A screenshot of a number

Description automatically generated

Pandas Cheat sheet for Data Science

Creating Dataframe [](https://pandas.pydata.org/docs/reference/api/pandas.DataFrame.html)

1. **df = pd.DataFrame({‘Name’:[‘A’,’B’],’ID’:[1,2]})**

creating dataframe from dictionary

1. **df = pd.DataFrame([[‘A’,1],[ ’B’,2]],columns = [‘Name’,’ID’])**

creating dataframe from list of lists.

1. **df = pd.DataFrame({‘Name’:[‘A’,’B’,’C’],’ID’:[1,2,3]},index = pd.MultiIndex\_from \_tuples([(‘CSE,’IOT’),(‘CSE,’ML’),(‘ADS’,’ML’)],names=[‘Dept’,’Spn’]))**

**Import pandas as pd**

[**https://pandas.pydata.org/docs/index.html**](https://pandas.pydata.org/docs/index.html)

A screenshot of a computer program

Description automatically generated

Basic Python and Pandas Operators

|  |  |
| --- | --- |
| < | Less than |
| > | Greater than |
| == | Equal to |
| != | Not equal to |
| <= | Less than or equal to |
| >= | Greater than or equal to |
| df.column.isin(values) | Value in a set of elements |
| pd.isnull(val) | If val is NaN |
| pd.notnull(val) | If val is not NaN |
| &, |, ~, ^ | Logical and, or, not, xor |

Reshape data

1. **pd.**[**concat**](https://pandas.pydata.org/docs/reference/api/pandas.concat.html)**([df1,df2])**

Concat 2 dataframes vertically

**pd.**[**concat**](https://pandas.pydata.org/docs/reference/api/pandas.concat.html)**([df1,df2],axis=1)**

Concat 2 dataframes horizontally

1. **pd.**[**melt**](https://pandas.pydata.org/docs/reference/api/pandas.melt.html)**(df1)**

make rows as columns

1. **df.**[**pivot**](https://pandas.pydata.org/docs/reference/api/pandas.pivot.html)**(columns = ‘variable’, values = ‘value’)**

Convert columns as rows by grouping unique values with one or more discrete categories.

Input/ Output Functions [](https://pandas.pydata.org/docs/reference/io.html)

1. Read and write to CSV:

**pd.**[**read\_csv**](https://pandas.pydata.org/docs/reference/api/pandas.read_csv.html)**(‘customer.csv’)**

**df.**[**to\_csv**](https://pandas.pydata.org/docs/reference/api/pandas.DataFrame.to_csv.html)**(‘file.csv’)**

1. Read and write to Excel:

**pd.**[**read\_excel**](https://pandas.pydata.org/docs/reference/api/pandas.read_excel.html)**(‘customer.xlsx’)**

**df.**[**to\_excel**](https://pandas.pydata.org/docs/reference/api/pandas.DataFrame.to_excel.html)**(‘file.xlsx’,sheet\_name = ‘sheet1’)**

1. Read and write to Database Table:

**Import pyodbc**

**conn = pyodbc.connect(connection\_string)**

**pd.**[**read\_sql**](https://pandas.pydata.org/docs/reference/api/pandas.read_sql.html)**(‘select \* from customer’,conn)**

**df.**[**to\_sql**](https://pandas.pydata.org/docs/reference/api/pandas.DataFrame.to_sql.html)**(‘customer’,schema = ‘dbo’,conn)**

**conn.close()**

Handling missing data

1. **df.**[**dropna**](https://pandas.pydata.org/docs/reference/api/pandas.DataFrame.dropna.html)**()** – drop rows that have NaN
2. **df.**[**fillna**](https://pandas.pydata.org/docs/reference/api/pandas.DataFrame.fillna.html)**(value)** – fill NaN values with value

value = {A:0,B:1} 🡪 fill column A that has NaN with 0 and column B with 1

Conditional Statements:

1. **df.**[**where**](https://pandas.pydata.org/docs/reference/api/pandas.DataFrame.where.html)**(cond,false expression)**
2. **df.**[**mask**](https://pandas.pydata.org/docs/reference/api/pandas.DataFrame.mask.html)**(cond, true expression)**

Combining datasets (joins) [](https://pandas.pydata.org/docs/reference/api/pandas.merge.html)

1. **pd.merge(df1,df2,how = ‘left/right/outer/inner’,on=’id’)**

Join 2 dataframe’s. define the type – left or right or inner or outer, define the key column

1. **df1[~df1.id.isin(df2.id)]**

Left Anti-semi join – Get the data from df1 where the id is not there in df2

1. **df1[df1.id.isin(df2.id)]**

Left Semi join – Get the data from df1 where the id is there in df2

1. **pd.merge(df1,df2)**

Intersection – get the common records between 2 dataframe

1. **pd.merge(df1,df2,how = ‘outer’)**

Union 2 dataframe 2 get all the records from both the dataframe

1. **pd.merge(df1,df2,how = ‘outer’,indicator = True).query(‘\_merge==”left\_only”’).drop (columns =[‘\_merge’])**

get the data from the first dataframe which is not available in second dataframe.

(same as left anti semi join)

A diagram of a venn diagram

Description automatically generated

Plotting [](https://pandas.pydata.org/docs/reference/plotting.html)

1. **df.plot.**[**hist**](https://pandas.pydata.org/docs/reference/api/pandas.DataFrame.plot.hist.html)**()** – plot a histogram
2. **df.plot.**[**scatter**](https://pandas.pydata.org/docs/reference/api/pandas.DataFrame.plot.scatter.html)**(x = 0,y=1)** – plot a scatter chart with x and y coordinates
3. **df.plot.**[**bar**](https://pandas.pydata.org/docs/reference/api/pandas.DataFrame.plot.bar.html)**()** – plot a vertical bar chart
4. **df.plot.**[**barh**](https://pandas.pydata.org/docs/reference/api/pandas.DataFrame.plot.barh.html)**()** – plot a horizontal oriented bar chart

A graph with numbers and a number

Description automatically generated A graph with blue dots

Description automatically generated

A graph of sales by year and country

Description automatically generated`A graph of sales and discounts

Description automatically generated A graph of sales and discounts

Description automatically generated A graph of sales and discounts

Description automatically generated

A diagram of a blue circle with black text

Description automatically generated

A blue and black circles

Description automatically generated

A diagram of a blue circle with black text

Description automatically generated

A blue and black circle with black text

Description automatically generated

A diagram of a venn diagram

Description automatically generated

Make New Columns

1. **df.**[**assign**](https://pandas.pydata.org/pandas-docs/stable/reference/api/pandas.DataFrame.assign.html)**(area = lamda x=x\*2)**

Compute and append column to the dataframe

1. **df[‘new’] = df.old\*2**
2. **df.max(axis = 1)** – row-wise max
3. **df.min(axis=1)** – row-wise min
4. **df.abs()** – absolute value

Summarize Data

1. **df.**[**describe**](https://pandas.pydata.org/pandas-docs/stable/reference/api/pandas.DataFrame.describe.html)**()** – summarize the average, std, median, mode, count, min, max values of numeric columns
2. **df.**[**info**](https://pandas.pydata.org/pandas-docs/stable/reference/api/pandas.DataFrame.info.html)**()** – provides info about the dataset such as non-null count, datatype for each column
3. **df.**[**nunique**](https://pandas.pydata.org/pandas-docs/stable/reference/api/pandas.DataFrame.nunique.html)**()** – outputs count of distinct elements in each column of the dataframe
4. **df.**[**shape**](https://pandas.pydata.org/pandas-docs/stable/reference/api/pandas.DataFrame.shape.html) – returns the number of rows and columns as a tuple
5. **df.**[**size**](https://pandas.pydata.org/pandas-docs/stable/reference/api/pandas.DataFrame.size.html) – returns the total number of elements in the dataframe
6. **df.**[**dtypes**](https://pandas.pydata.org/pandas-docs/stable/reference/api/pandas.DataFrame.dtypes.html) – returns the datatype of each column
7. **df.**[**columns**](https://pandas.pydata.org/pandas-docs/stable/reference/api/pandas.DataFrame.columns.html) – returns the column names in the dataframe
8. **df.**[**memory\_usage**](https://pandas.pydata.org/pandas-docs/stable/reference/api/pandas.DataFrame.memory_usage.html)**()** – summarizes the memory usage for each column

Subsets / Indexing

1. **Row-Wise:**

* **df.**[**head**](https://pandas.pydata.org/pandas-docs/stable/reference/api/pandas.DataFrame.head.html)**(n)** – return first n rows
* **df.**[**tail**](https://pandas.pydata.org/pandas-docs/stable/reference/api/pandas.DataFrame.tail.html)**(n)** – return last n rows
* **df.**[**drop\_duplicates**](https://pandas.pydata.org/pandas-docs/stable/reference/api/pandas.DataFrame.drop_duplicates.html)**(subset = [c1,c2])** – removes duplicates based on the column set provided
* **df[df.id==100]** – provides the rows where id=100
* **df.**[**sample**](https://pandas.pydata.org/pandas-docs/stable/reference/api/pandas.DataFrame.sample.html)**(10)** – randomly select 10 rows

1. **Column-Wise:**

* **df[‘c1’]** or **df.c1** – select only c1 column
* **df[[‘c1’,’c2’]]** – select both c1 and c2 column
* **df.**[**filter**](https://pandas.pydata.org/pandas-docs/stable/reference/api/pandas.DataFrame.filter.html)**(regex = ‘regex’)** – select column names that match the regular expression

1. **Both column and row-wise:**

* **df.**[**loc**](https://pandas.pydata.org/pandas-docs/stable/reference/api/pandas.DataFrame.loc.html)**[:,:]** – select all rows and columns

**df.**[**loc**](https://pandas.pydata.org/pandas-docs/stable/reference/api/pandas.DataFrame.loc.html)**[:,[‘c1’]]** – select all rows of c1 column

**df.**[**loc**](https://pandas.pydata.org/pandas-docs/stable/reference/api/pandas.DataFrame.loc.html)**[:,c1:c4]** - select all rows of between column c1 and c4

**df.**[**loc**](https://pandas.pydata.org/pandas-docs/stable/reference/api/pandas.DataFrame.loc.html)**[1:3,’c1’]** – select rows 1 to 3 and only c1 column

* **df.**[**iloc**](https://pandas.pydata.org/pandas-docs/stable/reference/api/pandas.DataFrame.iloc.html)**[0:5,0:2]** – same as loc but columns and rows are accessed using their index
* **df.**[**iat**](https://pandas.pydata.org/pandas-docs/stable/reference/api/pandas.DataFrame.iat.html)**[1,2]** – to access only one value in row 1 and column 2
* **df.**[**at**](https://pandas.pydata.org/pandas-docs/stable/reference/api/pandas.DataFrame.at.html)**[4,’A’]** – to access only one value of row 4 and column ‘A’

1. **Query method:**

**df.**[**query**](https://pandas.pydata.org/pandas-docs/stable/reference/api/pandas.DataFrame.query.html)**(‘c1>100’)** – result all columns and rows which match the logical condition

**df.**[**query**](https://pandas.pydata.org/pandas-docs/stable/reference/api/pandas.DataFrame.query.html)**(‘c2 == “One” ’)** – string comparison

Created By: Deivanai Thiyagarajan

**All the functions have links aligned**

Chaining of Methods

Pandas methods can be stacked one on top of another. Eg:

**df = df1.groupby(‘Dept’).max(‘MathScore’).loc[:,[MathScore]]**

Reset, rename, sort

1. **df.**[**reset\_index**](https://pandas.pydata.org/pandas-docs/stable/reference/api/pandas.DataFrame.reset_index.html)**() –** Reset Index of DataFrame by row\_numbers
2. **df.**[**rename**](https://pandas.pydata.org/pandas-docs/stable/reference/api/pandas.DataFrame.rename.html)**(columns = {‘A’:0,’B’:1}) –** rename the column A to 0 and B to 1

**df.**[**rename**](https://pandas.pydata.org/pandas-docs/stable/reference/api/pandas.DataFrame.rename.html)**(index = {0:’a’,1:’b’})** – rename the columns with index 0 as a and index 1 as b

1. **df.**[**reindex**](https://pandas.pydata.org/pandas-docs/stable/reference/api/pandas.DataFrame.reindex.html)**(new\_index) –** reorders the row based on the new\_index array if not available in existing df creates the row with NaN values
2. **df.**[**set\_index**](https://pandas.pydata.org/pandas-docs/stable/reference/api/pandas.DataFrame.set_index.html)**(‘A’) –** Sets column A as index for the dataframe
3. **df.**[**sort\_index**](https://pandas.pydata.org/pandas-docs/stable/reference/api/pandas.DataFrame.sort_index.html)**() –** Sorts the df based on the index
4. **df.**[**sort\_values**](https://pandas.pydata.org/pandas-docs/stable/reference/api/pandas.DataFrame.sort_values.html)**(‘A’,ascending = True) –** Sorts the rows of df based on column A and in ascending order

Grouping Data

|  |  |  |
| --- | --- | --- |
| **df.**[**groupby**](https://pandas.pydata.org/pandas-docs/stable/reference/api/pandas.DataFrame.groupby.html)**(by=[‘A'])** – Groups the dataframe based on the column ‘A’. | **Aggregation Functions** (Applied on top of Groupby):  [**Note**: Refer chaining methods section to use these functions with groupby]   1. [**sum**](https://pandas.pydata.org/pandas-docs/stable/reference/api/pandas.DataFrame.sum.html)**()** – sum of values of each group 2. [**count**](https://pandas.pydata.org/pandas-docs/stable/reference/api/pandas.DataFrame.count.html)**()** – count of elements in each group 3. [**min**](https://pandas.pydata.org/pandas-docs/stable/reference/api/pandas.DataFrame.min.html)**(),** [**max**](https://pandas.pydata.org/pandas-docs/stable/reference/api/pandas.DataFrame.max.html)**()** – min and max value of each group 4. [**mean**](https://pandas.pydata.org/pandas-docs/stable/reference/api/pandas.DataFrame.mean.html)**()** – average of value in each group 5. [**median**](https://pandas.pydata.org/pandas-docs/stable/reference/api/pandas.DataFrame.median.html)**()** – median value in each group 6. [**std**](https://pandas.pydata.org/pandas-docs/stable/reference/api/pandas.DataFrame.std.html)**()** – standard deviation of each group 7. [**var**](https://pandas.pydata.org/pandas-docs/stable/reference/api/pandas.DataFrame.var.html)**()** – variance of each group 8. [**agg**](https://pandas.pydata.org/pandas-docs/stable/reference/api/pandas.DataFrame.agg.html)**()** – Aggregate group | **Window functions** (Applied top of group by for each row)**:**  Count of the row is preserved as original dataframe   1. [**cumsum**](https://pandas.pydata.org/pandas-docs/stable/reference/api/pandas.DataFrame.cumsum.html)**(),[cumprod](https://pandas.pydata.org/pandas-docs/stable/reference/api/pandas.DataFrame.cumprod.html)()** – Cumulative sum & prod of each group 2. [**cummax**](https://pandas.pydata.org/pandas-docs/stable/reference/api/pandas.DataFrame.cummax.html)**(),**[**cummin**](https://pandas.pydata.org/pandas-docs/stable/reference/api/pandas.DataFrame.cummin.html)**()** – Cumulative max & min of each group 3. [**shift**](https://pandas.pydata.org/pandas-docs/stable/reference/api/pandas.DataFrame.shift.html)**(1)** – lead function in sql, [**shift**](https://pandas.pydata.org/pandas-docs/stable/reference/api/pandas.DataFrame.shift.html)**(-1)** – lag function 4. [**rank**](https://pandas.pydata.org/pandas-docs/stable/reference/api/pandas.DataFrame.rank.html)**(method = ‘dense/min/first/pct’)** – rank the df based in the column specified and method specified |

A screenshot of a computer

Description automatically generatedA screenshot of a computer

Description automatically generated