DCS 540 Data Preparation (DSC540-T301 2225-1)

Bellevue University

Assignment: Weeks 11 & 12 Exercises

Author: Jake Meyer

Date: 05/24/2022

1. Data Wrangling with Python: Activity 11, page 320

Activity 11: Retrieving Data Correctly from Databases

In [40]:
```
'''
Import the essential libraries to complete Activity 11 (Activities for Weeks 11 & 12).
Any additional libraries will be imported later in the notebook.
'''
import numpy as np

import pandas as pd

import sqlite3
```

In [41]:
```
'''
Move over a copy of 'petsdb' into the working directory.
Connect to the 'petsdb' which should contain a persons table and a pets table per the a
As seen on pg 305 from DataWrangling with Python text, create a connection with sqlite3
'''
conn = sqlite3.connect('petsdb')
```

In [42]:
```
'''
Per the assignment, check whether the connection has been successful.
Write a function with try/except handling to check the connection.
'''
def test_open(conn):
    try:
        conn.execute("SELECT * FROM persons LIMIT 1")
        print("Connection Successful!")
        return True
    except sqlite3.ProgrammingError as e:
        print("Connection closed. {}".format(e))
        return False
```

In [43]:

```
'''
Test the connection with test_open() function.
'''
print(test_open(conn))
```

```
Connection Successful!
True
```

In [44]:
```
'''
Close the connection to 'petsdb'.
'''
conn.close()
```

In [45]:
```
'''
Retest the test_open(conn) function to ensure the connection shows closed.
'''
print(test_open(conn))
```

```
Connection closed. Cannot operate on a closed database.
False
```

In [46]:
```
'''
Reopen the connection to 'petsdb' to answer the questions for the assignment.
'''
conn = sqlite3.connect('petsdb')
```

In [47]:
```
'''
Create a cursor object using conn.cursor() to communicate with 'petsdb'
'''
cursor = conn.cursor()
```

In [48]:
```
'''
Per the assignment, find the different age groups in the persons database.
Create a for loop counting ages and executing commands (SELECT, FROM, GROUP) from perso
'''
for people, age in cursor.execute("SELECT count(*), age FROM persons GROUP BY age"):
    print("Total of {} people aged {}".format(people, age))
```

```
Total of 2 people aged 5
Total of 1 people aged 6
Total of 1 people aged 7
Total of 3 people aged 8
Total of 1 people aged 9
Total of 2 people aged 11
Total of 3 people aged 12
Total of 1 people aged 13
Total of 4 people aged 14
Total of 2 people aged 16
Total of 2 people aged 17
Total of 3 people aged 18
Total of 1 people aged 19
Total of 3 people aged 22
Total of 2 people aged 23
Total of 3 people aged 24
```

```
Total of 2 people aged 25
Total of 1 people aged 27
Total of 1 people aged 30
Total of 3 people aged 31
Total of 1 people aged 32
Total of 1 people aged 33
Total of 2 people aged 34
Total of 3 people aged 35
Total of 3 people aged 36
Total of 1 people aged 37
Total of 2 people aged 39
Total of 1 people aged 40
Total of 1 people aged 42
Total of 2 people aged 44
Total of 2 people aged 48
Total of 1 people aged 49
Total of 1 people aged 50
Total of 2 people aged 51
Total of 2 people aged 52
Total of 2 people aged 53
Total of 2 people aged 54
Total of 1 people aged 58
Total of 1 people aged 59
Total of 1 people aged 60
Total of 1 people aged 61
Total of 2 people aged 62
Total of 1 people aged 63
Total of 2 people aged 65
Total of 2 people aged 66
Total of 1 people aged 67
Total of 3 people aged 68
Total of 1 people aged 69
Total of 1 people aged 70
Total of 4 people aged 71
Total of 1 people aged 72
Total of 5 people aged 73
Total of 3 people aged 74
```

In [49]:

```
'''
Per the assignment, find the age group with the maximum number of people.
Create another for loop for people and age and put in descending order. Break after
the first output.
'''
for people, age in cursor.execute("SELECT count(*), age FROM persons GROUP BY age ORDER
    print("Maximum number of people {} came from {} age group.".format(people, age))
    break
```

```
Maximum number of people 5 came from 73 age group.
```

In [50]:

```
'''
Per the assignment, find the number of people who do not have a last name.
Utilize the count(*) on null last_name values from persons table.
Read the data from the database by using a for loop on the rows of data.
'''
null_last_name = cursor.execute("SELECT count(*) FROM persons WHERE last_name IS null")
for row in null_last_name:
    print('There are {} individuals with a null last name.'.format(row[0]))
```

```
There are 60 individuals with a null last name.
```

In [51]:
```
'''
Per the assignment, find the number of people with more than one pet.
Utilize the count(*) on owner_id for pets with owner_id >1.
Read the data from the database by using a for loop on the rows of data.
'''
more_pets = cursor.execute("SELECT count(*) FROM (SELECT count(owner_id) FROM pets GROU
for row in more_pets:
    print("{} People have more than one pet.".format(row[0]))
```

```
43 People have more than one pet.
```

In [52]:
```
'''
Per the assignment, find out how many pets have received treatment.
Utilize the count(*) from pets where on treatment_done = 1.
Read the data from the database by using a for loop on the rows of data.
'''
pet_treatment = cursor.execute("SELECT count(*) FROM pets WHERE treatment_done=1")
for row in pet_treatment:
    print('There are {} pets that received treatment.'.format(row[0]))
```

```
There are 36 pets that received treatment.
```

In [27]:
```
'''
Per the assignment, find out how many pets received treatement and the type of pet is k
Utilize the count(*) from pets where on treatment_done = 1 and non-null values for pet_
Read the data from the database by using a for loop on the rows of data.
'''
pet_treatment_type = cursor.execute("SELECT count(*) FROM pets WHERE treatment_done=1 A
for row in pet_treatment_type:
    print('There are {} pets that received treatment and the pet type is known.'.format
```

```
There are 16 pets that received treatment and the pet type is known.
```

In [29]:
```
'''
Per the assignment, find out the number of pets from east port.
JOIN pets, persons based on pets.owner_id WHERE persons.city='east port'. SELECT count(
Read the data from the database by using a for loop on the rows of data.
'''
pet_ep = cursor.execute("SELECT count(*) FROM pets JOIN persons ON pets.owner_id = pers
for row in pet_ep:
    print('There are {} pets from east port.'.format(row[0]))
```

```
There are 49 pets from east port.
```

In [30]:
```
'''
Per the assignment, find out the number of pets from east port.
JOIN pets, persons based on pets.owner_id WHERE persons.city='east port'. SELECT count(
Also include AND pets.treatment_done=1.
Read the data from the database by using a for loop on the rows of data.
'''
pet_ep_treat = cursor.execute("SELECT count(*) FROM pets JOIN persons ON pets.owner_id
for row in pet_ep_treat:
    print('There are {} pets from east port that had treatment.'.format(row[0]))
```

```
There are 11 pets from east port that had treatment.
```