Week9_10_PuppalaSucharitha

August 8, 2022

0.0.1 Exercise : Activity 11

0.0.2 Retrieving Data Correctly From Databases.

Step 1 : Connect to petsDB and check wheather the connection has been successful.

[1]:
```
# Importing sqlite3 andcConnecting to the petsDB
import sqlite3
conn = sqlite3.connect("petsdb")
```

[2]:
```
# Writing a  function to check wheather the connection is successful
def is_opened(conn):
    try:
        conn.execute("SELECT * FROM persons LIMIT 1")
        return True
    except sqlite3.ProgrammingError as e:
        print("Connection closed {}".format(e))
        return False
```

[3]:
```
# Printing the output
print(is_opened(conn))
```

```
True
```

[4]:
```
# Closing the Database Connection
conn.close()
```

[5]:
```
# Checking wheather the connection is open or close
print(is_opened(conn))
```

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

Step 2 : Find the different age groups in the persons database.

[6]:
```
# We have to connect to pets database with the following command
conn = sqlite3.connect("petsdb")
```

[7]:
```
# To get access to Database
c = conn.cursor()
```

[8]:
```
# Listing out the different age groups from the persons database
for ppl, age in c.execute("SELECT count(*), age FROM persons GROUP BY age"):
    print("We have {} people aged {}".format(ppl, age))
```

```
We have 2 people aged 5
We have 1 people aged 6
We have 1 people aged 7
We have 3 people aged 8
We have 1 people aged 9
We have 2 people aged 11
We have 3 people aged 12
We have 1 people aged 13
We have 4 people aged 14
We have 2 people aged 16
We have 2 people aged 17
We have 3 people aged 18
We have 1 people aged 19
We have 3 people aged 22
We have 2 people aged 23
We have 3 people aged 24
We have 2 people aged 25
We have 1 people aged 27
We have 1 people aged 30
We have 3 people aged 31
We have 1 people aged 32
We have 1 people aged 33
We have 2 people aged 34
We have 3 people aged 35
We have 3 people aged 36
We have 1 people aged 37
We have 2 people aged 39
We have 1 people aged 40
We have 1 people aged 42
We have 2 people aged 44
We have 2 people aged 48
We have 1 people aged 49
We have 1 people aged 50
We have 2 people aged 51
We have 2 people aged 52
We have 2 people aged 53
We have 2 people aged 54
We have 1 people aged 58
We have 1 people aged 59
We have 1 people aged 60
We have 1 people aged 61
We have 2 people aged 62
We have 1 people aged 63
We have 2 people aged 65
```

```
We have 2 people aged 66
We have 1 people aged 67
We have 3 people aged 68
We have 1 people aged 69
We have 1 people aged 70
We have 4 people aged 71
We have 1 people aged 72
We have 5 people aged 73
We have 3 people aged 74
```

Step 3 : Find the age group that has maximum number of people.

[9]:
```
# The age group that has more number of people can be known by executing the␣
↪following command

for ppl, age in c.execute("SELECT count(*), age FROM persons GROUP BY age ORDER␣
↪BY count(*) DESC"):
    print("Highest number of people i.e a count of {} came from {} age group".
↪format(ppl, age))
    break
```

```
Highest number of people i.e a count of 5 came from 73 age group
```

Step 4 : Find the people who do not have a Last Name.

[10]:
```
# The number of people that do not have a last name can be known by executing␣
↪the following command

res = c.execute("SELECT count(*) FROM persons WHERE last_name IS null")
for row in res:
    print(row)
```

```
(60,)
```

Step 5 : Find out how many people have more than one pet.

[11]:
```
# The number of people having more than one pet can be known by executing the␣
↪following command
res = c.execute("SELECT count(*) FROM (SELECT count(owner_id) FROM pets GROUP␣
↪BY owner_id HAVING count(owner_id) >1)")
for row in res:
    print("{} People are having more than one pets".format(row[0]))
```

```
43 People are having more than one pets
```

Step 6 : FInd out how many pets have received treatment.

[12]:
```
# The number of pets received treatment can be known by executing the␣
↪followingcommand
res = c.execute("SELECT count(*) FROM pets WHERE treatment_done=1")
for row in res:
```

```
    print(row)
```

```
(36,)
```

Step 7 : FInd out how many pets have received treatment and the type of pet is known.

[13]:
```
# The number of pets received treatment and the type of pet is known , can be␣
↪known by executing the following command
res = c.execute("SELECT count(*) FROM pets WHERE treatment_done=1 AND pet_type␣
↪IS NOT null")
for row in res:
    print(row)
```

```
(16,)
```

Step 8 : Find out how many pets are from the city called east port.

[14]:
```
# The number of pets that are from east port city can be known by executing the␣
↪following command
res = c.execute("SELECT count(*) FROM pets JOIN persons ON pets.owner_id =␣
↪persons.id WHERE persons.city='east port'")
for row in res:
    print(row)
```

```
(49,)
```

Step 9 : Find out how many pets are from the city called east port and who received a treatment.

[15]:
```
# The number of pets that are from east port city and received a treatment can␣
↪be known by executing the following command
res = c.execute("SELECT count(*) FROM pets JOIN persons ON pets.owner_id =␣
↪persons.id WHERE persons.city='east port' AND pets.treatment_done=1")
for row in res:
    print(row)
```

```
(11,)
```