STA 141B Assignment 2

Due **Februrary 9, 2024** by **11:59pm**. Submit your work by uploading it to Gradescope through Canvas.

Instructions:

1. Provide your solutions in new cells following each exercise description. Create as many new cells as necessary. Use code cells for your Python scripts and Markdown cells for explanatory text or answers to non-coding questions. Answer all textual questions in complete sentences.
2. The use of assistive tools is permitted, but must be indicated. You will be graded on you proficiency in coding. Produce high quality code by adhering to proper programming principles.
3. Export the .jpynb as .pdf and submit it on Gradescope in time. To facilitate grading, indicate the area of the solution on the submission. Submissions without indication will be marked down. No late submissions accepted.
4. If test cases are given, your solution must be in the same format.
5. The total number of points is 10.

Exercise 1

We will use the lichess API to retrieve some information about the current state of chess in the world. In order to answer below questions, make precise and economical requests. You may use:

```
import requests
import json
import pandas

from datetime import datetime
```

```
In [129…   import requests
           import json
           import pandas
           from datetime import datetime
```

(a) What is the real name of the player `muisback` ?

```
In [130…   ###Access API Token
           def read_key(keyfile):
               with open(keyfile) as f:
                   return f.readline().strip("\n")
           key = read_key("../hw2/lichessAPItoken.txt")
```

```
In [131...  response = requests.get("https://lichess.org/api/user/muisback", params={"pr
                'Authorization' : f'Bearer {key}',
            })
            account_data = response.json()
            full_name = f'{account_data["profile"]["firstName"]}, {account_data["profile
            full_name

Out[131...  'Rauf, Mamedov'
```

SOLUTION

Rauf, Mamedov

(b, i) Get the username of the last player that played a rapid game against user `athena-pallada` in 2023. **(ii)** In all games against this user, what is the win-to-loss ratio of `athena-pallada` ?

```
In [173...  response = requests.get("https://lichess.org/api/games/user/athena-pallada",
                params={'perfType' : "rapid",'max' : 5, 'until' : 1704009600000}, #17040
                headers= {'Authorization' : f'Bearer {key}'})
            response = response.text.split('\n')
```

```
In [174...  response[4]
```

```
Out[174...  '[Black "Bacio129"]'
```

```
In [177...  response = requests.get("https://lichess.org/api/games/user/athena-pallada",
                params = {
                    'vs' : 'Bacio129'
                },
                headers = {
                    'Authorization' : f'Bearer {key}'
                })
            response = response.text.split('\n')
```

```
In [185...  for line in response:
                if '[Result' in line:
                    print(line)

            [Result "1-0"]
            [Result "1-0"]
```

SOLUTION

The user name of the last player athena-pallada has played against is Bacio129. As can be seen in the output, in all the games against this player, which is two games, Athena-Pallada has won both games, so they have a win ratio of 2:0. (We know Athena won because the score for both games is 1-0)

(c) Consider the top ten players in the bullet leaderboard. **(i)** Which player has the most bullet games overall? **(ii)** Which player has played the most bullet games relative to account age in days? **(iii)** Which player has the worst win-to-loss ratio over all formats?

```
In [134...   response = requests.get('https://lichess.org/api/player/top/10/bullet',
                 params = {

                 },
                 headers = {
                     'Authorization' : f'Bearer {key}'
                 })
             top_bullet_players = response.json()
```

```
In [135...   bullet_player_user = []
             for i in range(10):
                 bullet_player_user.append(top_bullet_players['users'][i]['username'])
             bullet_player_user
```

```
Out[135...   ['Ediz_Gurel',
             'anhghOst24',
             'V_M',
             'Yulkaaa',
             'Italianchessstar',
             'HowardXue',
             'aaryan_varshney',
             'klari64',
             'TheGreenCloud',
             'iamstraw']
```

```
In [136...   total_bullet_games = []
             for player in bullet_player_user:
                 link = f'https://lichess.org/api/user/{player}/perf/bullet'
                 indiv_player = requests.get(link,
                     params = {

                     },
                     headers = {
                         'Authorization' : f'Bearer {key}'
                     })
                 total_bullet_games.append([player, indiv_player.json()['stat']['count'][
             total_bullet_games
```

```
Out[136…  [['Ediz_Gurel', 5560],
           ['anhghOst24', 3429],
           ['V_M', 2541],
           ['Yulkaaa', 785],
           ['Italianchessstar', 2797],
           ['HowardXue', 5418],
           ['aaryan_varshney', 7806],
           ['klari64', 2825],
           ['TheGreenCloud', 1206],
           ['iamstraw', 6877]]
```

SOLUTION

```
In [137…  max_player = max(total_bullet_games, key=lambda x: x[1])
          max_player
```

```
Out[137…  ['aaryan_varshney', 7806]
```

```
In [138…  player_counter = 0
          for player in bullet_player_user:
              link = f'https://lichess.org/api/user/{player}'
              indiv_player = requests.get(link,
                  params = {

                  },
                  headers = {
                      'Authorization' : f'Bearer {key}'
                  })
              account_creation = indiv_player.json()["createdAt"]
              date_created = datetime.utcfromtimestamp(account_creation / 1000.0)
              curr_date = datetime.strptime('2024-02-05', '%Y-%m-%d')
              days_since = (curr_date - date_created).days
              total_bullet_games[player_counter].append(days_since)
              total_wins = indiv_player.json()["count"]["win"]
              total_loss = indiv_player.json()["count"]["loss"]
              total_bullet_games[player_counter].append(total_wins/total_loss)
              player_counter += 1
```

```
In [139…  total_bullet_games
```

```
Out[139…  [['Ediz_Gurel', 5560, 326, 2.5202774813233724],
           ['anhghOst24', 3429, 683, 2.429595640952108],
           ['V_M', 2541, 2861, 1.9039623908663532],
           ['Yulkaaa', 785, 394, 2.1130434782608694],
           ['Italianchessstar', 2797, 752, 2.035742035742036],
           ['HowardXue', 5418, 2206, 1.8037122969837587],
           ['aaryan_varshney', 7806, 1750, 2.4489672544080605],
           ['klari64', 2825, 1872, 1.4618991793669402],
           ['TheGreenCloud', 1206, 2655, 2.1775067750677506],
           ['iamstraw', 6877, 1338, 1.6606470053267506]]
```

SOLUTION

```python
bullet_games_to_days = max(total_bullet_games, key=lambda x: x[1]/x[2])
bullet_games_to_days
```

Out[140… `['Ediz_Gurel', 5560, 326, 2.5202774813233724]`

SOLUTION

In [141…

```python
worst_winloss_ratio = min(total_bullet_games, key=lambda x: x[3])
worst_winloss_ratio
```

Out[141… `['klari64', 2825, 1872, 1.4618991793669402]`

i) Considering the top 10 players in bullet, the player Zhigalko_Sergei has the most bullet games with 74408 bullet games ii) Similarly, Zhigalko_Sergei has the most bullet games played relative to account age in days with 74408 games played in an account age of 1963 days. iii) Across the top 10 players in bullet the played HowardXue has the worst win loss ratio with a win loss ratio of 1.7906.

(d) Get all games from user `manwithavan`. Group them by opening and print the ten most popular.

In [142…

```python
link = f'https://lichess.org/api/games/user/manwithavan'
indiv_player = requests.get(link,
    params = {
        "opening" : True,
    },
    headers = {
        'Authorization' : f'Bearer {key}'
    })
```

In [143…

```python
openings = []
player_text = indiv_player.text.split('\n')
for player in player_text:
    if player[:8] == '[Opening':
        player = player.strip('[]')[8:]
        player = player.strip('"')
        openings.append(player)
```

```
In [144... import pandas as pd
         num_opening = {}
         for opening in openings:
             if opening in num_opening:
                 num_opening[opening] += 1
             else:
                 num_opening[opening] = 1
         openings = pd.DataFrame(num_opening.items(), columns=['Opening', 'Occurences
```

SOLUTION

```
In [145... openings = openings.sort_values(by="Occurences",ascending=False)
         openings = openings.reset_index(drop=True)
         openings.head(10)
```

Out[145...

	Opening	Occurences
0	Van't Kruijs Opening	7
1	Nimzo-Larsen Attack: Modern Variation	7
2	Mieses Opening	6
3	Pirc Defense	6
4	Zukertort Opening: Kingside Fianchetto	5
5	Caro-Kann Defense: Breyer Variation	5
6	?	5
7	Nimzo-Larsen Attack	5
8	Zukertort Opening: Queenside Fianchetto Variation	5
9	Queen's Pawn Game	5

Exercise 2

As a public organization, the compensations of employees of all institutions of the University of California are freely accessible. These reports cover UC's career faculty and staff employees, as well as part-time, temporary and student employees. See here. Internally, the data requested by the search mask is queried using an undocumented API. For this exercise, you may use:

```
import requests
import pandas

from json import loads
```

Hint: If you encounter an error when parsing the data, try to use string methods (e.g., `str.replace` *) to deal with them.*

(a) Get the compensation information of all UC Davis employees that received a gross pay that exceeded 300000 USD per year for the years 2019 to 2020. Sort the resulting table by year and last name, and print the first six entries.

```
In [146... import requests
         import pandas as pd
         from json import loads
```

```
In [147... url = 'https://ucannualwage.ucop.edu/wage/search.action'
         result = requests.post(url, params = {
             '_search': 'false',
             'nd': 1707338411241,
             'rows': 1000,
             'page': 1,
             'sidx': "EAW_LST_NAM",
             'sord': 'asc',
             'year': 2019,
             'location': "Davis",
             'startSal': 300000,
             'endSal': 9999999,

         })
         result.raise_for_status()
```

```
In [148... result = result.text
         result = result.replace("\'", '\"')
         result = loads(result)
```

```
In [149... columns=['id','year', 'location', 'firstname', 'lastname', 'title', 'gross',
         table = pd.DataFrame(columns=columns)
```

```
In [150... #table for 2019
         for i in range(len(result['rows'])):
```

```
    table.loc[i] = pd.Series(result['rows'][i]['cell'], index=columns)
table
```

Out[150...

	id	year	location	firstname	lastname	title	gross	regular
0	1	2019	Davis	LEONARD	ABBEDUTO	PROF-HCOMP	374157.00	315450.00
1	2	2019	Davis	MEHRDAD	ABEDI	PROF OF CLIN-HCOMP	310945.00	173831.00
2	3	2019	Davis	ALAA	AFIFY	PROF OF CLIN-HCOMP	305852.00	198489.00
3	4	2019	Davis	OMA	AGBAI	HS ASST CLIN PROF-HCOMP	347554.00	117700.00
4	5	2019	Davis	SERGIO	AGUILAR-GAXIOLA	PROF OF CLIN-HCOMP	303159.00	297935.00
...	...	...	...	...	...	...	...	...
507	508	2019	Davis	MANNY	ZEWDU	ASC PHYSCN	436745.00	273006.00
508	509	2019	Davis	YUNLI	ZHENG	ASC PHYSCN	403416.00	275000.00
509	510	2019	Davis	JON	ZHOU	HS ASST CLIN PROF-HCOMP	394174.00	141392.00
510	511	2019	Davis	JORDAN	ZIEGLER	HS CLIN PROF-HCOMP	510902.00	177172.00
511	512	2019	Davis	MARIKE	ZWIENENBERG	HS ASSOC CLIN PROF-HCOMP	462515.00	186040.00

512 rows × 10 columns

In [151...

```
int_columns = ['gross', 'regular', 'overtime', 'other']
table[int_columns] = table[int_columns].astype('float')
table = table.drop(columns=['id'])
table
```

Out[151...

	year	location	firstname	lastname	title	gross	regular	overtime
0	2019	Davis	LEONARD	ABBEDUTO	PROF-HCOMP	374157.0	315450.0	0.0
1	2019	Davis	MEHRDAD	ABEDI	PROF OF CLIN-HCOMP	310945.0	173831.0	0.0
2	2019	Davis	ALAA	AFIFY	PROF OF CLIN-HCOMP	305852.0	198489.0	0.0
3	2019	Davis	OMA	AGBAI	HS ASST CLIN PROF-HCOMP	347554.0	117700.0	0.0
4	2019	Davis	SERGIO	AGUILAR-GAXIOLA	PROF OF CLIN-HCOMP	303159.0	297935.0	0.0
...	...	...	...	...	...	...	...	..
507	2019	Davis	MANNY	ZEWDU	ASC PHYSCN	436745.0	273006.0	0.0
508	2019	Davis	YUNLI	ZHENG	ASC PHYSCN	403416.0	275000.0	0.0
509	2019	Davis	JON	ZHOU	HS ASST CLIN PROF-HCOMP	394174.0	141392.0	0.0
510	2019	Davis	JORDAN	ZIEGLER	HS CLIN PROF-HCOMP	510902.0	177172.0	0.0
511	2019	Davis	MARIKE	ZWIENENBERG	HS ASSOC CLIN PROF-HCOMP	462515.0	186040.0	0.0

512 rows × 9 columns

In [152...

```python
#table for 2020
url = 'https://ucannualwage.ucop.edu/wage/search.action'
result = requests.post(url, params = {
    '_search': 'false',
    'nd': 1707338411241,
    'rows': 1000,
    'page': 1,
```

```
        'sidx': "EAW_LST_NAM",
        'sord': 'asc',
        'year': 2020,
        'location': "Davis",
        'startSal': 300000,
        'endSal': 9999999,

})
result.raise_for_status()
result = result.text
result = result.replace("\'", '\"')
result = loads(result)
```

In [153...
```
columns=['id','year', 'location', 'firstname', 'lastname', 'title', 'gross',
table2 = pd.DataFrame(columns=columns)
```

In [154...
```
for i in range(len(result['rows'])):
    table2.loc[i] = pd.Series(result['rows'][i]['cell'], index=columns)
table2
```

	id	year	location	firstname	lastname	title	gross	regular
0	1	2020	Davis	LEONARD	ABBEDUTO	PROF-HCOMP	397500.00	354193.00
1	2	2020	Davis	MEHRDAD	ABEDI	PROF OF CLIN-HCOMP	392816.00	188476.00
2	3	2020	Davis	MARIANNE	ABOUYARED	ASST PROF OF CLIN-HCOMP	373400.00	150714.00
3	4	2020	Davis	JASON	ADAMS	ASST PROF OF CLIN-HCOMP	388210.00	150154.00
4	5	2020	Davis	ALAA	AFIFY	PROF OF CLIN-HCOMP	307000.00	212959.00
...	...	...	...	...	...	...	...	...
541	542	2020	Davis	XIAO	ZHAO	HS ASST CLIN PROF-HCOMP	409767.00	134659.00
542	543	2020	Davis	YUNLI	ZHENG	ASC PHYSCN	387954.00	275000.00
543	544	2020	Davis	JON	ZHOU	HS ASST CLIN PROF-HCOMP	340028.00	128634.00
544	545	2020	Davis	LARA	ZIMMERMANN	ASST PROF OF CLIN-HCOMP	302334.00	144071.00
545	546	2020	Davis	MARIKE	ZWIENENBERG	HS ASSOC CLIN PROF-HCOMP	513500.00	217762.00

546 rows × 10 columns

```python
int_columns = ['gross', 'regular', 'overtime', 'other']
table2[int_columns] = table2[int_columns].astype('float')
table2 = table2.drop(columns=['id'])
table2
```

Out[155...

	year	location	firstname	lastname	title	gross	regular	overtim
0	2020	Davis	LEONARD	ABBEDUTO	PROF-HCOMP	397500.0	354193.0	0
1	2020	Davis	MEHRDAD	ABEDI	PROF OF CLIN-HCOMP	392816.0	188476.0	0
2	2020	Davis	MARIANNE	ABOUYARED	ASST PROF OF CLIN-HCOMP	373400.0	150714.0	0
3	2020	Davis	JASON	ADAMS	ASST PROF OF CLIN-HCOMP	388210.0	150154.0	0
4	2020	Davis	ALAA	AFIFY	PROF OF CLIN-HCOMP	307000.0	212959.0	0
...	...	...	...	...	...	...	...	...
541	2020	Davis	XIAO	ZHAO	HS ASST CLIN PROF-HCOMP	409767.0	134659.0	0
542	2020	Davis	YUNLI	ZHENG	ASC PHYSCN	387954.0	275000.0	0
543	2020	Davis	JON	ZHOU	HS ASST CLIN PROF-HCOMP	340028.0	128634.0	0
544	2020	Davis	LARA	ZIMMERMANN	ASST PROF OF CLIN-HCOMP	302334.0	144071.0	0
545	2020	Davis	MARIKE	ZWIENENBERG	HS ASSOC CLIN PROF-HCOMP	513500.0	217762.0	0

546 rows × 9 columns

```
In [156…  table = pd.concat([table, table2], ignore_index=True)
          table
```

Out[156…

	year	location	firstname	lastname	title	gross	regular	overtin
0	2019	Davis	LEONARD	ABBEDUTO	PROF-HCOMP	374157.0	315450.0	C
1	2019	Davis	MEHRDAD	ABEDI	PROF OF CLIN-HCOMP	310945.0	173831.0	C
2	2019	Davis	ALAA	AFIFY	PROF OF CLIN-HCOMP	305852.0	198489.0	C
3	2019	Davis	OMA	AGBAI	HS ASST CLIN PROF-HCOMP	347554.0	117700.0	C
4	2019	Davis	SERGIO	AGUILAR-GAXIOLA	PROF OF CLIN-HCOMP	303159.0	297935.0	C
...	...	...	...	...	...	...	...	
1053	2020	Davis	XIAO	ZHAO	HS ASST CLIN PROF-HCOMP	409767.0	134659.0	C
1054	2020	Davis	YUNLI	ZHENG	ASC PHYSCN	387954.0	275000.0	C
1055	2020	Davis	JON	ZHOU	HS ASST CLIN PROF-HCOMP	340028.0	128634.0	C
1056	2020	Davis	LARA	ZIMMERMANN	ASST PROF OF CLIN-HCOMP	302334.0	144071.0	C
1057	2020	Davis	MARIKE	ZWIENENBERG	HS ASSOC CLIN PROF-HCOMP	513500.0	217762.0	C

1058 rows × 9 columns

SOLUTION

```
In [157… table = table.sort_values(by=['year', 'lastname'])
         table.head(6)
```

Out[157…

	year	location	firstname	lastname	title	gross	regular	overtime	
0	2019	Davis	LEONARD	ABBEDUTO	PROF-HCOMP	374157.0	315450.0	0.0	58
1	2019	Davis	MEHRDAD	ABEDI	PROF OF CLIN-HCOMP	310945.0	173831.0	0.0	137
2	2019	Davis	ALAA	AFIFY	PROF OF CLIN-HCOMP	305852.0	198489.0	0.0	107
3	2019	Davis	OMA	AGBAI	HS ASST CLIN PROF-HCOMP	347554.0	117700.0	0.0	229
4	2019	Davis	SERGIO	AGUILAR-GAXIOLA	PROF OF CLIN-HCOMP	303159.0	297935.0	0.0	5
5	2019	Davis	DEBBIE	AIZENBERG	HS ASSOC CLIN PROF-HCOMP	315975.0	209652.0	0.0	106

(b) Report the mean compensation for each title type: For gross pay, other pay and overtime pay, report the top six titles together with the number of counts.

```
In [159… unique_titles = table['title'].unique()
         unique_titles
```

```
Out[159… array(['PROF-HCOMP', 'PROF OF CLIN-HCOMP', 'HS ASST CLIN PROF-HCOMP',
              'HS ASSOC CLIN PROF-HCOMP', 'HS CLIN PROF-HCOMP',
              'ASSOC PROF OF CLIN-HCOMP', 'PROF-AY-B/E/E', 'ASC PHYSCN',
              'ASST PROF OF CLIN-HCOMP', 'VIS ASST PROF-HCOMP',
              'NURSE SVC MGR 4', 'PROF IN RES-HCOMP', 'MGD CARE MGR 3', 'DEAN',
              'PROF-AY', 'ASST PROF IN RES-HCOMP', 'PHYSCN SR',
              'CLIN APPLICATIONS MGR 4', 'PROF-SFT-VM',
              'AMBUL CARE ADMSTN MGR 4', 'AGRON AES-SFT-VM', 'CIO MED CTR',
              'ADMIN MGR 4', 'RECALL HCOMP', 'INFO SYS MGR 4', 'AGRON AES',
              'PERFUSIONIST SR NEX', 'EXEC VC AND PROVOST', 'PROF-AY-LAW',
              'VC DEV AND UNIV REL', 'CMO MED CTR', 'PROF-FY',
              'LECT-AY-CONTINUING', 'ASC PHYSCN DIPLOMATE', 'VC AND DEAN SOM',
              'CHF NURSE OFCR', 'CFO MED CTR', 'CHAN', 'VC RSCH',
              'ASSOC PROF IN RES-HCOMP', 'PERFUSION SUPV 2', 'ASSOC PROF-HCOMP',
              'VC IT', 'REVENUE CYCLE HC MGR 4', 'MGN COUNSEL 3',
              'FUNDRAISING MGR 4', 'REGL AND CMPLNC HC MGR 3', 'VC BUS ADMSTN',
              'AST PHYSCN', 'AGRON AES-AY', 'NURSE ANESTHETIST MGR 1',
              'CLIN LAB SCI', 'CLIN PROFL SVC MGR 4', 'COO MED CTR',
              'DECISION SUPP MGR 4', 'DIRECTOR', 'CHF CAMPUS COUNSEL',
              'FAC PROJECT MGR 3', 'MGD CARE MGR 4', 'PROF-FY-B/E/E',
              'ATH MGR 4', 'ASSOC PROF-AY-B/E/E', 'CLIN NURSE 2',
              'FINANCIAL SVC MGR 4', 'HR MGR 4', 'NURSE PD'], dtype=object)
```

```
In [160… grouped_table = table.groupby('title').agg({'gross': 'mean', 'regular': 'mea
         grouped_table
```

Out[160…

	title	gross	regular	overtime	other
0	ADMIN MGR 4	315212.00	315212.00	0.0	0.0
1	AGRON AES	339860.00	339860.00	0.0	0.0
2	AGRON AES-AY	353895.00	231213.00	0.0	122682.0
3	AGRON AES-SFT-VM	311520.50	291520.50	0.0	20000.0
4	AMBUL CARE ADMSTN MGR 4	362160.00	309783.00	0.0	52377.0
...	...	...	...	...	...
61	VC BUS ADMSTN	358360.50	358343.00	0.0	17.5
62	VC DEV AND UNIV REL	431460.50	422544.50	0.0	8916.0
63	VC IT	365548.00	365548.00	0.0	0.0
64	VC RSCH	347230.50	347230.50	0.0	0.0
65	VIS ASST PROF-HCOMP	314808.75	280025.75	0.0	34783.0

66 rows × 5 columns

```
In [161… grouped_table['count'] = table.groupby('title').size().reset_index(name='cou
         grouped_table
```

Out[161...

	title	gross	regular	overtime	other	count
0	ADMIN MGR 4	315212.00	315212.00	0.0	0.0	5
1	AGRON AES	339860.00	339860.00	0.0	0.0	2
2	AGRON AES-AY	353895.00	231213.00	0.0	122682.0	1
3	AGRON AES-SFT-VM	311520.50	291520.50	0.0	20000.0	2
4	AMBUL CARE ADMSTN MGR 4	362160.00	309783.00	0.0	52377.0	2
...	...	...	...	...	...	...
61	VC BUS ADMSTN	358360.50	358343.00	0.0	17.5	2
62	VC DEV AND UNIV REL	431460.50	422544.50	0.0	8916.0	2
63	VC IT	365548.00	365548.00	0.0	0.0	2
64	VC RSCH	347230.50	347230.50	0.0	0.0	2
65	VIS ASST PROF-HCOMP	314808.75	280025.75	0.0	34783.0	4

66 rows × 6 columns

In [162...
```python
grouped_table = grouped_table.sort_values(by='gross', ascending=False)
grouped_table
```

Out[162...

	title	gross	regular	overtime	other	count
60	VC AND DEAN SOM	960075.0	759375.0	0.0	200700.0	1
25	COO MED CTR	877439.5	622980.5	0.0	254459.0	2
24	CMO MED CTR	683718.0	575187.5	0.0	108530.5	2
15	CFO MED CTR	647717.5	553678.0	0.0	94039.5	2
16	CHAN	519136.0	510220.0	0.0	8916.0	2
...	...	...	...	...	...	...
17	CHF CAMPUS COUNSEL	305646.5	305646.5	0.0	0.0	2
33	HR MGR 4	302000.0	302000.0	0.0	0.0	1
32	FUNDRAISING MGR 4	301577.0	293041.0	0.0	8536.0	2
43	NURSE PD	300754.0	170057.0	110187.0	20510.0	1
30	FAC PROJECT MGR 3	300134.0	217356.0	0.0	82778.0	1

66 rows × 6 columns

In [163...
```python
grouped_table = grouped_table.reset_index(drop=True)
```

SOLUTION

```
In [164…  grouped_table.head(6)
```

Out[164…

	title	gross	regular	overtime	other	count
0	VC AND DEAN SOM	960075.00000	759375.000000	0.0	200700.000000	1
1	COO MED CTR	877439.50000	622980.500000	0.0	254459.000000	2
2	CMO MED CTR	683718.00000	575187.500000	0.0	108530.500000	2
3	CFO MED CTR	647717.50000	553678.000000	0.0	94039.500000	2
4	CHAN	519136.00000	510220.000000	0.0	8916.000000	2
5	PROF OF CLIN-HCOMP	488237.76378	251241.480315	0.0	236996.283465	127

```
In [ ]:
```