DATA ANALYSIS

USING PYTHON, R

Ravi Maggon, Senior Member Technical | D.E. Shaw & Co.

AGENDA

1. Goal/Achievement from this session
2. Introduction to Data Analysis
3. Using data analytic toolkits in python
4. Data Analysis
5. Data Collection
6. Seasonality removal using Holt Winter analysis

GOAL OF THIS SESSION

As I deal in financial sector, throughout my session, I will walk you through some real world challenges in trading firm.

1. How data analysis works in financial sector?
2. Developing a data centric financial model using tools/concepts of machine learning.

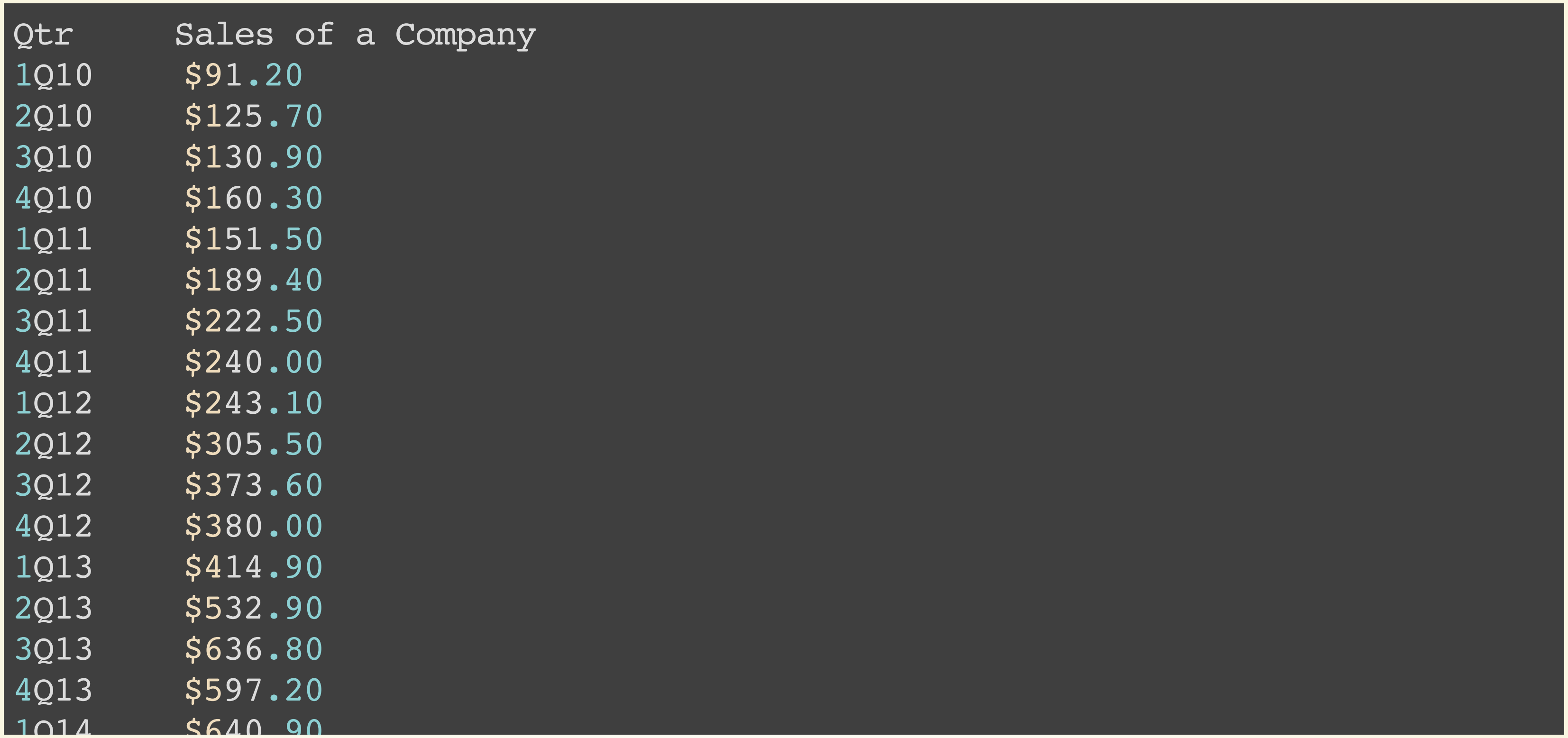
DATA ANALYSIS IN FINANCIAL MODEL

INPUTS

SALES NUMBER/QUARTER REVENUE

INPUTS

SALES NUMBER/QUARTER REVENUE



```
Qtr     Sales of a Company
1Q10     $91.20
2Q10     $125.70
3Q10     $130.90
4Q10     $160.30
1Q11     $151.50
2Q11     $189.40
3Q11     $222.50
4Q11     $240.00
1Q12     $243.10
2Q12     $305.50
3Q12     $373.60
4Q12     $380.00
1Q13     $414.90
2Q13     $532.90
3Q13     $636.80
4Q13     $597.20
1Q14     $640.90
```

INPUTS

RAW DATA

INPUTS

RAW DATA

```
Qtr       Raw Data
1Q10     12
2Q10     19
3Q10     32
4Q10     24
1Q11     28
2Q11     60
3Q11     132
4Q11     198
1Q12     228
2Q12     212
3Q12     321
4Q12     418
1Q13     661
2Q13     615
3Q13     628
4Q13     631
1Q14     627
```

DATA ANALYSIS IN FINANCE DOMAIN

DATA ANALYSIS (RAW DATA + SALES) = COMING QUARTER REVENUE ESTIMATION

DATA ANALYSIS

COMPONENTS OF DATA ANALYSIS

- Data Inspection
- Data Cleaning
- Transformation
- Data Modeling

DATA INSPECTION

- First step in data analysis.
- Check available data sources
- Check available data
- Select data which you feel would yield result
- Proceed with the dataset

DATA CLEANING

- Data can't be used directly in most cases
- Data contain noise
- Example of reviews: Spam Reviews

DATA TRANSFORMATION

- Converting data into required form

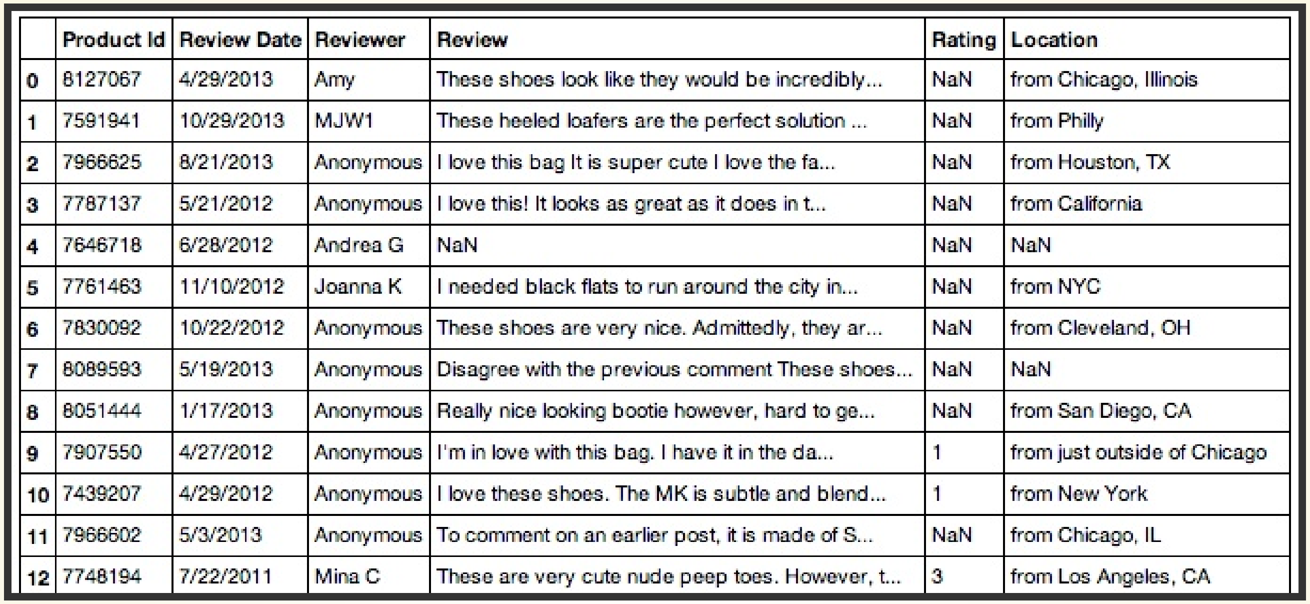
DATA MODELLING

- Discovering useful information
- Suggesting conclusions
- Supporting decision making

EXAMPLES

- Choose website like Amazon.com
- Get the reviews for a given brand
- Get the daily reviews count
- Apply components of data analysis
- Get the sales number of company
- Apply data analytics tools

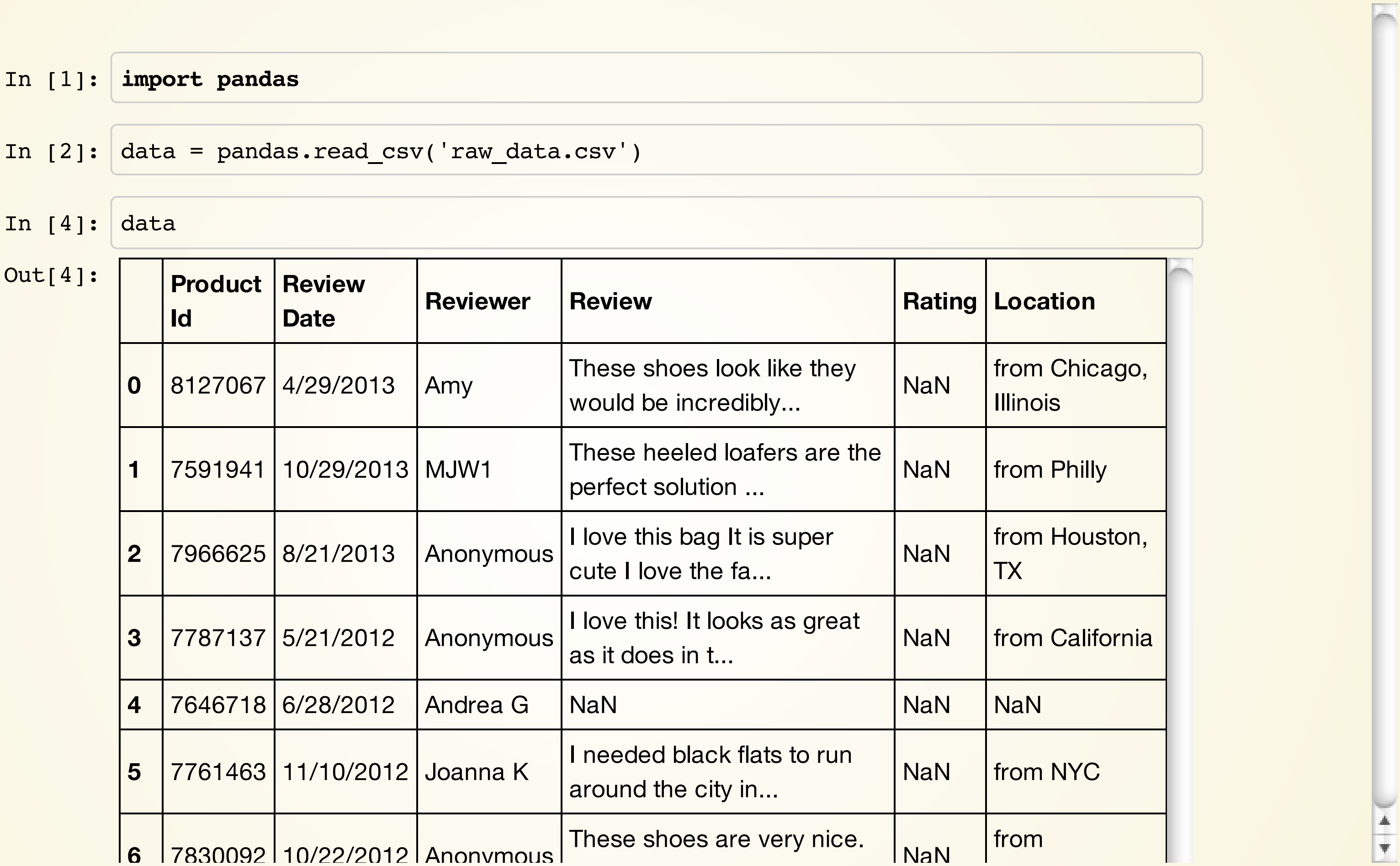
EXAMPLES



	Product Id	Review Date	Reviewer	Review	Rating	Location
0	8127067	4/29/2013	Amy	These shoes look like they would be incredibly...	NaN	from Chicago, Illinois
1	7591941	10/29/2013	MJW1	These heeled loafers are the perfect solution ...	NaN	from Philly
2	7966625	8/21/2013	Anonymous	I love this bag It is super cute I love the fa...	NaN	from Houston, TX
3	7787137	5/21/2012	Anonymous	I love this! It looks as great as it does in t...	NaN	from California
4	7646718	6/28/2012	Andrea G	NaN	NaN	NaN
5	7761463	11/10/2012	Joanna K	I needed black flats to run around the city in...	NaN	from NYC
6	7830092	10/22/2012	Anonymous	These shoes are very nice. Admittedly, they ar...	NaN	from Cleveland, OH
7	8089593	5/19/2013	Anonymous	Disagree with the previous comment These shoes...	NaN	NaN
8	8051444	1/17/2013	Anonymous	Really nice looking bootie however, hard to ge...	NaN	from San Diego, CA
9	7907550	4/27/2012	Anonymous	I'm in love with this bag. I have it in the da...	1	from just outside of Chicago
10	7439207	4/29/2012	Anonymous	I love these shoes. The MK is subtle and blend...	1	from New York
11	7966602	5/3/2013	Anonymous	To comment on an earlier post, it is made of S...	NaN	from Chicago, IL
12	7748194	7/22/2011	Mina C	These are very cute nude peep toes. However, t...	3	from Los Angeles, CA

**Show data analysis components on raw_data

EXAMPLES



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

```
In [2]: data = pandas.read_csv('raw_data.csv')
```

```
In [4]: data
```

Out[4]:

	Product Id	Review Date	Reviewer	Review	Rating	Location
0	8127067	4/29/2013	Amy	These shoes look like they would be incredibly...	NaN	from Chicago, Illinois
1	7591941	10/29/2013	MJW1	These heeled loafers are the perfect solution ...	NaN	from Philly
2	7966625	8/21/2013	Anonymous	I love this bag It is super cute I love the fa...	NaN	from Houston, TX
3	7787137	5/21/2012	Anonymous	I love this! It looks as great as it does in t...	NaN	from California
4	7646718	6/28/2012	Andrea G	NaN	NaN	NaN
5	7761463	11/10/2012	Joanna K	I needed black flats to run around the city in...	NaN	from NYC
6	7830092	10/22/2012	Anonymous	These shoes are very nice.	NaN	from

DATA ANALYTIC TOOLKITS

- Numpy
- Pandas
- R

NUMPY/PANDAS

If you are in touch with any topic related to data mining, machine learning, data analysis you will keep hearing about numpy, pandas, rpy, etc. packages in python.

You would be wondering what exactly these things are, where to use them?

NUMPY

Fundamental package for scientific computing with Python.

- Powerful N-dimensional array
- Sophisticated functions

```
import numpy
a = numpy.array([1,2,3,4,5,8,10,15])
a.cumsum()

Output: array([ 1,  3,  6, 10, 15, 23, 33, 48])

type(a)
numpy.ndarray
```

INTRODUCTION TO PANDAS

Pandas is python package built on top of numpy.
Why pandas?

1. Fast
2. Flexible
3. Expressive data structure
4. Easy and intuitive way of working with labelled data

PANDAS DATA STRUCTURES

* Series (1-Dimentional)
* DataFrame (2-Dimentional)

If you understand pandas data frame, R data frame will fall in the same line. Pandas is intended to integrate well with a scientific computing environment.

SERIES

A Series is a single vector of data (like a NumPy array) with an index that labels each element in the vector.

```
import pandas
data = pandas.Series([632, 1638, 569, 115])

Output:
0      632
1     1638
2      569
3      115
dtype: int64

data.values

Output:
array([ 632, 1638,  569,  115])

data.index

Output:
```

SERIES INDEX

If an index is not specified, a default sequence of integers is assigned as the index. A NumPy array comprises the values of the Series, while the index is a pandas Index object.

```
bacteria = pd.Series([632, 1638, 569, 115],
    index=['Firmicutes', 'Proteobacteria', 'Actinobacteria',
           'Bacteroidetes'])

bacteria

Output:

Firmicutes         632
Proteobacteria    1638
Actinobacteria     569
Bacteroidetes      115
dtype: int64

These labels can be used to refer to the values in the Series.
```

DATAFRAME

Inevitably, we want to be able to store, view and manipulate data that is multivariate, where for every index there are multiple fields or columns of data, often of varying data type. A DataFrame is a tabular data structure, encapsulating multiple series like columns in a spreadsheet. Data are stored internally as a 2-dimensional object, but the DataFrame allows us to represent and manipulate higher-dimensional data.

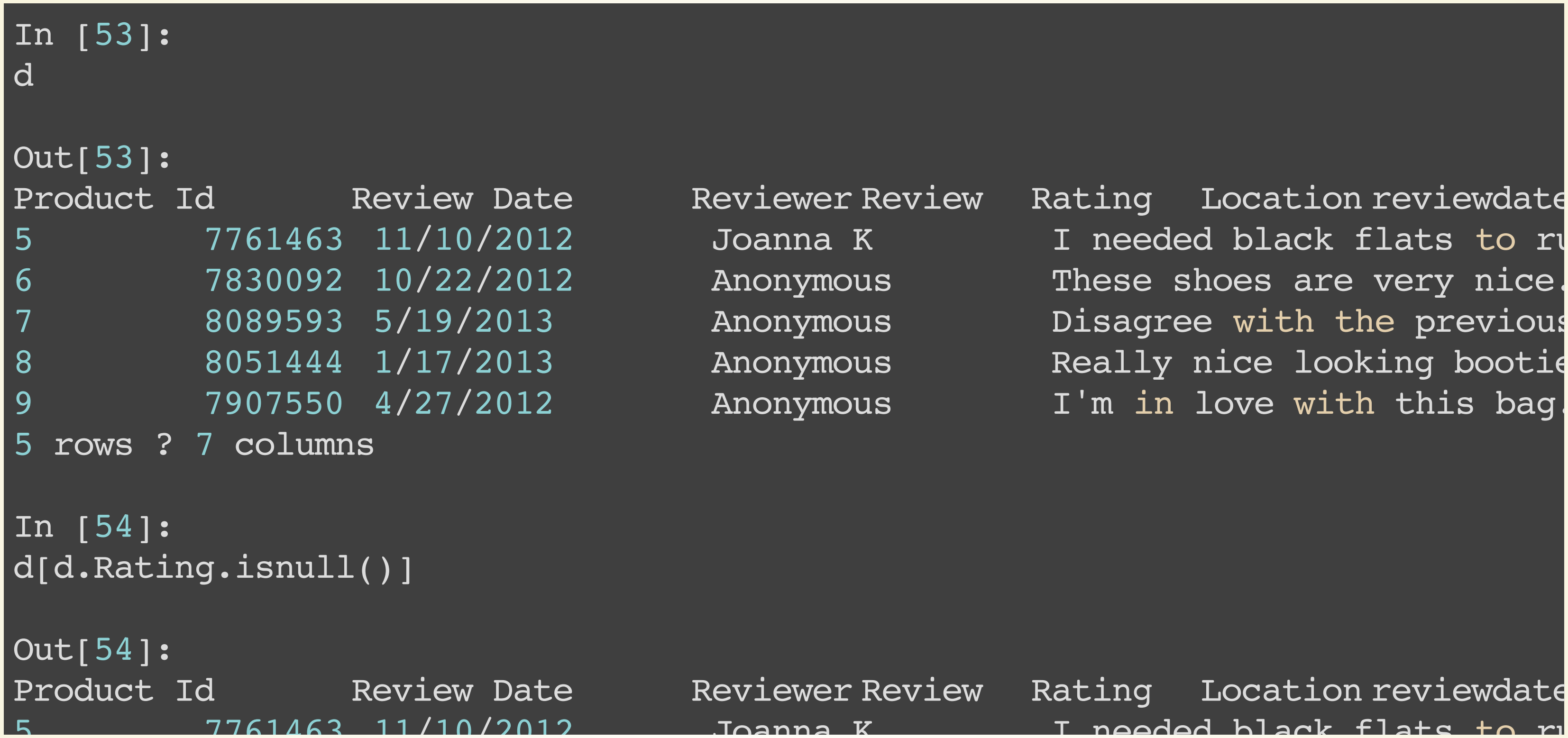
DATAFRAME CODE SNIPPETS

```
In [19]:
data = pd.DataFrame({'value':[632, 1638, 569, 115, 433, 1130, 754, 555],
                     'patient':[1, 1, 1, 1, 2, 2, 2, 2],
                     'phylum':['Firmicutes', 'Proteobacteria', 'Actinobacteri
    'Bacteroidetes', 'Firmicutes', 'Proteobacteria', 'Actinobacteria', 'Bacte
data
Out[19]:
   patient           phylum  value
0        1       Firmicutes    632
1        1  Proteobacteria   1638
2        1  Actinobacteria    569
3        1   Bacteroidetes    115
4        2       Firmicutes    433
5        2  Proteobacteria   1130
6        2  Actinobacteria    754
7        2   Bacteroidetes    555
Notice the DataFrame is sorted by column name. We can change the order by ind
```

BASIC FUNCTIONALITY OF PANDAS

- Check for null values
- Basic available methods
- Modifying columns names
- Dropping columns
- Adding new columns
- Sorting
- Looping over dataframe
- Indexing

NULL VALUES



```
In [53]:
d

Out[53]:
Product Id        Review Date       Reviewer Review   Rating   Location reviewdate
5         7761463  11/10/2012        Joanna K          I needed black flats to ru
6         7830092  10/22/2012        Anonymous         These shoes are very nice.
7         8089593  5/19/2013         Anonymous         Disagree with the previous
8         8051444  1/17/2013         Anonymous         Really nice looking bootie
9         7907550  4/27/2012         Anonymous         I'm in love with this bag.
5 rows ? 7 columns

In [54]:
d[d.Rating.isnull()]

Out[54]:
Product Id        Review Date       Reviewer Review   Rating   Location reviewdate
5         7761463  11/10/2012        Joanna K          I needed black flats to ru
```

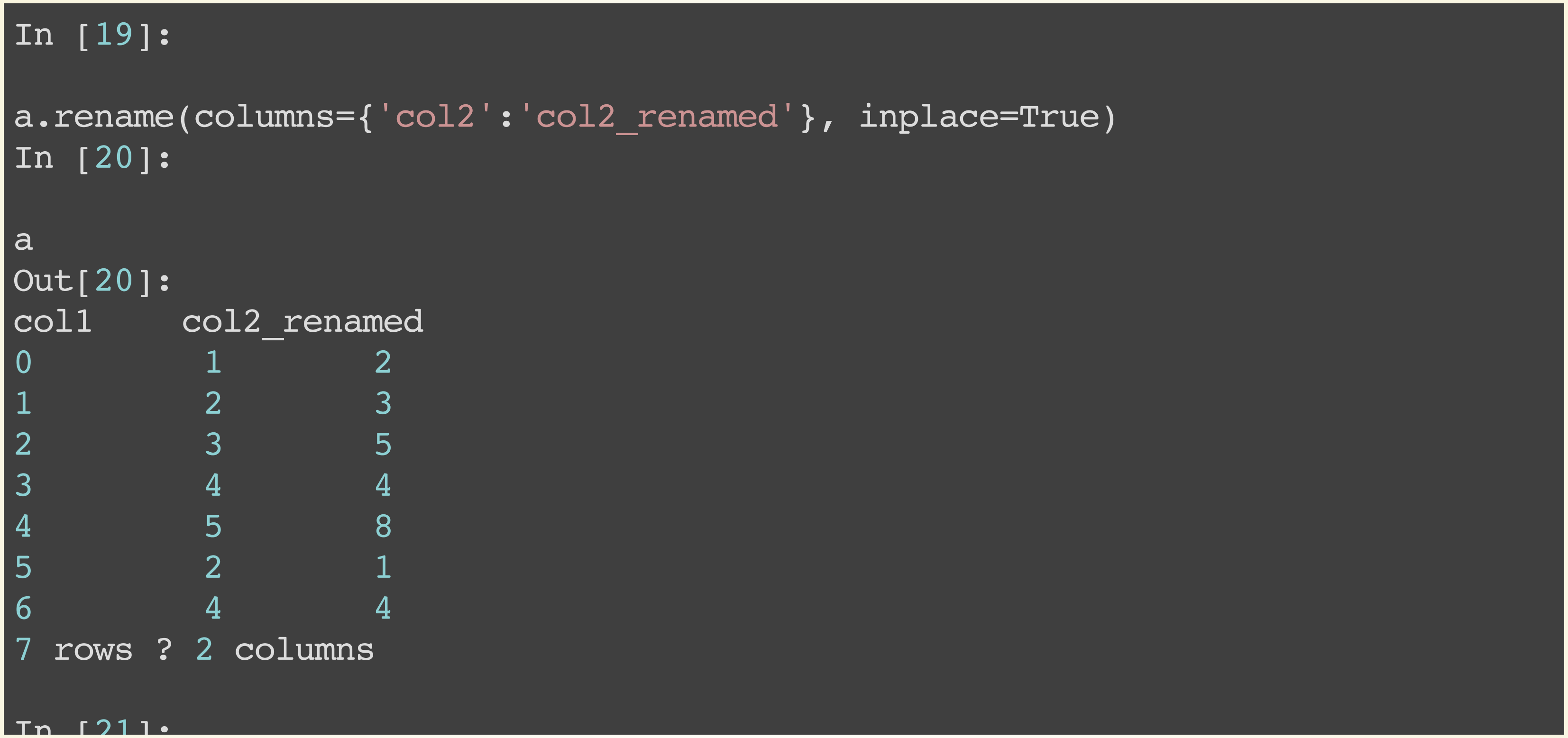
BASIC AVAILABLE METHODS

```
In [1]:

import pandas
In [11]:

a = pandas.DataFrame({'col1': [1,2,3,4,5,2,4], 'col2': [2,3,5,4,8,1,4]})
In [12]:

a
Out[12]:
col1    col2
0         1       2
1         2       3
2         3       5
3         4       4
4         5       8
5         2       1
6         4       4
```

MORE METHODS

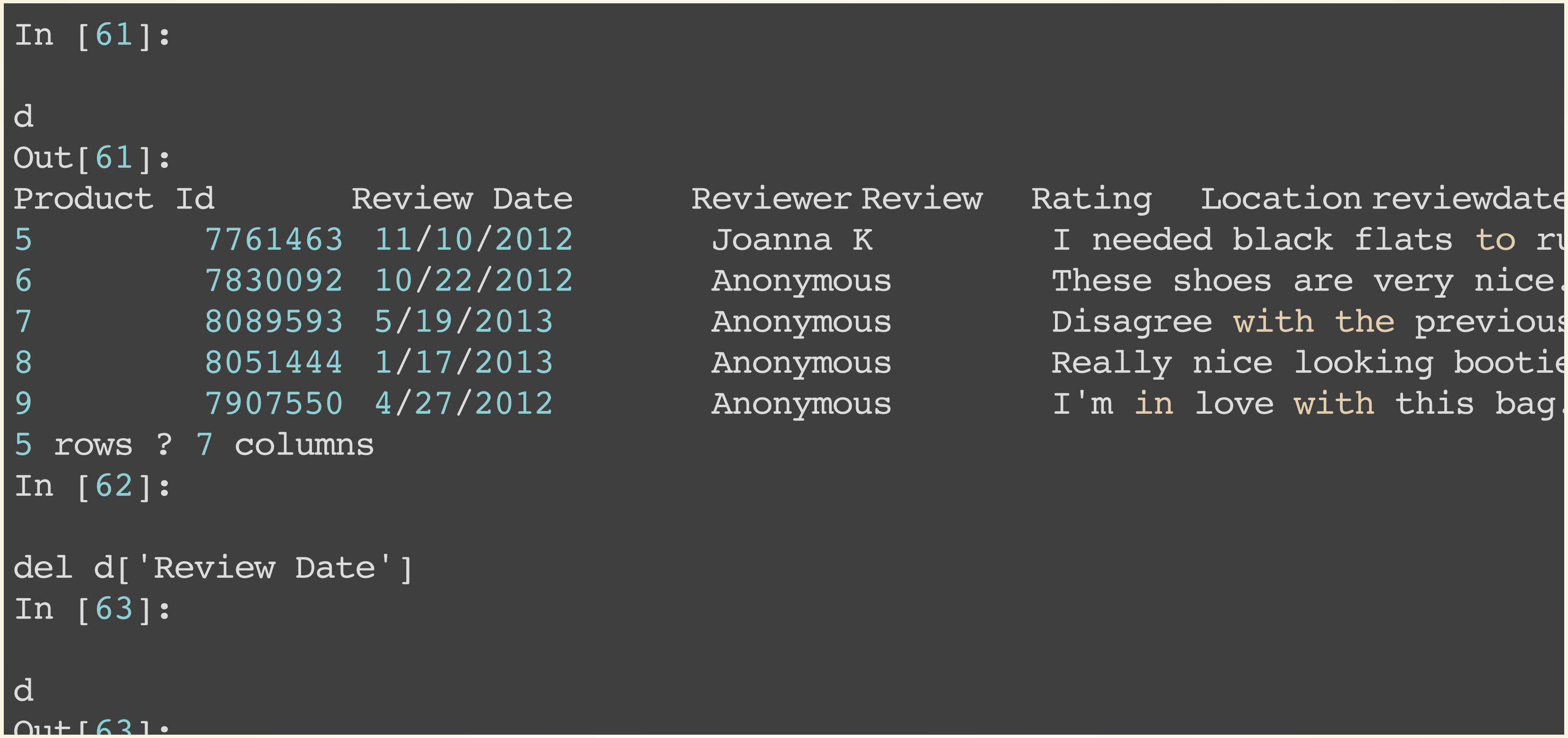
```
In [15]:

a.sum()
Out[15]:
col1    21
col2    27
dtype: int64
In [17]:

a.diff(periods=2)
Out[17]:
col1     col2
0        NaN      NaN
1        NaN      NaN
2         2        3
3         2        1
4         2        3
5        -2       -3
```

MODIFYING COLUMN NAMES



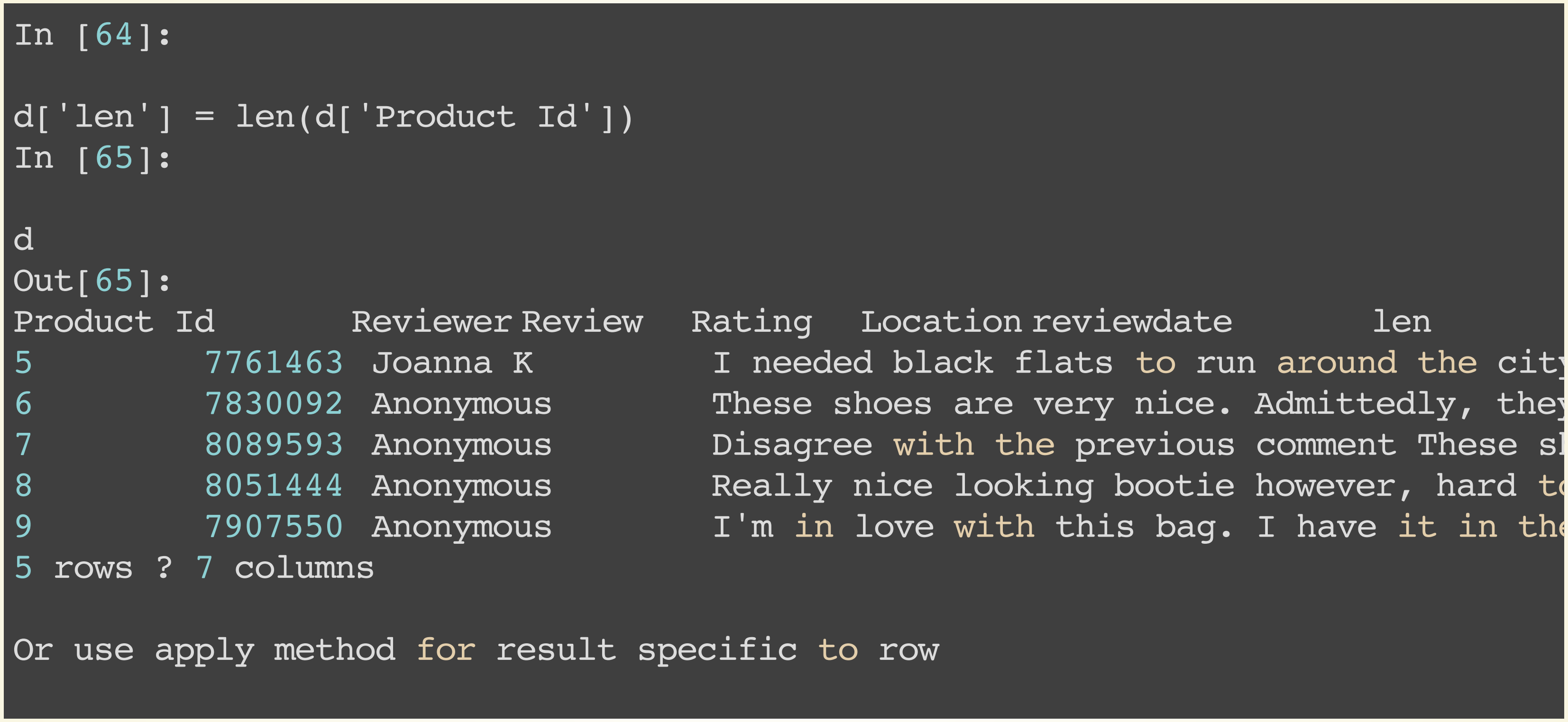
```
In [19]:

a.rename(columns={'col2':'col2_renamed'}, inplace=True)
In [20]:

a
Out[20]:
col1    col2_renamed
0        1        2
1        2        3
2        3        5
3        4        4
4        5        8
5        2        1
6        4        4
7 rows ? 2 columns

In [21]:
```

DROPPING COLUMNS



```
In [61]:

d
Out[61]:
Product Id       Review Date      Reviewer Review   Rating   Location reviewdate
5         7761463  11/10/2012       Joanna K          I needed black flats to ru
6         7830092  10/22/2012       Anonymous         These shoes are very nice.
7         8089593  5/19/2013        Anonymous         Disagree with the previous
8         8051444  1/17/2013        Anonymous         Really nice looking bootie
9         7907550  4/27/2012        Anonymous         I'm in love with this bag.
5 rows ? 7 columns
In [62]:

del d['Review Date']
In [63]:

d
Out[63]:
```

ADDING NEW COLUMNS

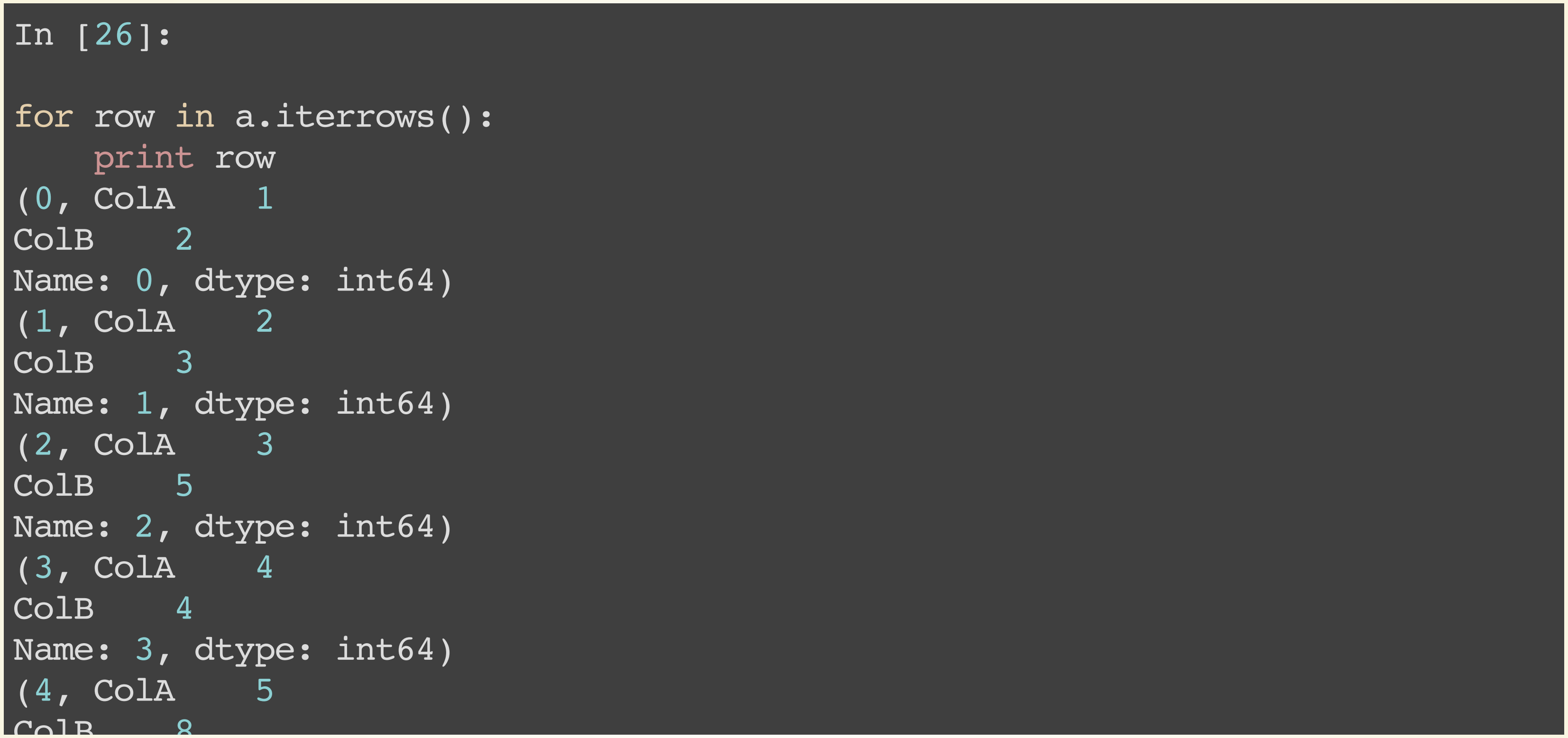


```
In [64]:

d['len'] = len(d['Product Id'])
In [65]:

d
Out[65]:
Product Id       Reviewer Review   Rating   Location reviewdate       len
5         7761463  Joanna K         I needed black flats to run around the city
6         7830092  Anonymous        These shoes are very nice. Admittedly, they
7         8089593  Anonymous        Disagree with the previous comment These sl
8         8051444  Anonymous        Really nice looking bootie however, hard to
9         7907550  Anonymous        I'm in love with this bag. I have it in the
5 rows ? 7 columns

Or use apply method for result specific to row
```

SORTING

```
In [24]:

a.sort(columns=['ColB'])
Out[24]:
ColA    ColB
5        2        1
0        1        2
1        2        3
3        4        4
6        4        4
2        3        5
4        5        8
7 rows ? 2 columns
```

LOOPING OVER DATA FRAME



```
In [26]:

for row in a.iterrows():
    print row
(0, ColA    1
ColB    2
Name: 0, dtype: int64)
(1, ColA    2
ColB    3
Name: 1, dtype: int64)
(2, ColA    3
ColB    5
Name: 2, dtype: int64)
(3, ColA    4
ColB    4
Name: 3, dtype: int64)
(4, ColA    5
ColB    8
```

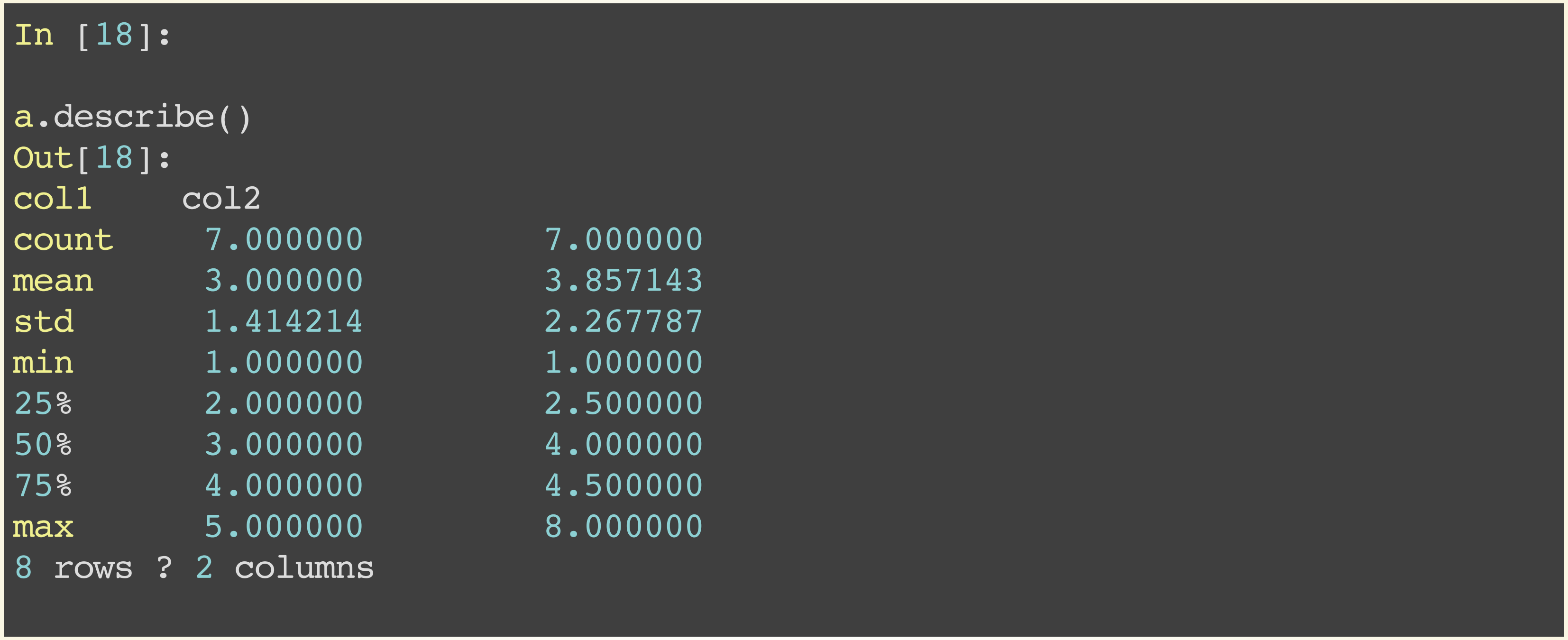
INDEXING

You can make a unique column as an index.

SUMMARIZING DATA/COMPUTING DESCRIPTIVE STATISTICS

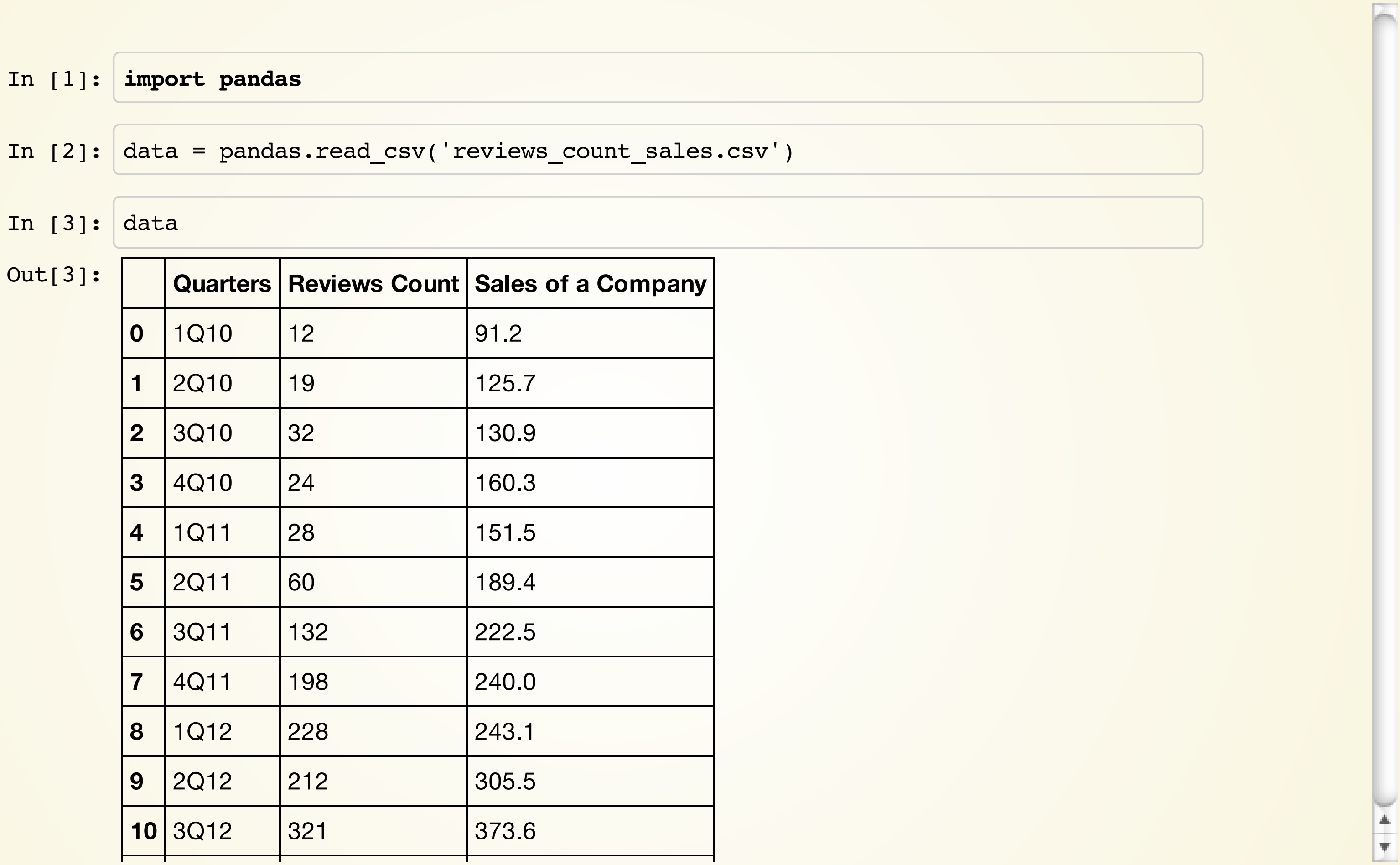
- Calculating mean, median, max, min, describe
- Computing correlation
- Computing covariance
- Finding uniques
- isin method

DESCRIBE METHOD



```
In [18]:

a.describe()
Out[18]:
col1     col2
count     7.000000         7.000000
mean      3.000000         3.857143
std       1.414214         2.267787
min       1.000000         1.000000
25%       2.000000         2.500000
50%       3.000000         4.000000
75%       4.000000         4.500000
max       5.000000         8.000000
8 rows ? 2 columns
```

COMPUTING CORRELATION



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

```
In [2]: data = pandas.read_csv('reviews_count_sales.csv')
```

```
In [3]: data
```

Out[3]:

	Quarters	Reviews Count	Sales of a Company
0	1Q10	12	91.2
1	2Q10	19	125.7
2	3Q10	32	130.9
3	4Q10	24	160.3
4	1Q11	28	151.5
5	2Q11	60	189.4
6	3Q11	132	222.5
7	4Q11	198	240.0
8	1Q12	228	243.1
9	2Q12	212	305.5
10	3Q12	321	373.6

COMPUTING COVARIANCE

```
In [27]:

a.cov()
Out[27]:
ColA    ColB
ColA     2.000000        2.666667
ColB     2.666667        5.142857
2 rows ? 2 columns
```

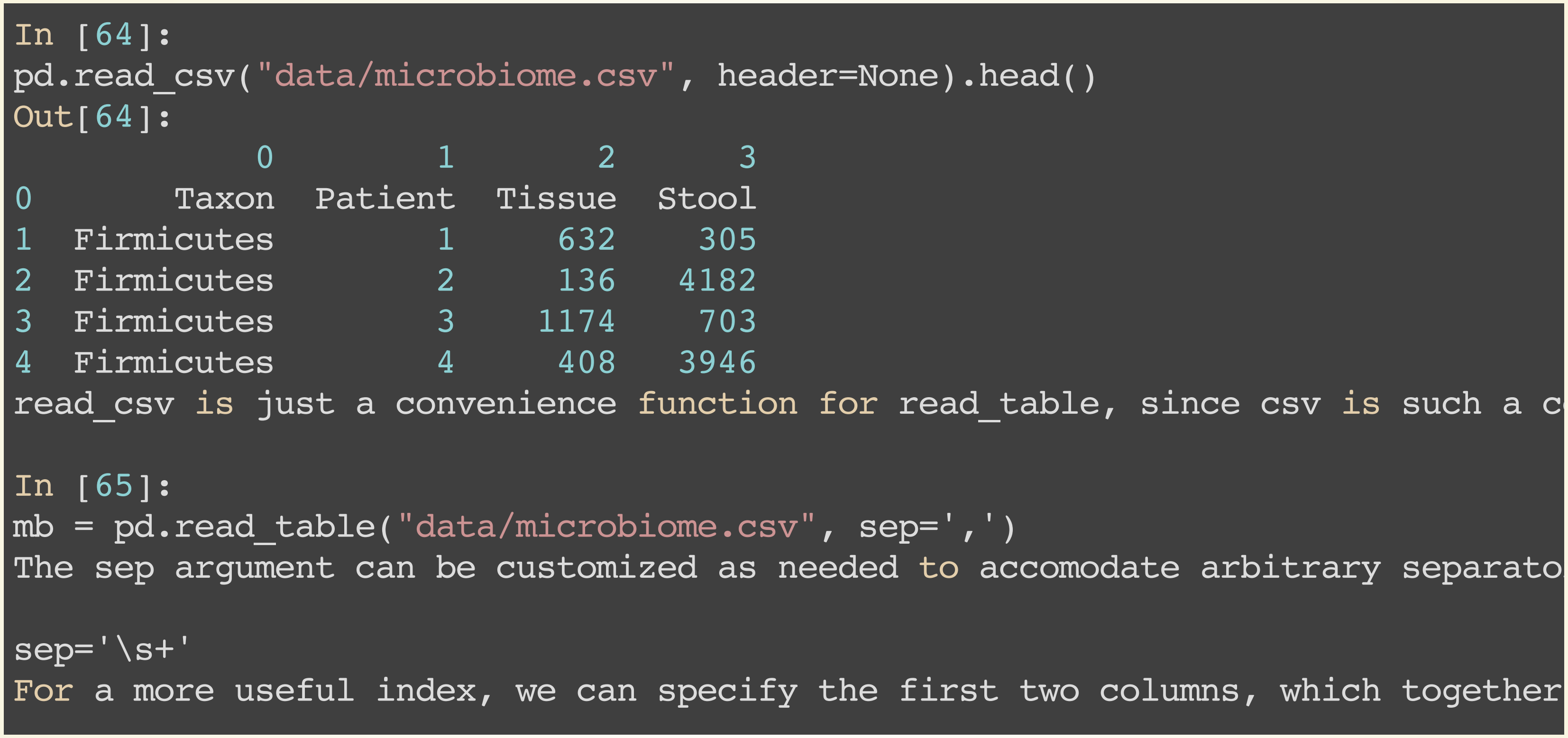
FILTERING DATA

```
In [57]:

d[d.reviewdate>'20130401']
Out[57]:
Product Id       Review Date      Reviewer Review   Rating   Location reviewdate
7         8089593  5/19/2013        Anonymous         Disagree with the previous
1 rows ? 7 columns
```

DATA LOADING

- Using read_csv, read_excel, read_table
- Reading in pieces
- Using json data
- Storing data

DATA LOADING CODE

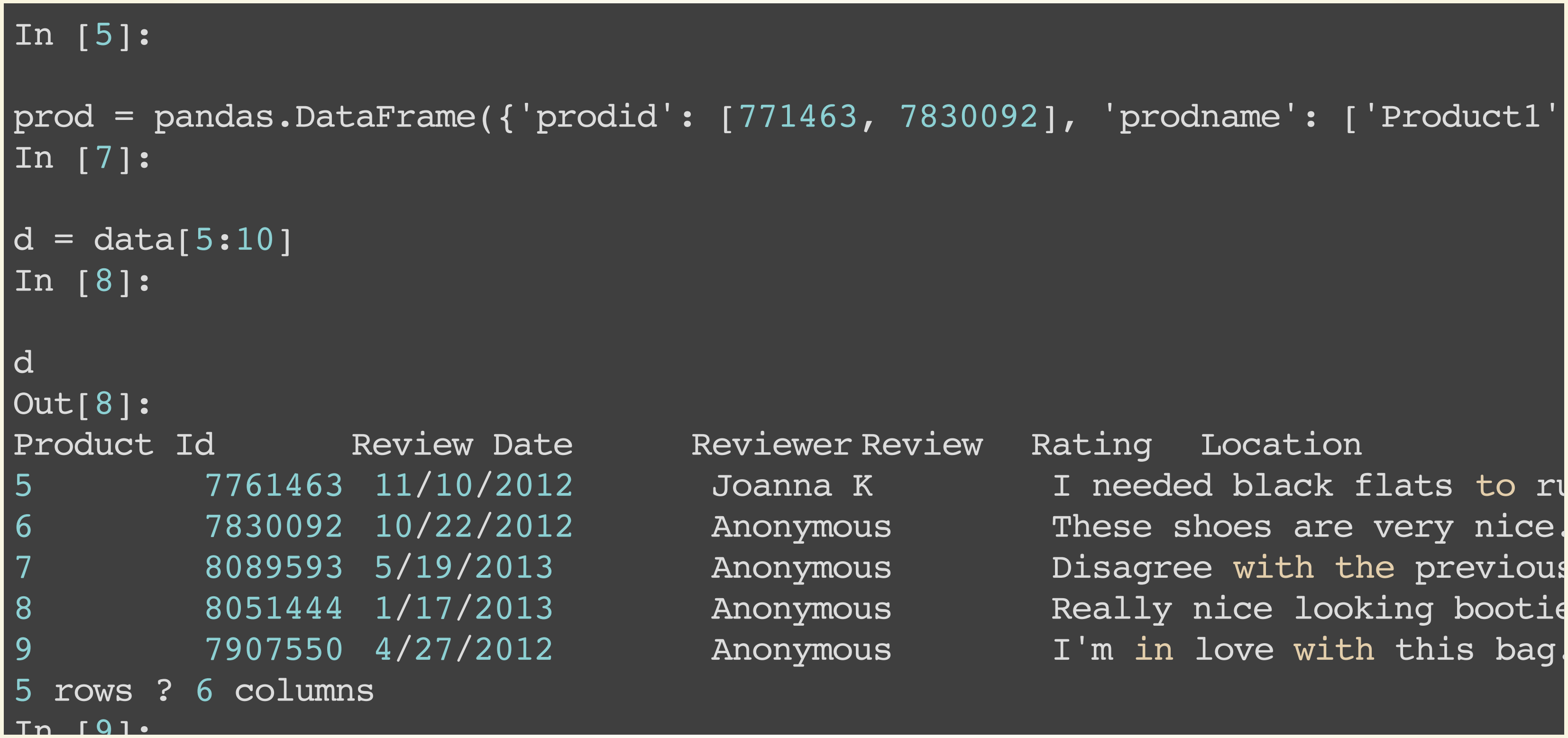


```
In [64]:
pd.read_csv("data/microbiome.csv", header=None).head()
Out[64]:
            0        1       2      3
0       Taxon  Patient  Tissue  Stool
1  Firmicutes        1     632    305
2  Firmicutes        2     136   4182
3  Firmicutes        3    1174    703
4  Firmicutes        4     408   3946
read_csv is just a convenience function for read_table, since csv is such a c

In [65]:
mb = pd.read_table("data/microbiome.csv", sep=',')
The sep argument can be customized as needed to accomodate arbitrary separato

sep='\s+'
For a more useful index, we can specify the first two columns, which together
```

DATA TRANSFORMATIONS

- Merge/Joins
- Concatenation
- Pivoting
- Removing duplicates (Already convered)

MERGE/JOINS

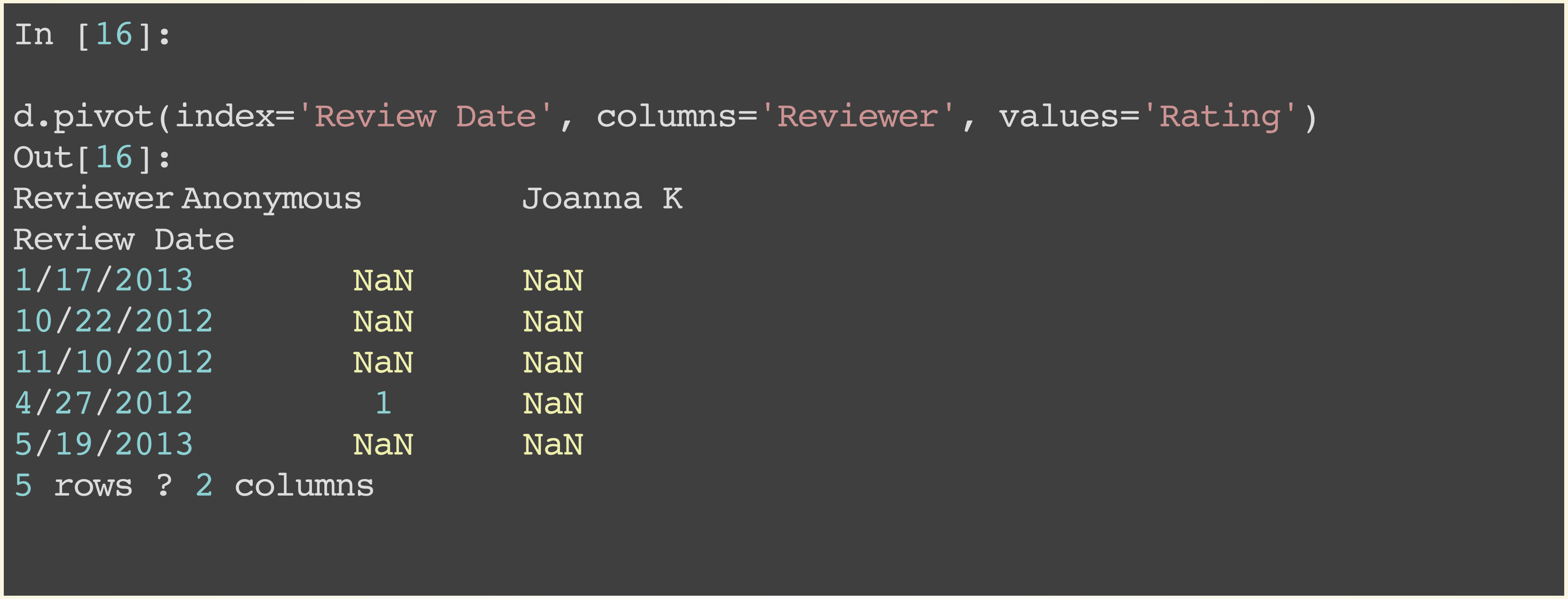


```
In [5]:

prod = pandas.DataFrame({'prodid': [771463, 7830092], 'prodname': ['Product1'
In [7]:

d = data[5:10]
In [8]:

d
Out[8]:
Product Id       Review Date      Reviewer Review   Rating   Location
5         7761463  11/10/2012       Joanna K          I needed black flats to ru
6         7830092  10/22/2012       Anonymous         These shoes are very nice.
7         8089593  5/19/2013        Anonymous         Disagree with the previous
8         8051444  1/17/2013        Anonymous         Really nice looking bootie
9         7907550  4/27/2012        Anonymous         I'm in love with this bag.
5 rows ? 6 columns
In [9]:
```

CONCATENATION

```
In [1]: df = DataFrame(np.random.randn(10, 4))

In [2]: df
Out[2]:
          0         1         2         3
0  0.469112 -0.282863 -1.509059 -1.135632
1  1.212112 -0.173215  0.119209 -1.044236
2 -0.861849 -2.104569 -0.494929  1.071804
3  0.721555 -0.706771 -1.039575  0.271860
4 -0.424972  0.567020  0.276232 -1.087401
5 -0.673690  0.113648 -1.478427  0.524988
6  0.404705  0.577046 -1.715002 -1.039268
7 -0.370647 -1.157892 -1.344312  0.844885
8  1.075770 -0.109050  1.643563 -1.469388
9  0.357021 -0.674600 -1.776904 -0.968914

[10 rows x 4 columns]
```

PIVOTING



```
In [16]:

d.pivot(index='Review Date', columns='Reviewer', values='Rating')
Out[16]:
Reviewer Anonymous        Joanna K
Review Date
1/17/2013        NaN      NaN
10/22/2012       NaN      NaN
11/10/2012       NaN      NaN
4/27/2012          1      NaN
5/19/2013        NaN      NaN
5 rows ? 2 columns
```

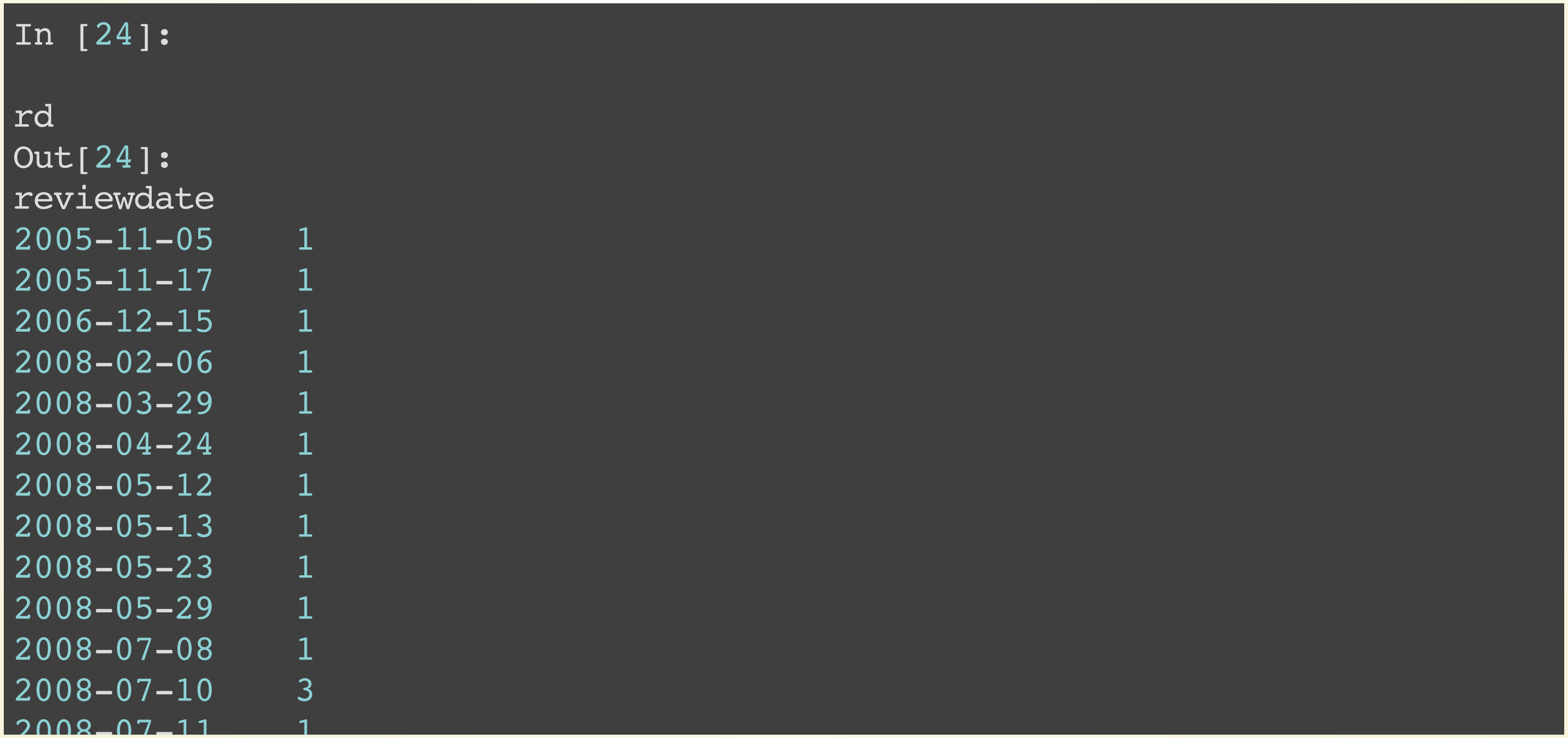
VECTORIZED COMPUTATION

```
In [59]:

import numpy
In [60]:

numpy.log(d['Product Id'])
Out[60]:
5    15.864681
6    15.873485
7    15.906089
8    15.901362
9    15.883329
Name: Product Id, dtype: float64

In [46]:

data['reviewdate'] = data['Review Date'].apply(lambda x: datetime.strptime(x,
```

DATA AGGREGATION

- Grouping data
- Using quantile
- Bucketing data

QUANTILE & BUCKETING DATA



```
In [24]:

rd
Out[24]:
reviewdate
2005-11-05    1
2005-11-17    1
2006-12-15    1
2008-02-06    1
2008-03-29    1
2008-04-24    1
2008-05-12    1
2008-05-13    1
2008-05-23    1
2008-05-29    1
2008-07-08    1
2008-07-10    3
2008-07-11    1
```

TIMESERIES DATA (COVERED IN EARLIER SLIDES)

- Converting dates using lambda functions
- Difference using frequencies
- Generate date ranges
- Timeseries plotting
- Moving average, sum, etc. methods

DATA COLLECTION

Method available for scraping data

- urllib2
- mechanize
- phantomjs

SEASONAL DATA/HW ANALYSIS

Demo on R

QUESTIONS?