')

start\_time = time.time()

# TO DO: Display counts of user types / counts = value\_counts

print(df['User Type'].value\_counts(), '\n')

# TO DO: Display counts of gender counts / counts = value\_counts

print(df['Gender'].value\_counts(), '\n')

# TO DO: Display earliest, most recent, and most common year of birth / earliest = min, recent = max, common = mode

#Earliest / min

print('The oldest birth year is: ', int(df['Birth Year'].min()))

#Recent / max

print('The most recent birth year is: ', int(df['Birth Year'].max()))

#Most common birth year

print('The most common birth year is: ', int(df['Birth Year'].mode()[0]))

print("\nThis took %s seconds." % (time.time() - start\_time))

print('-'\*40)

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)

restart = input('\nWould you like to restart? Enter yes or no.\n')

if restart.lower() != 'yes':

break

if \_\_name\_\_ == "\_\_main\_\_":

main()