构造数据框

```
DataFrame([data, index, columns, dtype, copy]) #构造数据框
```

属性和数据

```
DataFrame.axes                                #index: 行标签；columns: 列标签
DataFrame.as_matrix([columns])                #转换为矩阵
DataFrame.dtypes                              #返回数据的类型
DataFrame.ftypes                              #返回每一列的 数据类型float64:dense
DataFrame.get_dtype_counts()                  #返回数据框数据类型的个数
DataFrame.get_ftype_counts()                  #返回数据框数据类型float64:dense的个数
DataFrame.select_dtypes([include, include])   #根据数据类型选取子数据框
DataFrame.values                              #Numpy的展示方式
DataFrame.axes                                #返回横纵坐标的标签名
DataFrame.ndim                                #返回数据框的纬度
DataFrame.size                                #返回数据框元素的个数
DataFrame.shape                               #返回数据框的形状
DataFrame.memory_usage()                      #每一列的存储
```

类型转换

```
DataFrame.astype(dtype[, copy, errors])       #转换数据类型
DataFrame.copy([deep])                        #deep深度复制数据
DataFrame.isnull()                            #以布尔的方式返回空值
DataFrame.notnull()                           #以布尔的方式返回非空值
```

索引和迭代

```
DataFrame.head([n])                           #返回前n行数据
DataFrame.at                                  #快速标签常量访问器
DataFrame.iat                                 #快速整型常量访问器
DataFrame.loc                                 #标签定位，使用名称
DataFrame.iloc                                #整型定位，使用数字
DataFrame.insert(loc, column, value)          #在特殊地点loc[数字]插入column[列名]某列
数据
DataFrame.iter()                              #Iterate over infor axis
DataFrame.iteritems()                         #返回列名和序列的迭代器
DataFrame.iterrows()                          #返回索引和序列的迭代器
DataFrame.itertuples([index, name])           #Iterate over DataFrame rows as
namedtuples, with index value as first element of the tuple.
DataFrame.lookup(row_labels, col_labels)      #Label-based “fancy indexing” function
for DataFrame.
DataFrame.pop(item)                           #返回删除的项目
DataFrame.tail([n])                           #返回最后n行
DataFrame.xs(key[, axis, level, drop_level])  #Returns a cross-section (row(s) or
column(s)) from the Series/DataFrame.
DataFrame.isin(values)                        #是否包含数据框中的元素
DataFrame.where(cond[, other, inplace, …])    #条件筛选
DataFrame.mask(cond[, other, inplace, …])     #Return an object of same shape as
self and whose corresponding entries are from self where cond is False and otherwise
are from other.
DataFrame.query(expr[, inplace])              #Query the columns of a frame with a
boolean expression.
```

二元运算

```
DataFrame.add(other[,axis,fill_value])          #加法，元素指向
DataFrame.sub(other[,axis,fill_value])          #减法，元素指向
DataFrame.mul(other[, axis,fill_value])         #乘法，元素指向
DataFrame.div(other[, axis,fill_value])         #小数除法，元素指向
DataFrame.truediv(other[, axis, level, …])      #真除法，元素指向
DataFrame.floordiv(other[, axis, level, …])     #向下取整除法，元素指向
DataFrame.mod(other[, axis,fill_value])         #模运算，元素指向
DataFrame.pow(other[, axis,fill_value])         #幂运算，元素指向
DataFrame.radd(other[, axis,fill_value])        #右侧加法，元素指向
DataFrame.rsub(other[, axis,fill_value])        #右侧减法，元素指向
DataFrame.rmul(other[, axis,fill_value])        #右侧乘法，元素指向
DataFrame.rdiv(other[, axis,fill_value])        #右侧小数除法，元素指向
DataFrame.rtruediv(other[, axis, …])            #右侧真除法，元素指向
DataFrame.rfloordiv(other[, axis, …])           #右侧向下取整除法，元素指向
DataFrame.rmod(other[, axis,fill_value])        #右侧模运算，元素指向
DataFrame.rpow(other[, axis,fill_value])        #右侧幂运算，元素指向
DataFrame.lt(other[, axis, level])              #类似Array.lt
DataFrame.gt(other[, axis, level])              #类似Array.gt
DataFrame.le(other[, axis, level])              #类似Array.le
DataFrame.ge(other[, axis, level])              #类似Array.ge
DataFrame.ne(other[, axis, level])              #类似Array.ne
DataFrame.eq(other[, axis, level])              #类似Array.eq
DataFrame.combine(other,func[,fill_value, …]) #Add two DataFrame objects and do not
propagate NaN values, so if for a
DataFrame.combine_first(other)                  #Combine two DataFrame objects and
default to non-null values in frame calling the method.
```

函数应用 & 分组 & 窗口

```
DataFrame.apply(func[, axis, broadcast, …])     #应用函数
DataFrame.applymap(func)                        #Apply a function to a DataFrame that
is intended to operate elementwise, i.e.
DataFrame.aggregate(func[, axis])               #Aggregate using callable, string,
dict, or list of string/callables
DataFrame.transform(func, *args, **kwargs)      #Call function producing a like-
indexed NDFrame
DataFrame.groupby([by, axis, level, …])         #分组
DataFrame.rolling(window[, min_periods, …])     #滚动窗口
DataFrame.expanding([min_periods, freq, …])     #拓展窗口
DataFrame.ewm([com, span, halflife,  …])        #指数权重窗口
```

描述统计学

```
DataFrame.abs()                              #返回绝对值
DataFrame.all([axis, bool_only, skipna])     #Return whether all elements are True
over requested axis
DataFrame.any([axis, bool_only, skipna])     #Return whether any element is True
over requested axis
DataFrame.clip([lower, upper, axis])         #Trim values at input threshold(s).
DataFrame.clip_lower(threshold[, axis])      #Return copy of the input with values
below given value(s) truncated.
DataFrame.clip_upper(threshold[, axis])      #Return copy of input with values
above given value(s) truncated.
DataFrame.corr([method, min_periods])        #返回本数据框成对列的相关性系数
DataFrame.corrwith(other[, axis, drop])      #返回不同数据框的相关性
DataFrame.count([axis, level, numeric_only]) #返回非空元素的个数
DataFrame.cov([min_periods])                 #计算协方差
DataFrame.cummax([axis, skipna])             #Return cumulative max over requested
axis.
DataFrame.cummin([axis, skipna])             #Return cumulative minimum over
requested axis.
DataFrame.cumprod([axis, skipna])            #返回累积
DataFrame.cumsum([axis, skipna])             #返回累和
DataFrame.describe([percentiles,include, …]) #整体描述数据框
DataFrame.diff([periods, axis])              #1st discrete difference of object
DataFrame.eval(expr[, inplace])              #Evaluate an expression in the context
of the calling DataFrame instance.
DataFrame.kurt([axis, skipna, level, …])     #返回无偏峰度Fisher’s  (kurtosis of
normal == 0.0).
DataFrame.mad([axis, skipna, level])         #返回偏差
DataFrame.max([axis, skipna, level, …])      #返回最大值
DataFrame.mean([axis, skipna, level, …])     #返回均值
DataFrame.median([axis, skipna, level, …])   #返回中位数
DataFrame.min([axis, skipna, level, …])      #返回最小值
DataFrame.mode([axis, numeric_only])         #返回众数
DataFrame.pct_change([periods, fill_method]) #返回百分比变化
DataFrame.prod([axis, skipna, level, …])     #返回连乘积
DataFrame.quantile([q, axis, numeric_only])  #返回分位数
DataFrame.rank([axis, method, numeric_only]) #返回数字的排序
DataFrame.round([decimals])                  #Round a DataFrame to a variable
number of decimal places.
DataFrame.sem([axis, skipna, level, ddof])   #返回无偏标准误
DataFrame.skew([axis, skipna, level, …])     #返回无偏偏度
DataFrame.sum([axis, skipna, level, …])      #求和
DataFrame.std([axis, skipna, level, ddof])   #返回标准误差
DataFrame.var([axis, skipna, level, ddof])   #返回无偏误差
```

从新索引 & 选取 & 标签操作

```
DataFrame.add_prefix(prefix)                  #添加前缀
DataFrame.add_suffix(suffix)                  #添加后缀
DataFrame.align(other[, join, axis, level])   #Align two object on their axes with
the
DataFrame.drop(labels[, axis, level, …])      #返回删除的列
DataFrame.drop_duplicates([subset, keep, …])  #Return DataFrame with duplicate rows
removed, optionally only
DataFrame.duplicated([subset, keep])          #Return boolean Series denoting
duplicate rows, optionally only
DataFrame.equals(other)                       #两个数据框是否相同
DataFrame.filter([items, like, regex, axis])  #过滤特定的子数据框
DataFrame.first(offset)                       #Convenience method for subsetting
initial periods of time series data based on a date offset.
DataFrame.head([n])                           #返回前n行
DataFrame.idxmax([axis, skipna])              #Return index of first occurrence of
maximum over requested axis.
DataFrame.idxmin([axis, skipna])              #Return index of first occurrence of
minimum over requested axis.
DataFrame.last(offset)                        #Convenience method for subsetting
final periods of time series data based on a date offset.
DataFrame.reindex([index, columns])           #Conform DataFrame to new index with
optional filling logic, placing NA/NaN in locations having no value in the previous
index.
DataFrame.reindex_axis(labels[, axis, …])     #Conform input object to new index
with optional filling logic, placing NA/NaN in locations having no value in the
previous index.
DataFrame.reindex_like(other[, method, …])    #Return an object with matching
indices to myself.
DataFrame.rename([index, columns])            #Alter axes input function or
functions.
DataFrame.rename_axis(mapper[, axis, copy])   #Alter index and / or columns using
input function or functions.
DataFrame.reset_index([level, drop, …])       #For DataFrame with multi-level index,
return new DataFrame with labeling information in the columns under the index names,
defaulting to ‘level_0’, ‘level_1’, etc.
DataFrame.sample([n, frac, replace, …])       #返回随机抽样
DataFrame.select(crit[, axis])                #Return data corresponding to axis
labels matching criteria
DataFrame.set_index(keys[, drop, append ])    #Set the DataFrame index (row labels)
using one or more existing columns.
DataFrame.tail([n])                           #返回最后几行
DataFrame.take(indices[, axis, convert])      #Analogous to ndarray.take
DataFrame.truncate([before, after, axis ])    #Truncates a sorted NDFrame before
and/or after some particular index value.
```

处理缺失值

```
DataFrame.dropna([axis, how, thresh, …])      #Return object with labels on given
axis omitted where alternately any
DataFrame.fillna([value, method, axis, …])    #填充空值
DataFrame.replace([to_replace, value, …])     #Replace values given in ‘to_replace’
with ‘value’.
```

从新定型 & 排序 & 转变形态

```
DataFrame.pivot([index, columns, values])     #Reshape data (produce a “pivot”
table) based on column values.
DataFrame.reorder_levels(order[, axis])       #Rearrange index levels using input
order.
DataFrame.sort_values(by[, axis, ascending])  #Sort by the values along either axis
DataFrame.sort_index([axis, level, …])        #Sort object by labels (along an axis)
DataFrame.nlargest(n, columns[, keep])        #Get the rows of a DataFrame sorted by
the n largest values of columns.
DataFrame.nsmallest(n, columns[, keep])       #Get the rows of a DataFrame sorted by
the n smallest values of columns.
DataFrame.swaplevel([i, j, axis])             #Swap levels i and j in a MultiIndex
on a particular axis
DataFrame.stack([level, dropna])              #Pivot a level of the (possibly
hierarchical) column labels, returning a DataFrame (or Series in the case of an
object with a single level of column labels) having a hierarchical index with a new
inner-most level of row labels.
DataFrame.unstack([level, fill_value])        #Pivot a level of the (necessarily
hierarchical) index labels, returning a DataFrame having a new level of column
labels whose inner-most level consists of the pivoted index labels.
DataFrame.melt([id_vars, value_vars, …])      #“Unpivots” a DataFrame from wide
format to long format, optionally
DataFrame.T                                   #Transpose index and columns
DataFrame.to_panel()                          #Transform long (stacked) format
(DataFrame) into wide (3D, Panel) format.
DataFrame.to_xarray()                         #Return an xarray object from the
pandas object.
DataFrame.transpose(*args, **kwargs)          #Transpose index and columns
```

Combining& joining&merging

```
DataFrame.append(other[, ignore_index, …])    #追加数据
DataFrame.assign(**kwargs)                    #Assign new columns to a DataFrame,
returning a new object (a copy) with all the original columns in addition to the new
ones.
DataFrame.join(other[, on, how, lsuffix, …])  #Join columns with other DataFrame
either on index or on a key column.
DataFrame.merge(right[, how, on, left_on, …]) #Merge DataFrame objects by performing
a database-style join operation by columns or indexes.
DataFrame.update(other[, join, overwrite, …]) #Modify DataFrame in place using non-
NA values from passed DataFrame.
```

时间序列

```
DataFrame.asfreq(freq[, method, how, …])      #将时间序列转换为特定的频次
DataFrame.asof(where[, subset])               #The last row without any NaN is taken
(or the last row without
DataFrame.shift([periods, freq, axis])        #Shift index by desired number of
periods with an optional time freq
DataFrame.first_valid_index()                 #Return label for first non-NA/null
value
DataFrame.last_valid_index()                  #Return label for last non-NA/null
value
DataFrame.resample(rule[, how, axis, …])      #Convenience method for frequency
conversion and resampling of time series.
DataFrame.to_period([freq, axis, copy])       #Convert DataFrame from DatetimeIndex
to PeriodIndex with desired
DataFrame.to_timestamp([freq, how, axis])     #Cast to DatetimeIndex of timestamps,
at beginning of period
DataFrame.tz_convert(tz[, axis, level, copy]) #Convert tz-aware axis to target time
zone.
DataFrame.tz_localize(tz[, axis, level, …])   #Localize tz-naive TimeSeries to
target time zone.
```

作图

```
DataFrame.plot([x, y, kind, ax, ….])          #DataFrame plotting accessor and
method
DataFrame.plot.area([x, y])                   #面积图Area plot
DataFrame.plot.bar([x, y])                    #垂直条形图Vertical bar plot
DataFrame.plot.barh([x, y])                   #水平条形图Horizontal bar plot
DataFrame.plot.box([by])                      #箱图Boxplot
DataFrame.plot.density(**kwds)                #核密度Kernel Density Estimate plot
DataFrame.plot.hexbin(x, y[, C, …])           #Hexbin plot
DataFrame.plot.hist([by, bins])               #直方图Histogram
DataFrame.plot.kde(**kwds)                    #核密度Kernel Density Estimate plot
DataFrame.plot.line([x, y])                   #线图Line plot
DataFrame.plot.pie([y])                       #饼图Pie chart
DataFrame.plot.scatter(x, y[, s, c])          #散点图Scatter plot
DataFrame.boxplot([column, by, ax, …])        #Make a box plot from DataFrame column
optionally grouped by some columns or
DataFrame.hist(data[, column, by, grid, …])   #Draw histogram of the DataFrame’s
series using matplotlib / pylab.
```

转换为其他格式

```
DataFrame.from_csv(path[, header, sep, …])    #Read CSV file (DEPRECATED, please use
pandas.read_csv() instead).
DataFrame.from_dict(data[, orient, dtype])    #Construct DataFrame from dict of
array-like or dicts
DataFrame.from_items(items[,columns,orient])  #Convert (key, value) pairs to
DataFrame.
DataFrame.from_records(data[, index, …])      #Convert structured or record ndarray
to DataFrame
DataFrame.info([verbose, buf, max_cols, …])   #Concise summary of a DataFrame.
DataFrame.to_pickle(path[, compression, …])   #Pickle (serialize) object to input
file path.
DataFrame.to_csv([path_or_buf, sep, na_rep])  #Write DataFrame to a comma-separated
values (csv) file
DataFrame.to_hdf(path_or_buf, key, **kwargs)  #Write the contained data to an HDF5
file using HDFStore.
DataFrame.to_sql(name, con[, flavor, …])      #Write records stored in a DataFrame
to a SQL database.
DataFrame.to_dict([orient, into])             #Convert DataFrame to dictionary.
DataFrame.to_excel(excel_writer[, …])         #Write DataFrame to an excel sheet
DataFrame.to_json([path_or_buf, orient, …])   #Convert the object to a JSON string.
DataFrame.to_html([buf, columns, col_space])  #Render a DataFrame as an HTML table.
DataFrame.to_feather(fname)                   #write out the binary feather-format
for DataFrames
DataFrame.to_latex([buf, columns, …])         #Render an object to a tabular
environment table.
DataFrame.to_stata(fname[, convert_dates, …]) #A class for writing Stata binary dta
files from array-like objects
DataFrame.to_msgpack([path_or_buf, encoding]) #msgpack (serialize) object to input
file path
DataFrame.to_sparse([fill_value, kind])       #Convert to SparseDataFrame
DataFrame.to_dense()                          #Return dense representation of
NDFrame (as opposed to sparse)
DataFrame.to_string([buf, columns, …])        #Render a DataFrame to a console-
friendly tabular output.
DataFrame.to_clipboard([excel, sep])          #Attempt to write text representation
of object to the system clipboard This can be pasted into Excel, for example.
```