Week 3_Quiz #3 (Working)

March 11, 2021

1 Week 3 Quiz 3 Working

[4]:
```
import pandas as pd
import numpy as np
# First we create two DataFrames, staff and students. (List of Dicts)
staff_df = pd.DataFrame([{'Name': 'Kelly', 'Role': 'Director of HR'},
                         {'Name': 'Sally', 'Role': 'Course liasion'},
{'Name': 'James', 'Role': 'Grader'}]) # And lets index these staff by name
staff_df = staff_df.set_index('Name')
# Now we'll create a student dataframe
student_df = pd.DataFrame([{'Name': 'James', 'School': 'Business'},
                           {'Name': 'Mike', 'School': 'Law'},
{'Name': 'Sally', 'School': 'Engineering'}])
student_df = student_df.set_index('Name')
```

1.1 Q1

[6]:
```
q1_df = pd.merge(student_df, staff_df, how = 'left', left_index= True,␣
 ↪right_index = True)
q1_df
```

[6]:
```
            School            Role
Name
James     Business          Grader
Mike           Law             NaN
Sally  Engineering  Course liasion
```

pd.merge(staff_df, student_df, how = 'right', left_index = False, right_index = True) will throw a merge error as there is no index to merge by if left_index = false.

1.2 Q2

x = 1 since the operation is occurring across **all rows** which is technically a **column operation**. Since frames are columns in the original database, y = 1

[12]:
```
frames = ['School']
student_df['Modified'] = student_df[frames].apply(lambda z: z*2, axis=1)
result_df = student_df.drop(frames,axis=1)
```

```
result_df
```

[12]:
```
                        Modified
Name
James          BusinessBusiness
Mike                     LawLaw
Sally   EngineeringEngineering
```

1.3 Q3

[14]:
```
import pandas as pd

df = pd.DataFrame(['A+', 'A', 'A-', 'B+', 'B', 'B-', 'C+', 'C', 'C-', 'D+',␣
 ↪'D'], index=['excellent', 'excellent', 'excellent', 'good', 'good', 'good',␣
 ↪'ok', 'ok', 'ok', 'poor', 'poor'], columns = ['Grades'])
my_categories= pd.
 ↪CategoricalDtype(categories=['D','D+','C-','C','C+','B-','B','B+','A-','A','A+'],␣
 ↪ordered = True)
grades = df['Grades'].astype(my_categories)
result = grades[(grades>'B') & (grades<'A')]
result
```

[14]:
```
excellent    A-
good         B+
Name: Grades, dtype: category
Categories (11, object): [D < D+ < C- < C ... B+ < A- < A < A+]
```

1.4 Q4

df.pivot_table(values = ‘score’, index = ‘country’, columns = ‘Rank_Level’, aggfunc = [np.median], margins = True) is the answer. The index is by country (indices are bolded), and margin = True is required, as there is an “All” column is a marginal value.

1.5 Q5

11/29/2019 is the 4th day of the week (indexed 0), Month ends on 30th Nov(11/30/2019 - 5th day of week). Since weekdays are indexed from 0, .weekday() will return a value of **5**.

[15]:
```
import pandas as pd
(pd.Timestamp('11/29/2019') + pd.offsets.MonthEnd()).weekday()
```

[15]:
```
5
```

1.6 Q6

Answer is either df.groupby(group_key).apply(filling_mean) or df.groupby(group_key).transform(filling_mean)

I’m not too sure about this though.

1.7 Q7

result_df = pd.merge(staff_df, student_df, how = ‘right’, on = [‘First Name’, ‘Last Name’])
Now student df is the set on the right, so “right” is used. It is **not a full outer join**.

[17]:
```
student_df = student_df.reset_index()
staff_df = staff_df.reset_index()
```

[19]:
```
q7_df = pd.merge(staff_df, student_df, how = 'right', on = ['Name'])
q7_df
```

[19]:
```
    Name          Role  index       School                Modified
0  Sally  Course liasion      2  Engineering  EngineeringEngineering
1  James        Grader      0     Business        BusinessBusiness
2   Mike           NaN      1          Law                  LawLaw
```

1.8 Q8

Since the dataframe contains missing values, np.nanmean, np.nanstd have to be used omn the column reviews_per_month.
Since it is required to group by different review scores, we have to call groupby(‘review_scores_value’). Hence, df.groupby(‘review_scores_value’).agg)({‘name’:len, ‘reviews_per_month’:(np.nanmean,np.nanstd)})

1.9 Q9

Since the granularity of the period is set to M, the day parameter may be left out. So the expression pd.Period(‘01/12/2019’, ‘M’) evaluates to 01/2019.
Hence, adding 5 to that will give Period(‘2019-06’, ‘M’)

[20]:
```
import pandas as pd
pd.Period('01/12/2019', 'M') + 5
```

[20]:
```
Period('2019-06', 'M')
```

1.10 Q10

df[‘vegetable’] will fail with a Key Error as it is an element in the table, but not a column name.

[6]:
```
import pandas as pd
q10_df = pd.DataFrame([{'name': 'apple', 'class': 'fruit', 'avg calories per␣
 ↪unit': 95.0},
                       {'name': 'mango', 'class': 'fruit', 'avg calories per␣
 ↪unit': 202.0},
                      {'name': 'potato', 'class': 'vegetable', 'avg calories␣
 ↪per unit': 164.0},
                      {'name': 'onion', 'class': 'vegetable', 'avg calories per␣
 ↪unit': float("NaN")},
                      {'name': 'brocolli', 'class': 'vegetable', 'avg calories␣
 ↪per unit':207.0}])
q10_df.dropna()
```

```
[6]:        name      class  avg calories per unit
     0     apple      fruit                   95.0
     1     mango      fruit                  202.0
     2    potato  vegetable                  164.0
     4  brocolli  vegetable                  207.0
```

```
[29]: q10_df.groupby('vegetable')
```

```
       ␣
 ↪---------------------------------------------------------------------------

       KeyError                                  Traceback (most recent call␣
 ↪last)

       <ipython-input-29-ac59c1cf0651> in <module>
    ----> 1 q10_df.groupby('vegetable')


       /opt/conda/lib/python3.7/site-packages/pandas/core/generic.py in␣
 ↪groupby(self, by, axis, level, as_index, sort, group_keys, squeeze, observed,␣
 ↪**kwargs)
      7894             squeeze=squeeze,
      7895             observed=observed,
   -> 7896             **kwargs
      7897         )
      7898


       /opt/conda/lib/python3.7/site-packages/pandas/core/groupby/groupby.py in␣
 ↪groupby(obj, by, **kwds)
      2476         raise TypeError("invalid type: {}".format(obj))
      2477
   -> 2478     return klass(obj, by, **kwds)


       /opt/conda/lib/python3.7/site-packages/pandas/core/groupby/groupby.py in␣
 ↪__init__(self, obj, keys, axis, level, grouper, exclusions, selection,␣
 ↪as_index, sort, group_keys, squeeze, observed, **kwargs)
       389                 sort=sort,
       390                 observed=observed,
   --> 391                 mutated=self.mutated,
       392             )
       393


       /opt/conda/lib/python3.7/site-packages/pandas/core/groupby/grouper.py in␣
 ↪_get_grouper(obj, key, axis, level, sort, observed, mutated, validate)
```

```
    619                     in_axis, name, level, gpr = False, None, gpr, None
    620                 else:
--> 621                     raise KeyError(gpr)
    622             elif isinstance(gpr, Grouper) and gpr.key is not None:
    623                 # Add key to exclusions


KeyError: 'vegetable'
```

[30]:
```
q10_df.groupby('class', axis = 0)
```

[30]:
```
<pandas.core.groupby.generic.DataFrameGroupBy object at 0x7fd79b32e470>
```

[31]:
```
q10_df.groupby('class')
```

[31]:
```
<pandas.core.groupby.generic.DataFrameGroupBy object at 0x7fd7ab4462e8>
```

[32]:
```
q10_df.groupby(['class','avg calories per unit'])
```

[32]:
```
<pandas.core.groupby.generic.DataFrameGroupBy object at 0x7fd7ab42bd30>
```

[ ]:
```

```