219114/115 Programming I

**Week 12: OOP programming laboratory II**

You can access a solution repo for last week’s lab here:

<https://github.com/parujr/oop_lesson_2>

It will also serve as a starting point for this week’s laboratory.

Please fork it and then clone it to your local repo. If you are not familiar with forking a Github repository, consult the following link for instructions:

<https://docs.github.com/en/get-started/quickstart/fork-a-repo>

You will then do the lab tasks for this week in that directory, commit your change and push it back to your Github repository for grading later. Please also name your remote repo as oop\_lesson\_2.

Access the data files that we will use via the following link:

[week12\_oop\_lab](https://drive.google.com/drive/folders/1SOlXqoLu4t621qpgvVydCFhW0ODtUckI?usp=sharing)

**Task 1:**

* Modify your data\_processing.py to add these data sets as tables into the database
  + Titanic.csv
  + Players.csv
  + Teams.csv
* Commit your change; write a meaningful and relevant message to go with it
* For Players and Teams, modify your data\_processing.py to accommodate the following queries:
  + What player on a team with “ia” in the team name played less than 200 minutes and made more than 100 passes? Select to display the player surname, team, and position
  + The average number of games played for teams ranking below 10 versus teams ranking above or equal 10
  + The average number of passes made by forwards versus by midfielders
* Commit your change;write a meaningful and relevant message to go with it
* For Titanic, modify your data\_processing.py to accommodate the following queries:
  + The average fare paid by passengers in the first class versus in the third class
  + The survival rate of male versus female passengers
* Commit your change and write a meaningful and relevant message to go with it
* Do a git push to your remote repo at this point and show it to the TAs or the instructor

**Task 2:**

* To begin this task, make sure you are in the same working directory as Task 1
* Use the following git commands to create a branch called pivot\_feature

git branch pivot\_feature

git checkout pivot\_feature

and you will switch to pivot\_feature from the main branch

* Create a file named combination\_gen.py that is to contain a function named gen\_comb\_list with the following specifications

def gen\_comb\_list(list\_set):

'''

Parameters:

list\_set: a list of lists where each contains at least one element

Returns:

a list of lists, each of which is made from a combination of elements in each list in list\_set

Examples:

gen\_comb\_list([[1, 2, 3]]) returns [[1], [2], [3]]

gen\_comb\_list([[1, 2, 3], [4, 5]]) returns [[1, 4], [2, 4], [3, 4], [1, 5], [2, 5], [3, 5]]

gen\_comb\_list([[1, 2, 3], [4, 5], [6, 7, 8]]) returns [[1, 4, 6], [2, 4, 6], [3, 4, 6], [1, 5, 6], [2, 5, 6], [3, 5, 6], [1, 4, 7], [2, 4, 7], [3, 4, 7], [1, 5, 7], [2, 5, 7], [3, 5, 7], [1, 4, 8], [2, 4, 8], [3, 4, 8], [1, 5, 8], [2, 5, 8], [3, 5, 8]]

'''

* Code up this function and commit this file to this pivot\_feature branch
* Do a git push to your remote repo and you should see this new branch there as well

**Task 3:**

* At this point, you are ready to add a pivot table operation to the Table class
* The pivot\_table method in that class will make use of the gen\_comb\_list function you implemented in Task 2
* However, your boss interrupts you as he needs to see the result of the following query related to the Titanic data:
  + *Find the total number of male passengers embarked at Southampton*
* At this point, you have to go back to the main branch using the following git command

git checkout main

Then, you need to open the file data\_processing.py to add in the code to get the result for your boss. Once you are done, commit your change to this main branch and write a meaningful and relevant message to go with it

* Now, you are ready to get back to your pivot table task. Use the following git command to go to the pivot\_feature branch again

git checkout pivot\_feature

* Before, you continue to code up the pivot\_table method, let’s see how your commit history look for all the branches by using the following git command

git log --all --graph

You should see two diverging branches, main and pivot\_feature, which will be

merged together at some point.

* Before you add the pivot\_table method, make an enhancement to the aggregate method as follows:

def \_\_is\_float(self, element):

if element is None:

return False

try:

float(element)

return True

except ValueError:

return False

def aggregate(self, function, aggregation\_key):

temps = []

for item1 in self.table:

if self.\_\_is\_float(item1[aggregation\_key]):

temps.append(float(item1[aggregation\_key]))

else:

temps.append(item1[aggregation\_key])

return function(temps)

This improvement will enable more aggregations to be calculated such as one

that counts the number of entries, e.g., lambda x: len(x). Commit your change

once you are done.

* You are now ready to add the pivot\_table method to the Table class. Below is a guide to implement this function:

First, let’s see how this method can be used:

table4 = Table('titanic', titanic)

my\_DB.insert(table4)

my\_table4 = my\_DB.search('titanic')

Table my\_pivot = my\_table4.pivot\_table(['embarked', 'gender', 'class'], ['fare', 'fare', 'fare', 'last'], [lambda x: min(x), lambda x: max(x), lambda x: sum(x)/len(x), lambda x: len(x)])

my\_pivot will have the following format:

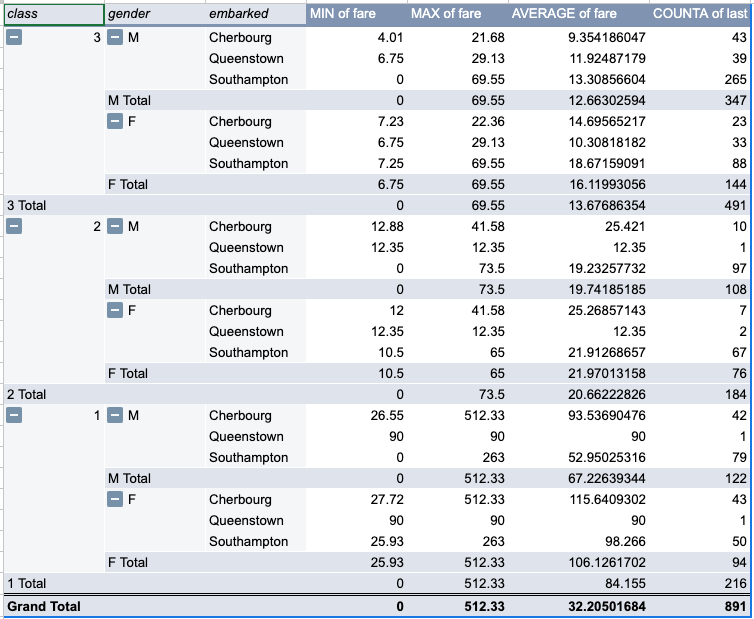
[[['Southampton', 'M', '3'], [0.0, 69.55, 13.308566037735877, 265]], [['Cherbourg', 'M', '3'], [4.01, 21.68, 9.35418604651163, 43]], [['Queenstown', 'M', '3'], [6.75, 29.13, 11.924871794871795, 39]], [['Southampton', 'F', '3'], [7.25, 69.55, 18.67159090909091, 88]], [['Cherbourg', 'F', '3'], [7.23, 22.36, 14.695652173913043, 23]], [['Queenstown', 'F', '3'], [6.75, 29.13, 10.308181818181817, 33]], [['Southampton', 'M', '2'], [0.0, 73.5, 19.232577319587627, 97]], [['Cherbourg', 'M', '2'], [12.88, 41.58, 25.421000000000003, 10]], [['Queenstown', 'M', '2'], [12.35, 12.35, 12.35, 1]],

[['Southampton', 'F', '2'], [10.5, 65.0, 21.91268656716418, 67]], [['Cherbourg', 'F', '2'], [12.0, 41.58, 25.268571428571427, 7]], [['Queenstown', 'F', '2'], [12.35, 12.35, 12.35, 2]],

[['Southampton', 'M', '1'], [0.0, 263.0, 52.95025316455699, 79]], [['Cherbourg', 'M', '1'], [26.55, 512.33, 93.53690476190476, 42]], [['Queenstown', 'M', '1'], [90.0, 90.0, 90.0, 1]],

[['Southampton', 'F', '1'], [25.93, 263.0, 98.26599999999999, 50]], [['Cherbourg', 'F', '1'], [27.72, 512.33, 115.64093023255815, 43]], [['Queenstown', 'F', '1'], [90.0, 90.0, 90.0, 1]]]

This emulates the same pivot operation that you can perform on Google Sheet.



Here is how you code it up in the Table class

class Table:

# code for other methods

def pivot\_table(self, keys\_to\_pivot\_list, keys\_to\_aggreagte\_list, aggregate\_func\_list):

# First create a list of unique values for each key

unique\_values\_list = []

# Here is an example of of unique\_values\_list for

# keys\_to\_pivot\_list = ['embarked', 'gender', 'class']

# unique\_values\_list =

# [['Southampton', 'Cherbourg', 'Queenstown'], ['M', 'F'], ['3', '2',

'1']]

# Get the combination of unique\_values\_list

# You will make use of the function you implemented in Task 2

import combination\_gen

# code that makes a call to combination\_gen.gen\_comb\_list

# Example output:

# [['Southampton', 'M', '3'],

# ['Cherbourg', 'M', '3'],

# ...

# ['Queenstown', 'F', '1']]

# code that filters each combination

# for each filter table applies the relevant aggregate functions

# to keys to aggregate

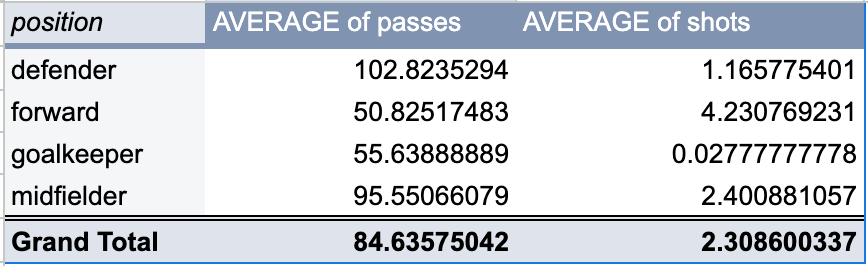
# the aggregate functions is listed in aggregate\_func\_list

# to keys to aggregate is listed in keys\_to\_aggreagte\_list

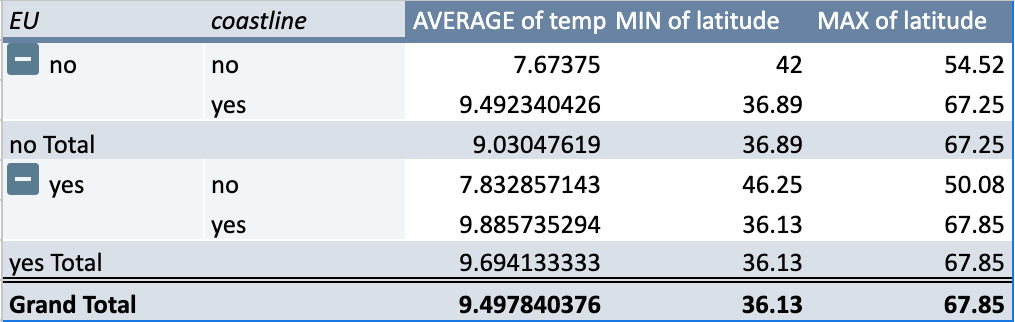
# return a pivot table

* Once you are done with the code, test at with the test case above and commit your change
* Add two three more test cases to emulate the following pivot table operations:

1.



2.



3.

