In [ ]:

```
"""
DSC540 Week 11 and 12 assignment Exercise
"""

from __future__ import print_function
from itertools import zip_longest

import csv
import logging
import sys
import numpy as np
import pandas as pd
import random
import thinkplot
import thinkstats2
import datetime
import regression
import statsmodels.formula.api as smf
import statsmodels.api as sm
import matplotlib.pyplot as plt
import math
import pandas as pd
import sqlite3
import urllib.request, urllib.parse, urllib.error
import requests
from bs4 import BeautifulSoup
import re
import json
import tweepy
from requests_oauthlib import OAuth1Session
import os
import json
from textblob import TextBlob
import sqlite3
import pandas as pd

### Data Wrangling with Python: Activity 11, page 320 exercise: With following exercise actrivities.
# 1. Connect to petsDB and check whether the connection has been successful.
# 2. Find the different age groups in the persons database.
# 3. Find the age group that has the maximum number of people.
# 4. Find the people who do not have a last name.
# 5. Find out how many people have more than one pet.
# 6. Find out how many pets have received treatment.
# 7. Find out how many pets have received treatment and the type of pet is known.
# 8. Find out how many pets are from the city called east port.
# 9. Find out how many pets are from the city called east port and who received a treatment.

conn = sqlite3.connect("petsdb")

# check if connection is successful by creating cursor
def check_conn(conn):
     try:
        conn.cursor()
        return True
     except Exception as ex:
        return False

def PerformExerciseActivity_11():
    ### 1. Connect to petsDB and check whether the connection has been successful.
    print(check_conn(conn))
    # so the connection to petsDB database is sucessfull.

    ### 2. Find the different age groups in persons database.
    c = conn.cursor()
    for ppl, age in c.execute("SELECT count(*), age FROM persons GROUP BY age"):
        print(f" Age {age} -> {ppl} People ")

    ### 3. Find the age group that has the maximum number of people.
    ppl_age = [ {x:y} for x,y in c.execute("SELECT count(*), age FROM persons GROUP BY age ORDER BY count(*)")]
    # Selecting the last entry from array as it has highest count.
    print(ppl_age[-1:])
    # So the highest number of people belong to age 73 and their count is 5.

    ### 4. Find the people who do not have last name
    result = c.execute("SELECT * FROM persons where last_name is null")
    for record in result:
        print(f"{record}")

    ### 5. Find out how may peoples have more than one Pet.
    ppl_gt_1_pet = c.execute("SELECT * FROM pets GROUP BY owner_id HAVING count(owner_id) >1")
    count=0
    for record in ppl_gt_1_pet:
        print(record)
        count=count+1;

    print(f"Total number of peoples with more than 1 pet are {count}")
    # Total number of peoples with more than 1 pet are 43
```

```
    ### 6. Find out how may pets have received treatment.
    treatment = c.execute("SELECT count(*) FROM pets WHERE treatment_done=1")
    for row in treatment:
        print(f"Pets who received treatment {row}")

    # Pets who received treatment (36,)

    ### 7. Find out how may pets have received treatment and the type of pet is known.
    pet_type_treated = c.execute("SELECT count(*) FROM pets WHERE treatment_done=1 AND pet_type IS NOT null")
    for row in pet_type_treated:
        print(f"Unknown type of pets who received treatment {row}")

    # Unknown type of pets who received treatment (16,)

    ### 8. Find out how may pets are from city east port
    pets_from_east_port = c.execute("SELECT count(*) FROM (select p.pet_name,per.city from pets p, persons per
    for row in pets_from_east_port:
        print(f"Number of Pets from city East Port {row}")

    # Number of Pets from city East Port (49,)

    ### 9. Find out how many pets are from the city called east port and who received a treatement
    east_port_pets_treated = c.execute("SELECT count(*) FROM (select p.pet_name,per.city from pets p, persons p
    for row in east_port_pets_treated:
        print(f"Number of Pets from city East Port for whom Treatment is done {row}")
    # Number of Pets from city East Port for whom Treatment is done (11,)

def main():
    print("Inside Main function")

    ### Data Wrangling with Python: Activity 11, page 320 exercise.
    # Retrieving Data Correctly From Databases. The pets table has the following columns:
    # pet_name: The name of the pet.
    # pet_type: What type of pet it is, for example, cat, dog, and so on. Due to a lack of
    # further information, we do not know which number represents what, but it is an integer and can be null.
    # treatment_done: It is also an integer column, and 0 here represents "No", whereas 1 represents "Yes".

    ### The name of the SQLite DB is petsdb and it is supplied along with the Activity notebook.These steps wil
    # 1. Connect to petsDB and check whether the connection has been successful.
    # 2. Find the different age groups in the persons database.
    # 3. Find the age group that has the maximum number of people.
    # 4. Find the people who do not have a last name.
    # 5. Find out how many people have more than one pet.
    # 6. Find out how many pets have received treatment.
    # 7. Find out how many pets have received treatment and the type of pet is known.
    # 8. Find out how many pets are from the city called east port.
    # 9. Find out how many pets are from the city called east port and who received a treatment.
    PerformExerciseActivity_11()

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

```
Inside Main function
True
 Age 5 -> 2 People
 Age 6 -> 1 People
 Age 7 -> 1 People
 Age 8 -> 3 People
 Age 9 -> 1 People
 Age 11 -> 2 People
 Age 12 -> 3 People
 Age 13 -> 1 People
 Age 14 -> 4 People
 Age 16 -> 2 People
 Age 17 -> 2 People
 Age 18 -> 3 People
 Age 19 -> 1 People
 Age 22 -> 3 People
 Age 23 -> 2 People
 Age 24 -> 3 People
 Age 25 -> 2 People
 Age 27 -> 1 People
 Age 30 -> 1 People
 Age 31 -> 3 People
 Age 32 -> 1 People
 Age 33 -> 1 People
 Age 34 -> 2 People
 Age 35 -> 3 People
 Age 36 -> 3 People
 Age 37 -> 1 People
 Age 39 -> 2 People
 Age 40 -> 1 People
 Age 42 -> 1 People
 Age 44 -> 2 People
 Age 48 -> 2 People
 Age 49 -> 1 People
 Age 50 -> 1 People
 Age 51 -> 2 People
```

```
 Age 52 -> 2 People
 Age 53 -> 2 People
 Age 54 -> 2 People
 Age 58 -> 1 People
 Age 59 -> 1 People
 Age 60 -> 1 People
 Age 61 -> 1 People
 Age 62 -> 2 People
 Age 63 -> 1 People
 Age 65 -> 2 People
 Age 66 -> 2 People
 Age 67 -> 1 People
 Age 68 -> 3 People
 Age 69 -> 1 People
 Age 70 -> 1 People
 Age 71 -> 4 People
 Age 72 -> 1 People
 Age 73 -> 5 People
 Age 74 -> 3 People
[{5: 73}]
(1, 'Erica', None, 22, 'south port', 2345678)
(2, 'Jordi', None, 73, 'east port', 123456)
(3, 'Chasity', None, 70, 'new port', 76856785)
(4, 'Gregg', None, 31, 'new port', 76856785)
(6, 'Cary', None, 73, 'new port', 76856785)
(8, 'Francisca', None, 14, 'west port', 123456)
(10, 'Raleigh', None, 68, 'new port', 2345678)
(11, 'Maria', None, 42, 'west port', 123456)
(12, 'Mariane', None, 62, 'south port', 9756543)
(13, 'Mona', None, 44, 'south port', 76856785)
(14, 'Kayla', None, 36, 'south port', 2345678)
(15, 'Karlie', None, 35, 'west port', 123456)
(16, 'Morris', None, 71, 'west port', 76856785)
(17, 'Sandy', None, 23, 'east port', 2345678)
(18, 'Hector', None, 63, 'east port', 9756543)
(19, 'Hiram', None, 52, 'west port', 2345678)
(20, 'Tressa', None, 59, 'new port', 123456)
(21, 'Berry', None, 22, 'south port', 2345678)
(22, 'Pearline', None, 73, 'new port', 9756543)
(23, 'Maynard', None, 25, 'east port', 123456)
(24, 'Dorian', None, 40, 'east port', 123456)
(25, 'Mylene', None, 5, 'east port', 76856785)
(26, 'Lafayette', None, 34, 'new port', 2345678)
(29, 'Tara', None, 39, 'west port', 123456)
(30, 'Destiny', None, 18, 'south port', 2345678)
(31, 'Lesly', None, 31, 'west port', 123456)
(32, 'Perry', None, 19, 'south port', 76856785)
(35, 'Maritza', None, 73, 'east port', 9756543)
(37, 'Grant', None, 61, 'east port', 76856785)
(39, 'Laury', None, 17, 'east port', 9756543)
(40, 'Name', None, 52, 'east port', 9756543)
(41, 'Estefania', None, 32, 'new port', 76856785)
(42, 'Destiney', None, 65, 'west port', 2345678)
(43, 'Jaquelin', None, 73, 'west port', 9756543)
(45, 'Alfonzo', None, 16, 'east port', 2345678)
(46, 'Lisandro', None, 11, 'new port', 76856785)
(49, 'Priscilla', None, 65, 'east port', 76856785)
(50, 'Elenora', None, 11, 'new port', 76856785)
(52, 'Rudolph', None, 14, 'east port', 76856785)
(56, 'Ona', None, 35, 'east port', 9756543)
(57, 'Rebeca', None, 50, 'new port', 76856785)
(59, 'Sigurd', None, 12, 'west port', 76856785)
(63, 'Alice', None, 8, 'west port', 76856785)
(64, 'Dane', None, 24, 'west port', 9756543)
(65, 'Judge', None, 17, 'south port', 76856785)
(66, 'Allene', None, 9, 'new port', 9756543)
(67, 'Jalen', None, 33, 'new port', 2345678)
(70, 'Myron', None, 36, 'new port', 9756543)
(73, 'Travon', None, 16, 'south port', 2345678)
(74, 'Shayna', None, 60, 'new port', 2345678)
(75, 'Myah', None, 14, 'east port', 2345678)
(82, 'Letha', None, 44, 'new port', 9756543)
(84, 'Felton', None, 74, 'east port', 2345678)
(85, 'London', None, 66, 'east port', 9756543)
(86, 'Koby', None, 31, 'west port', 9756543)
(87, 'Golden', None, 35, 'east port', 76856785)
(89, 'Anissa', None, 8, 'south port', 76856785)
(91, 'Sid', None, 22, 'west port', 123456)
(96, 'Ernesto', None, 69, 'east port', 9756543)
(97, 'Josianne', None, 14, 'west port', 76856785)
(2, 'mani', 1.0, 0)
(5, 'fenga', None, 0)
(6, 'milu', 1.0, 0)
(7, 'olga', 1.0, 0)
(9, 'gimir', None, 0)
(10, 'snoopy', 1.0, 0)
(13, 'gimir', None, 0)
(14, 'unsa', 1.0, 0)
(16, 'dara', None, 0)
```

```
(18, 'deru', None, 0)
(20, 'olga', None, 0)
(21, 'tamari', 1.0, 0)
(24, 'palu', None, 0)
(25, 'raba', None, 0)
(26, 'mani', None, 1)
(27, 'olga', None, 0)
(28, 'mani', 1.0, 0)
(31, 'chegal', 1.0, 1)
(33, 'chegal', 1.0, 0)
(35, 'milu', None, 0)
(36, 'dara', None, 0)
(37, 'dara', 1.0, 0)
(38, 'raba', None, 0)
(39, 'sapi', None, 0)
(40, 'sapi', None, 0)
(42, 'dara', None, 0)
(43, 'tamari', None, 0)
(45, 'sami', None, 1)
(48, 'snoopy', None, 0)
(49, 'sami', 1.0, 0)
(50, 'bulga', 1.0, 0)
(51, 'bulga', None, 0)
(53, 'sami', 1.0, 1)
(56, 'deru', None, 0)
(57, 'mani', 1.0, 0)
(58, 'sapi', 1.0, 0)
(60, 'raba', None, 0)
(64, 'milu', 1.0, 0)
(67, 'raba', None, 0)
(86, 'milu', 1.0, 0)
(98, 'bulga', 1.0, 1)
(99, 'milu', None, 1)
(100, 'chegal', None, 0)
Total number of peoples with more than 1 pet are 43
Pets who received treatment (36,)
Unknown type of pets who received treatment (16,)
Number of Pets from city East Port (49,)
Number of Pets from city East Port for whom Treatment is done (11,)
```