

## Summarize Data

**df['w'].value\_counts()**

Count number of rows with each unique value of variable

**len(df)**

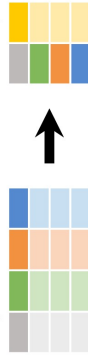
# of rows in DataFrame.

**df['w'].nunique()**

# of distinct values in a column.

**df.describe()**

Basic descriptive statistics for each column (or GroupBy)



pandas provides a large set of **summary functions** that operate on different kinds of pandas objects (DataFrame columns, Series, GroupBy, Expanding and Rolling (see below)) and produce single values for each of the groups. When applied to a DataFrame, the result is returned as a pandas Series for each column. Examples:

**sum()**

Sum values of each object.

**count()**

Count non-NA/null values of each object.

**median()**

Median value of each object.

**quantile([0.25,0.75])**

Quantiles of each object.

**apply(function)**

Apply function to each object.

**min()**

Minimum value in each object.

**max()**

Maximum value in each object.

**mean()**

Mean value of each object.

**var()**

Variance of each object.

**std()**

Standard deviation of each object.

## Group Data

**df.groupby(by="col")**

Return a GroupBy object, grouped by values in column named "col".

**df.groupby(level="ind")**

Return a GroupBy object, grouped by values in index level named "ind".

All of the summary functions listed above can be applied to a group.

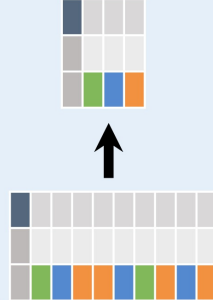
Additional GroupBy functions:

**size()**

Size of each group.

**agg(function)**

Aggregate group using function.



## Handling Missing Data

**df.dropna()**

Drop rows with any column having NA/null data.

**df.fillna(value)**

Replace all NA/null data with value.

## Make New Columns



**df.assign(Area=lambda df: df.Length\*df.Height)**

Compute and append one or more new columns.

**df['Volume'] = df.Length\*df.Height\*df.Depth**

Add single column.

**pd.qcut(df.col, n, labels=False)**

Bin column into n buckets.



pandas provides a large set of **vector functions** that operate on all columns of a DataFrame or a single selected column (a pandas Series). These functions produce vectors of values for each of the columns, or a single Series for the individual Series. Examples:

**max(axis=1)** **min(axis=1)**

Element-wise max.

**clip(lower=-10, upper=10)** **abs()**

Trim values at input thresholds Absolute value.

The examples below can also be applied to groups. In this case, the function is applied on a per-group basis, and the returned vectors are of the length of the original DataFrame.

**shift(1)**

Copy with values shifted by 1.

**rank(method='dense')**

Ranks with no gaps.

**rank(method='min')**

Ranks. Ties get min rank.

**rank(pct=True)**

Ranks rescaled to interval [0, 1].

**rank(method='first')**

Ranks. Ties go to first value.

**shift(-1)**

Copy with values lagged by 1.

**cumsum()**

Cumulative sum.

**cummax()**

Cumulative max.

**cummin()**

Cumulative min.

**cumprod()**

Cumulative product.

## Windows

**df.expanding()**

Return an Expanding object allowing summary functions to be applied cumulatively.

**df.rolling(n)**

Return a Rolling object allowing summary functions to be applied to windows of length n.

## Combine Data Sets

adf	x1	x2
A	1	
B	2	
C	3	



bdf	x1	x3
A	T	
B	F	
D	T	

Standard Joins

x1	x2	x3
A	1	T
B	2	F
C	3	NaN

**pd.merge(adf, bdf, how='left', on='x1')**  
Join matching rows from bdf to adf.

x1	x2	x3
A	1.0	T
B	2.0	F
D	NaN	T

**pd.merge(adf, bdf, how='right', on='x1')**  
Join matching rows from adf to bdf.

x1	x2	x3
A	1	T
B	2	F

**pd.merge(adf, bdf, how='inner', on='x1')**  
Join data. Retain only rows in both sets.

x1	x2	x3
A	1	T
B	2	F
C	3	NaN
D	NaN	T

**pd.merge(adf, bdf, how='outer', on='x1')**  
Join data. Retain all values, all rows.

Filtering Joins

x1	x2
A	1
B	2

**adf[adf.x1.isin(bdf.x1)]**  
All rows in adf that have a match in bdf.

x1	x2
C	3

**adf[~adf.x1.isin(bdf.x1)]**  
All rows in adf that do not have a match in bdf.

ydf	x1	x2
A	1	
B	2	
C	3	



zdf	x1	x2
B	2	
C	3	
D	4	

Set-like Operations

x1	x2
B	2
C	3

**pd.merge(ydf, zdf)**  
Rows that appear in both ydf and zdf (Intersection).

x1	x2
A	1
B	2
C	3
D	4

**pd.merge(ydf, zdf, how='outer')**  
Rows that appear in either or both ydf and zdf (Union).

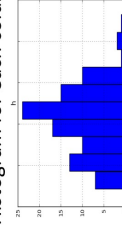
x1	x2
A	1

**pd.merge(ydf, zdf, how='outer', indicator=True)**  
**.query('\_merge == "left\_only"')**  
**.drop(['\_merge'],axis=1)**  
Rows that appear in ydf but not zdf (Setdiff).

## Plotting

**df.plot.hist()**

Histogram for each column



**df.plot.scatter(x='w',y='h')**

Scatter chart using pairs of points

