Assignment 5. Yelizaveta Semikina.

Exercise 1.

1. DataFrame from CSV The UIC2016MensBasketBall.csv records the game results of the 2016 Men's Basketball. Implement Python code to load the exact data into a data frame df2016.

In [1]:
```
import pandas as pd

# 1. Load CSV into a dataframe
df2016 = pd.read_csv('UIC2016Basketball.csv')
```

1. Column Names Assign column headers to: Date Opponent UIC Score Opp Score UIC Field Goal Percentage Opp Field Goal Percentage UIC 3 point Field Goal Percentage Opp 3 point Field Goal Percentage UIC Rebound Opp Rebound UIC Assists Opp Assist

In [3]:
```
df2016.columns = ['Date', 'Opponent', 'UIC Score', 'Opp Score', 'UIC Field Goal
                  'Opp Field Goal Percentage', 'UIC 3 point Field Goal Percentag
                  'Opp 3 point Field Goal Percentage', 'UIC Rebound', 'Opp Rebou
                  'UIC Assists', 'Opp Assists']
```

1. Missing data Fill the missing data cell with a hyphen symbol '-'.

In [4]:
```
df2016.fillna('-', inplace=True)
```

1. Data Types What are the data types used in this data frame? Provide both Python code and the answer.

In [5]:
```
print(df2016.dtypes)
```

```
Date                                 object
Opponent                             object
UIC Score                             int64
Opp Score                             int64
UIC Field Goal Percentage            object
Opp Field Goal Percentage             int64
UIC 3 point Field Goal Percentage    object
Opp 3 point Field Goal Percentage    object
UIC Rebound                           int64
Opp Rebound                          object
UIC Assists                          object
Opp Assists                           int64
dtype: object
```

1. Count How many school opponents did Men's Basketball played in 2016? Provide both Python code and the answer.

In [6]:
```
opponents = df2016['Opponent'].nunique()
print("Number of school opponents played in 2016:", opponents)
```

```
Number of school opponents played in 2016: 20
```

1. Filter List all the games that UIC scored more than 65 points. Provide both Python code and the answer.

In [7]:
```
uic_high_scoring_games = df2016[df2016['UIC Score'] > 65]
print(uic_high_scoring_games)
```

```
          Date            Opponent  UIC Score  Opp Score  \
0   11/13/2015       San Francisco         75         78
2   11/24/2015           Roosevelt         96         58
6   12/12/2015            Illinois         79         83
9   12/22/2015      Purdue Calumet         91         72
12    1/8/2016             Detroit         69         87
14   1/14/2016          Green Bay         76         78
17   1/22/2016  Northern Kentucky         69         82
18   1/24/2016       Wright State         66         80
19   1/28/2016   Youngstown State         78         82
20   1/30/2016    Cleveland State         72         70
23   2/13/2016  Northern Kentucky         79         77
24   2/16/2016   Youngstown State         91         92
25   2/19/2016             Detroit         72         83
27   2/26/2016          Green Bay         69         85
28   2/28/2016           Milwaukee         85         98

   UIC Field Goal Percentage  Opp Field Goal Percentage  \
0                       28.0                         24
2                       35.0                         22
6                          -                         25
9                       30.0                         26
12                      21.0                         26
14                      29.0                         29
17                      23.0                         25
18                      22.0                         29
19                         -                         29
20                      21.0                         26
23                      24.0                         27
24                      38.0                         35
25                      24.0                         27
27                      22.0                         30
28                      30.0                         30

   UIC 3 point Field Goal Percentage  Opp 3 point Field Goal Percentage  \
0                                8.0                                8.0
2                                1.0                                5.0
6                                9.0                                9.0
9                               10.0                                8.0
12                               2.0                                4.0
14                               5.0                                4.0
17                               1.0                                9.0
18                               2.0                                9.0
19                               5.0                               14.0
20                               4.0                                4.0
23                               7.0                                9.0
```

```
24                              6.0                              5.0
25                              4.0                              7.0
27                              6.0                              9.0
28                              9.0                             15.0
```

	UIC Rebound	Opp Rebound	UIC Assists	Opp Assists
0	36	42.0	17.0	15
2	52	23.0	-	15
6	38	25.0	18.0	18
9	36	-	21.0	18
12	35	32.0	6.0	10
14	33	40.0	20.0	19
17	36	34.0	3.0	14
18	36	25.0	-	14
19	49	37.0	16.0	21
20	35	34.0	13.0	16
23	45	30.0	13.0	16
24	60	45.0	12.0	11
25	34	36.0	9.0	19
27	31	44.0	7.0	17
28	36	39.0	13.0	25

1. Win-Los-Tie Calculate numbers of games of Wins, Losses and Ties, accordingly.

In [8]:
```
num_wins = len(df2016[df2016['UIC Score'] > df2016['Opp Score']])
num_losses = len(df2016[df2016['UIC Score'] < df2016['Opp Score']])
num_ties = len(df2016[df2016['UIC Score'] == df2016['Opp Score']])

print("Number of Wins:", num_wins)
print("Number of Losses:", num_losses)
print("Number of Ties:", num_ties)
```

```
Number of Wins: 5
Number of Losses: 25
Number of Ties: 0
```

Exercise 2.

1. Load data and add column names. (Hint: To load data from .txt with pandas, you can use data = pd.read_csv('test.txt', sep=","))

In [9]:
```
columns = ['ID', 'First Name', 'Last Name', 'Gender', 'Age', 'Country', 'Job Cat
data = pd.read_csv('customer-savings.txt', sep=',', header=None, names=columns)
```

1. What's the average balance for male and female? Provide both Python code and the answer

In [10]:
```
gender_mean_salary = data.groupby('Gender')['Salary'].mean()

print("Average balance for male: {:.2f}".format(gender_mean_salary['Male']))
print("Average balance for female: {:.2f}".format(gender_mean_salary['Female']))
```

```
Average balance for male: 40018.01
Average balance for female: 39471.90
```

1. What's the average balance for while collar and blue collar in England? Provide both Python code and the answer

In [11]:
```
filtered_data = data[(data['Country'] == 'England') & ((data['Job Category'] ==
job_category_mean_salary = filtered_data.groupby('Job Category')['Salary'].mean(

print("Average balance for White Collar workers in England: {:.2f}".format(job_c
print("Average balance for Blue Collar workers in England: {:.2f}".format(job_ca
```

```
Average balance for White Collar workers in England: 39156.31
Average balance for Blue Collar workers in England: 38567.84
```

Exercise 3.

1. Inner join on column "Account Number".
2. Full outer join on column "Account Number".

In [12]:
```
customer_status = pd.read_csv('customer-status.csv')
sales = pd.read_csv('sales.csv')

merged_inner = pd.merge(customer_status, sales, on="account number", how="inner"
merged_outer = pd.merge(customer_status, sales, on="account number", how="outer"

print(merged_inner)
print(merged_outer)
```

```
     account number                          name_x  status  \
0            740150                      Barton LLC    gold
1            740150                      Barton LLC    gold
2            740150                      Barton LLC    gold
3            714466                Trantow-Barrows  silver
4            714466                Trantow-Barrows  silver
..              ...                             ...     ...
113          257198  Cronin, Oberbrunner and Spencer    gold
114          257198  Cronin, Oberbrunner and Spencer    gold
115          257198  Cronin, Oberbrunner and Spencer    gold
116          257198  Cronin, Oberbrunner and Spencer    gold
117          257198  Cronin, Oberbrunner and Spencer    gold

                           name_y       sku  quantity  unit price  \
0                      Barton LLC  S1-82801        29       60.81
1                      Barton LLC  B1-20000        20       73.93
2                      Barton LLC  S2-83881        12       22.62
3                Trantow-Barrows   B1-33087        43       32.77
4                Trantow-Barrows   S2-16558        20       78.23
..                             ...       ...       ...         ...
113  Cronin, Oberbrunner and Spencer  S1-93683        34       79.57
114  Cronin, Oberbrunner and Spencer  S1-82801        22       12.01
115  Cronin, Oberbrunner and Spencer  S2-00301        19       41.81
116  Cronin, Oberbrunner and Spencer  S1-30248        11       58.82
117  Cronin, Oberbrunner and Spencer  S2-77896         4       23.04

     ext price                 date
0      1763.49  2014-03-07 10:24:54
1      1478.60  2014-03-15 18:21:23
2       271.44  2014-03-17 02:39:33
```

```
3      1409.11  2014-03-14 12:47:48
4      1564.60  2014-03-17 09:03:19
..         ...                  ...
113    2705.38  2014-03-12 08:58:47
114     264.22  2014-03-17 10:05:43
115     794.39  2014-03-27 03:52:01
116     647.02  2014-03-27 20:40:13
117      92.16  2014-03-30 18:12:17

[118 rows x 9 columns]
     account number           name_x  status                       name_y  \
0            740150       Barton LLC    gold                   Barton LLC
1            740150       Barton LLC    gold                   Barton LLC
2            740150       Barton LLC    gold                   Barton LLC
3            714466  Trantow-Barrows  silver                Trantow-Barrows
4            714466  Trantow-Barrows  silver                Trantow-Barrows
..              ...              ...     ...                           ...
137          604255              NaN     NaN  Halvorson, Crona and Champlin
138          604255              NaN     NaN  Halvorson, Crona and Champlin
139          604255              NaN     NaN  Halvorson, Crona and Champlin
140          604255              NaN     NaN  Halvorson, Crona and Champlin
141          604255              NaN     NaN  Halvorson, Crona and Champlin

          sku  quantity  unit price  ext price                 date
0    S1-82801        29       60.81    1763.49  2014-03-07 10:24:54
1    B1-20000        20       73.93    1478.60  2014-03-15 18:21:23
2    S2-83881        12       22.62     271.44  2014-03-17 02:39:33
3    B1-33087        43       32.77    1409.11  2014-03-14 12:47:48
4    S2-16558        20       78.23    1564.60  2014-03-17 09:03:19
..        ...       ...         ...        ...                  ...
137  B1-33087        28       61.35    1717.80  2014-03-21 05:41:09
138  S2-00301        35       24.33     851.55  2014-03-21 20:12:32
139  S2-77896        23       64.91    1492.93  2014-03-24 19:21:21
140  B1-53102        32       86.77    2776.64  2014-03-30 01:14:16
141  S2-78676        18       57.02    1026.36  2014-03-31 06:53:52

[142 rows x 9 columns]
```

1. Left join on column “Account Number”, using customer-status.csv as the base.

In [13]:
```
left_join_customer = pd.merge(customer_status, sales, on='account number', how='
print(left_join_customer)
```

```
     account number                          name_x  status  \
0            740150                     Barton LLC    gold
1            740150                     Barton LLC    gold
2            740150                     Barton LLC    gold
3            714466                Trantow-Barrows  silver
4            714466                Trantow-Barrows  silver
..              ...                             ...     ...
113          257198  Cronin, Oberbrunner and Spencer    gold
114          257198  Cronin, Oberbrunner and Spencer    gold
115          257198  Cronin, Oberbrunner and Spencer    gold
116          257198  Cronin, Oberbrunner and Spencer    gold
117          257198  Cronin, Oberbrunner and Spencer    gold

                          name_y       sku  quantity  unit price  \
0                     Barton LLC  S1-82801        29       60.81
1                     Barton LLC  B1-20000        20       73.93
```

```
2                        Barton LLC  S2-83881        12       22.62
3                    Trantow-Barrows  B1-33087        43       32.77
4                    Trantow-Barrows  S2-16558        20       78.23
..                               ...       ...       ...         ...
113  Cronin, Oberbrunner and Spencer  S1-93683        34       79.57
114  Cronin, Oberbrunner and Spencer  S1-82801        22       12.01
115  Cronin, Oberbrunner and Spencer  S2-00301        19       41.81
116  Cronin, Oberbrunner and Spencer  S1-30248        11       58.82
117  Cronin, Oberbrunner and Spencer  S2-77896         4       23.04

     ext price                 date
0      1763.49  2014-03-07 10:24:54
1      1478.60  2014-03-15 18:21:23
2       271.44  2014-03-17 02:39:33
3      1409.11  2014-03-14 12:47:48
4      1564.60  2014-03-17 09:03:19
..         ...                  ...
113    2705.38  2014-03-12 08:58:47
114     264.22  2014-03-17 10:05:43
115     794.39  2014-03-27 03:52:01
116     647.02  2014-03-27 20:40:13
117      92.16  2014-03-30 18:12:17

[118 rows x 9 columns]
```

1. Left join on column “Account Number”, using sales.csv as the base.

In [14]:

```
left_join_sales = pd.merge(sales, customer_status, on='account number', how='lef
print(left_join_sales)
```

```
     account number                      name_x       sku  quantity  \
0            163416                 Purdy-Kunde  S1-30248        19
1            527099             Sanford and Sons  S2-82423         3
2            527099             Sanford and Sons  B1-50809         8
3            737550  Fritsch, Russel and Anderson  B1-50809        20
4            688981                 Keeling LLC  B1-86481        -1
..              ...                         ...       ...       ...
137          737550  Fritsch, Russel and Anderson  B1-65551        12
138          642753                 Pollich LLC  S1-93683        21
139          412290               Jerde-Hilpert  B1-20000        30
140          307599   Kassulke, Ondricka and Metz  S2-16558        46
141          672390              Kuhn-Gusikowski  B1-04202        19

     unit price  ext price                 date                      name_y  \
0         65.03    1235.57  2014-03-01 16:07:40                         NaN
1         76.21     228.63  2014-03-01 17:18:01             Sanford and Sons
2         70.78     566.24  2014-03-01 18:53:09             Sanford and Sons
3         50.11    1002.20  2014-03-01 23:47:17                         NaN
4         97.16     -97.16  2014-03-02 01:46:44                 Keeling LLC
..          ...        ...                  ...                         ...
137       56.24     674.88  2014-03-31 08:43:24                         NaN
138       92.57    1943.97  2014-03-31 11:37:34                 Pollich LLC
139       22.38     671.40  2014-03-31 21:41:31               Jerde-Hilpert
140       56.04    2577.84  2014-03-31 22:11:22  Kassulke, Ondricka and Metz
141       27.86     529.34  2014-03-31 23:13:14              Kuhn-Gusikowski

     status
0       NaN
1    bronze
```

```
2    bronze
3       NaN
4    silver
..      ...
137     NaN
138  bronze
139  bronze
140  bronze
141  silver

[142 rows x 9 columns]
```