3-23-17 Data Aggre...

FINISHED

```
%pyspark
from pandas import Series, DataFrame
import pandas as pd
import numpy as np
people = DataFrame(np.random.randn(5,5), columns=['a','b','c','d','e'], index=['Joe','Steve',
people.ix[2:3, ['b','c']] = np.nan # Add a few NA values
people
```

```
              a         b         c         d         e
Joe    -0.000421  1.373888  0.157108 -0.639893 -0.535251
Steve  -0.752337 -0.531799 -0.426532  0.553767  1.140440
Wes     1.042815       NaN       NaN -0.349171  0.669340
Jim    -0.361800  0.268390 -0.794215  1.105553  0.833369
Travis  1.032759 -0.152801  1.406444 -0.598631  0.288329
```

Took 23 sec. Last updated by anonymous at March 23 2017, 6:04:48 PM.

FINISHED

```
%pyspark
mapping = {'a': 'red', 'b': 'red', 'c': 'blue', 'd': 'blue', 'e': 'red', 'f': 'orange'}

by_column = people.groupby(mapping, axis=1)
by_column.sum()

map_series = Series(mapping)
map_series

people.groupby(map_series, axis=1).count()
```

```
        blue  red
Joe        2    3
Steve      2    3
Wes        1    2
Jim        2    3
Travis     2    3
```

Took 0 sec. Last updated by anonymous at March 23 2017, 6:10:08 PM.

FINISHED

```
%pyspark
people.groupby(len).sum()

key_list = ['one', 'one', 'one', 'two', 'two']
people.groupby([len, key_list]).min()
```

```
              a         b         c         d         e
3 one -0.000421  1.373888  0.157108 -0.639893 -0.535251
  two -0.361800  0.268390 -0.794215  1.105553  0.833369
5 one -0.752337 -0.531799 -0.426532  0.553767  1.140440
6 two  1.032759 -0.152801  1.406444 -0.598631  0.288329
```

Took 0 sec. Last updated by anonymous at March 23 2017, 6:11:00 PM.

```
%pyspark
columns = pd.MultiIndex.from_arrays([['US', 'US', 'US', 'JP', 'JP'], [1, 3, 5, 1, 3]], names=
hier_df = DataFrame(np.random.randn(4, 5), columns=columns)
hier_df
```

FINISHED

```
cty          US                            JP
tenor         1         3         5         1         3
0     -1.041323 -0.233942  0.139431  0.315995 -1.785792
1      1.226120  0.078031 -0.025212 -0.405954  0.056555
2     -0.006141  0.135968 -0.655892 -0.043779 -1.470567
3     -1.038302 -1.855839 -0.091299 -0.460940  0.867787
```

Took 0 sec. Last updated by anonymous at March 23 2017, 6:12:42 PM.

```
%pyspark
hier_df.groupby(level='cty', axis=1).count()
```

FINISHED

```
cty  JP  US
0     2   3
1     2   3
2     2   3
3     2   3
```

Took 0 sec. Last updated by anonymous at March 23 2017, 6:13:29 PM.

```
%pyspark
df = DataFrame({'key1' : ['a','a','b','b','a'],
                'key2' : ['one','two','one','two','one'],
                'data1' : np.random.randn(5),
                'data2' : np.random.randn(5)})



df
```

FINISHED

```
      data1     data2 key1 key2
0  1.582854  0.010503    a  one
1 -0.155180 -1.299838    a  two
2 -0.177737 -1.156954    b  one
3 -1.196331  0.130139    b  two
4 -1.299236  1.304950    a  one
```

Took 0 sec. Last updated by anonymous at March 23 2017, 6:22:37 PM.

```
%pyspark
grouped = df.groupby('key1')
grouped['data1'].quantile(0.9)
```

FINISHED

```
key1
a    1.516743
b    0.798966
Name: data1, dtype: float64
```

Took 0 sec. Last updated by anonymous at March 23 2017, 6:22:10 PM.

```
%pyspark
```

FINISHED

```
def peak_to_peak(arr): return arr.max() - arr.min()
grouped.agg(peak_to_peak)
```

```
         data1     data2
key1
a     2.943037  2.462945
b     2.717135  0.044332
```

Took 0 sec. Last updated by anonymous at March 23 2017, 6:22:54 PM.

FINISHED

```
%pyspark
grouped.describe()
```

```
                data1     data2
key1
a    count  3.000000  3.000000
     mean   0.329400  1.369113
     std    1.472172  1.306333
     min   -1.116799  0.389290
     25%   -0.419019  0.627552
     50%    0.278761  0.865815
     75%    1.052500  1.859025
     max    1.826239  2.852234
b    count  2.000000  2.000000
     mean  -0.287888 -0.537040
     std    1.921304  0.031348
     min   -1.646456 -0.559207
     25%   -0.967172 -0.548123
     50%   -0.287888 -0.537040
     75%    0.391395 -0.525957
     max    1.070679 -0.514874
```

Took 0 sec. Last updated by anonymous at March 23 2017, 6:35:45 PM.

FINISHED

```
%pyspark
tips = pd.read_csv('/Users/geoffnes/Downloads/tips.csv')
```

Took 0 sec. Last updated by anonymous at March 23 2017, 6:39:00 PM.

FINISHED

```
%pyspark
tips['tip_pct'] = tips['tip'] / tips['total_bill']
tips[:6]
```

```
   total_bill   tip     sex smoker  day    time  size   tip_pct
0       16.99  1.01  Female     No  Sun  Dinner     2  0.059447
1       10.34  1.66    Male     No  Sun  Dinner     3  0.160542
2       21.01  3.50    Male     No  Sun  Dinner     3  0.166587
3       23.68  3.31    Male     No  Sun  Dinner     2  0.139780
4       24.59  3.61  Female     No  Sun  Dinner     4  0.146808
5       25.29  4.71    Male     No  Sun  Dinner     4  0.186240
```

Took 0 sec. Last updated by anonymous at March 23 2017, 6:39:16 PM.

FINISHED

```
%pyspark
grouped = tips.groupby(['sex','smoker'])
grouped_pct = grouped['tip_pct']
```

```
grouped_pct.agg('mean')
```

```
sex     smoker
Female  No        0.156921
        Yes       0.182150
Male    No        0.160669
        Yes       0.152771
Name: tip_pct, dtype: float64
```

Took 0 sec. Last updated by anonymous at March 23 2017, 6:39:35 PM.

```
%pyspark
grouped_pct.agg(['mean','std',peak_to_peak])
```

FINISHED

```
                   mean       std  peak_to_peak
sex    smoker
Female No      0.156921  0.036421      0.195876
       Yes     0.182150  0.071595      0.360233
Male   No      0.160669  0.041849      0.220186
       Yes     0.152771  0.090588      0.674707
```

Took 0 sec. Last updated by anonymous at March 23 2017, 6:41:43 PM.

```
%pyspark
grouped_pct.agg([('foo','mean'),('bar',np.std)])
```

FINISHED

```
                    foo       bar
sex    smoker
Female No      0.156921  0.036421
       Yes     0.182150  0.071595
Male   No      0.160669  0.041849
       Yes     0.152771  0.090588
```

Took 0 sec. Last updated by anonymous at March 23 2017, 6:41:57 PM.

```
%pyspark
functions = ['count','mean','max']
result = grouped['tip_pct','total_bill'].agg(functions)
result
```

FINISHED

```
              tip_pct                     total_bill
                count      mean       max      count       mean    max
sex    smoker
Female No          54  0.156921  0.252672         54  18.105185  35.83
       Yes         33  0.182150  0.416667         33  17.977879  44.30
Male   No          97  0.160669  0.291990         97  19.791237  48.33
       Yes         60  0.152771  0.710345         60  22.284500  50.81
```

Took 0 sec. Last updated by anonymous at March 23 2017, 6:42:17 PM.

```
%pyspark
result['tip_pct']
```

FINISHED

```
               count      mean       max
sex    smoker
Female No         54  0.156921  0.252672
```

```
       Yes        33  0.182150  0.416667
Male   No         97  0.160669  0.291990
       Yes        60  0.152771  0.710345
```

Took 0 sec. Last updated by anonymous at March 23 2017, 6:42:32 PM.

FINISHED

```
%pyspark
ftuples = [('Average', 'mean'), ('Standard Dev.', np.var)]
grouped['tip_pct','total_bill'].agg(ftuples)
```

```
                tip_pct               total_bill
                Average Standard Dev.    Average Standard Dev.
sex    smoker
Female No      0.156921      0.001327  18.105185     53.092422
       Yes     0.182150      0.005126  17.977879     84.451517
Male   No      0.160669      0.001751  19.791237     76.152961
       Yes     0.152771      0.008206  22.284500     98.244673
```

Took 0 sec. Last updated by anonymous at March 23 2017, 6:54:48 PM.

FINISHED

```
%pyspark
grouped.agg({'tip' : np.max, 'size' : 'sum'})
```

```
                tip  size
sex    smoker
Female No       5.2   140
       Yes      6.5    74
Male   No       9.0   263
       Yes     10.0   150
```

Took 0 sec. Last updated by anonymous at March 23 2017, 6:44:16 PM.

FINISHED

```
%pyspark
grouped.agg({'tip_pct' : ['min', 'max', 'mean', 'std'], 'size' : 'sum'})
```

```
                tip_pct                                 size
                    min       max      mean       std  sum
sex    smoker
Female No      0.056797  0.252672  0.156921  0.036421  140
       Yes     0.056433  0.416667  0.182150  0.071595   74
Male   No      0.071804  0.291990  0.160669  0.041849  263
       Yes     0.035638  0.710345  0.152771  0.090588  150
```

Took 0 sec. Last updated by anonymous at March 23 2017, 6:48:45 PM.

FINISHED

```
%pyspark
# Returning aggregated data in unindexed form
tips.groupby(['sex','smoker'], as_index=False).mean()
```

```
      sex smoker  total_bill       tip      size   tip_pct
0  Female     No   18.105185  2.773519  2.592593  0.156921
1  Female    Yes   17.977879  2.931515  2.242424  0.182150
2    Male     No   19.791237  3.113402  2.711340  0.160669
3    Male    Yes   22.284500  3.051167  2.500000  0.152771
```

Took 0 sec. Last updated by anonymous at March 23 2017, 6:52:01 PM.

```
%pyspark
# Group-wise operations and transformations
df
```

FINISHED

```
      data1     data2 key1 key2
0  1.582854  0.010503    a  one
1 -0.155180 -1.299838    a  two
2 -0.177737 -1.156954    b  one
3 -1.196331  0.130139    b  two
4 -1.299236  1.304950    a  one
```

Took 0 sec. Last updated by anonymous at March 23 2017, 6:58:19 PM.

```
%pyspark
k1_means = df.groupby('key1').mean().add_prefix('mean_')
k1_means
```

FINISHED

```
      mean_data1  mean_data2
key1
a       0.042813    0.005205
b      -0.687034   -0.513408
```

Took 0 sec. Last updated by anonymous at March 23 2017, 6:58:32 PM.

```
%pyspark
# Returning aggregated data in unindexed form
tips.groupby(['sex','smoker'], as_index=True).mean()
```

FINISHED

```
               total_bill       tip      size   tip_pct
sex    smoker
Female No       18.105185  2.773519  2.592593  0.156921
       Yes      17.977879  2.931515  2.242424  0.182150
Male   No       19.791237  3.113402  2.711340  0.160669
       Yes      22.284500  3.051167  2.500000  0.152771
```

Took 0 sec. Last updated by anonymous at March 23 2017, 6:59:23 PM.

```
%pyspark
pd.merge(df, k1_means, left_on='key1', right_index=True)
```

FINISHED

```
      data1     data2 key1 key2  mean_data1  mean_data2
0  1.582854  0.010503    a  one    0.042813    0.005205
1 -0.155180 -1.299838    a  two    0.042813    0.005205
4 -1.299236  1.304950    a  one    0.042813    0.005205
2 -0.177737 -1.156954    b  one   -0.687034   -0.513408
3 -1.196331  0.130139    b  two   -0.687034   -0.513408
```

Took 0 sec. Last updated by anonymous at March 23 2017, 7:00:31 PM.

```
%pyspark
key = ['one', 'two', 'one', 'two', 'one']
people.groupby(key).mean()
```

FINISHED

```
            a         b         c         d         e
one  0.691717  0.610544  0.781776 -0.529232  0.140806
```

```
two -0.557068 -0.131705 -0.610373  0.829660  0.986905
```

Took 0 sec. Last updated by anonymous at March 23 2017, 7:02:25 PM.

FINISHED

```
%pyspark
people.groupby(key).transform(np.mean)
```

```
              a         b         c         d         e
Joe     0.691717  0.610544  0.781776 -0.529232  0.140806
Steve  -0.557068 -0.131705 -0.610373  0.829660  0.986905
Wes     0.691717  0.610544  0.781776 -0.529232  0.140806
Jim    -0.557068 -0.131705 -0.610373  0.829660  0.986905
Travis  0.691717  0.610544  0.781776 -0.529232  0.140806
```

Took 0 sec. Last updated by anonymous at March 23 2017, 7:04:16 PM.

FINISHED

```
%pyspark
def demean(arr): return arr - arr.mean()

demeaned = people.groupby(key).transform(demean)
demeaned
```

```
              a         b         c         d         e
Joe    -0.692138  0.763344 -0.624668 -0.110662 -0.676057
Steve  -0.195269 -0.400094  0.183841 -0.275893  0.153536
Wes     0.351097       NaN       NaN  0.180061  0.528534
Jim     0.195269  0.400094 -0.183841  0.275893 -0.153536
Travis  0.341041 -0.763344  0.624668 -0.069399  0.147523
```

Took 0 sec. Last updated by anonymous at March 23 2017, 7:04:28 PM.

FINISHED

```
%pyspark
demeaned.groupby(key).mean()
```

```
                a    b    c             d             e
one  0.000000e+00  0.0  0.0  5.551115e-17  0.000000e+00
two  2.775558e-17  0.0  0.0  0.000000e+00 -5.551115e-17
```

Took 0 sec. Last updated by anonymous at March 23 2017, 7:04:39 PM.

FINISHED

```
%pyspark
# Apply general split-apply-combine

def top(df, n=5, column='tip_pct'): return df.sort_index(by=column)[-n:]

top(tips, n=6)
```

```
/var/folders/h4/2z0hx5wn6qzdby5b01n5x4640000gn/T/zeppelin_pyspark-4958191010723368530.py:1: Fu
tureWarning: by argument to sort_index is deprecated, pls use .sort_values(by=...)
  #
     total_bill   tip     sex smoker  day    time  size   tip_pct
109       14.31  4.00  Female    Yes  Sat  Dinner     2  0.279525
183       23.17  6.50    Male    Yes  Sun  Dinner     4  0.280535
232       11.61  3.39    Male     No  Sat  Dinner     2  0.291990
67         3.07  1.00  Female    Yes  Sat  Dinner     1  0.325733
178        9.60  4.00  Female    Yes  Sun  Dinner     2  0.416667
```

```
172        7.25  5.15    Male    Yes  Sun  Dinner     2  0.710345
```

Took 0 sec. Last updated by anonymous at March 23 2017, 7:04:56 PM.

```
%pyspark
tips.groupby('smoker').apply(top)
```

FINISHED

```
            total_bill   tip     sex smoker   day    time  size   tip_pct
smoker
No     88        24.71  5.85    Male     No  Thur   Lunch     2  0.236746
       185       20.69  5.00    Male     No   Sun  Dinner     5  0.241663
       51        10.29  2.60  Female     No   Sun  Dinner     2  0.252672
       149        7.51  2.00    Male     No  Thur   Lunch     2  0.266312
       232       11.61  3.39    Male     No   Sat  Dinner     2  0.291990
Yes    109       14.31  4.00  Female    Yes   Sat  Dinner     2  0.279525
       183       23.17  6.50    Male    Yes   Sun  Dinner     4  0.280535
       67         3.07  1.00  Female    Yes   Sat  Dinner     1  0.325733
       178        9.60  4.00  Female    Yes   Sun  Dinner     2  0.416667
       172        7.25  5.15    Male    Yes   Sun  Dinner     2  0.710345
```

Took 0 sec. Last updated by anonymous at March 23 2017, 7:05:09 PM.

```
%pyspark
tips.groupby(['smoker','day']).apply(top, n=1, column='total_bill')
```

FINISHED

```
                 total_bill    tip     sex smoker   day    time  size  \
smoker day
No     Fri  94        22.75   3.25  Female     No   Fri  Dinner     2
       Sat  212       48.33   9.00    Male     No   Sat  Dinner     4
       Sun  156       48.17   5.00    Male     No   Sun  Dinner     6
       Thur 142       41.19   5.00    Male     No  Thur   Lunch     5
Yes    Fri  95        40.17   4.73    Male    Yes   Fri  Dinner     4
       Sat  170       50.81  10.00    Male    Yes   Sat  Dinner     3
       Sun  182       45.35   3.50    Male    Yes   Sun  Dinner     3
       Thur 197       43.11   5.00  Female    Yes  Thur   Lunch     4
                  tip_pct
smoker day
No     Fri  94   0.142857
       Sat  212  0.186220
       Sun  156  0.103799
       Thur 142  0.121389
Yes    Fri  95   0.117750
       Sat  170  0.196812
```

Took 0 sec. Last updated by anonymous at March 23 2017, 7:05:36 PM.

```
%pyspark
result = tips.groupby('smoker')['tip_pct'].describe()
result
```

FINISHED

```
smoker
No      count    151.000000
        mean       0.159328
        std        0.039910
        min        0.056797
        25%        0.136906
        50%        0.155625
        75%        0.185014
        max        0.291990
Yes     count     93.000000
        mean       0.163196
        std        0.085119
        min        0.035638
        25%        0.106771
        50%        0.153846
        75%        0.195059
        max        0.710345
```

Took 0 sec. Last updated by anonymous at March 23 2017, 7:05:51 PM.

```
%pyspark
result.unstack('smoker')
```

FINISHED

```
smoker          No        Yes
count   151.000000  93.000000
mean      0.159328   0.163196
std       0.039910   0.085119
min       0.056797   0.035638
25%       0.136906   0.106771
50%       0.155625   0.153846
75%       0.185014   0.195059
max       0.291990   0.710345
```

Took 0 sec. Last updated by anonymous at March 23 2017, 7:20:43 PM.

```
%pyspark
f = lambda x: x.describe()
grouped.apply(f)
```

FINISHED

```
                     total_bill        tip       size    tip_pct
sex    smoker
Female No     count   54.000000  54.000000  54.000000  54.000000
              mean    18.105185   2.773519   2.592593   0.156921
              std      7.286455   1.128425   1.073146   0.036421
              min      7.250000   1.000000   1.000000   0.056797
              25%     12.650000   2.000000   2.000000   0.139708
              50%     16.690000   2.680000   2.000000   0.149691
              75%     20.862500   3.437500   3.000000   0.181630
              max     35.830000   5.200000   6.000000   0.252672
       Yes    count   33.000000  33.000000  33.000000  33.000000
              mean    17.977879   2.931515   2.242424   0.182150
              std      9.189751   1.219916   0.613917   0.071595
              min      3.070000   1.000000   1.000000   0.056433
              25%     12.760000   2.000000   2.000000   0.152439
              50%     16.270000   2.880000   2.000000   0.173913
              75%     22.120000   3.500000   2.000000   0.198216
              max     44.300000   6.500000   4.000000   0.416667
```

Took 0 sec. Last updated by anonymous at March 23 2017, 7:20:59 PM.

```
%pyspark
# Quantile and bucket analysis
```

FINISHED

```
frame = DataFrame({'data1': np.random.randn(1000), 'data2': np.random.randn(1000)})
factor = pd.cut(frame.data1, 4)
factor[:10]
```

```
0     (0.349, 2.355]
1    (-1.656, 0.349]
2    (-1.656, 0.349]
3    (-1.656, 0.349]
4     (0.349, 2.355]
5    (-1.656, 0.349]
6    (-1.656, 0.349]
7    (-3.67, -1.656]
8    (-1.656, 0.349]
9    (-1.656, 0.349]
Name: data1, dtype: category
Categories (4, object): [(-3.67, -1.656] < (-1.656, 0.349] < (0.349, 2.355] < (2.355, 4.361]]
```

Took 0 sec. Last updated by anonymous at March 23 2017, 7:21:25 PM.

FINISHED

```
%pyspark
def get_stats(group): return {'min': group.min(), 'max': group.max(), 'count': group.count(),

grouped = frame.data2.groupby(factor)
grouped.apply(get_stats).unstack()
```

```
                 count       max      mean       min
data1
(-3.67, -1.656]   61.0  2.609021  0.150625 -1.725483
(-1.656, 0.349]  586.0  3.080514 -0.045051 -2.920185
(0.349, 2.355]   345.0  3.032882  0.105853 -2.644108
(2.355, 4.361]     8.0  0.999218 -0.595542 -2.084712
```

Took 0 sec. Last updated by anonymous at March 23 2017, 7:21:38 PM.

FINISHED

```
%pyspark
# Return quantile numbers
grouping = pd.qcut(frame.data1, 10, labels=False)

grouped = frame.data2.groupby(grouping)
grouped.apply(get_stats).unstack()
```

```
       count       max      mean       min
data1
0      100.0  2.609021  0.070780 -2.017273
1      100.0  2.345962 -0.031875 -2.830138
2      100.0  2.105239 -0.178331 -2.920185
3      100.0  2.499579 -0.045897 -2.307751
4      100.0  1.958704 -0.070020 -2.354367
5      100.0  3.080514  0.126758 -2.587731
6      100.0  2.489365 -0.051686 -2.158392
7      100.0  2.301338 -0.008344 -2.644108
8      100.0  2.712539  0.183017 -2.322767
9      100.0  3.032882  0.151029 -2.440249
```

Took 0 sec. Last updated by anonymous at March 23 2017, 7:21:59 PM.

READY

```
%pyspark
```