Pandas Cheetsheet

```
import pandas as pd
axis = 0 ->Row
axis = 1 ->Column
```

1) Creating Series and Data

```
pd.Series(list)
pd.DataFrame(dict)
```

2) Import and Export CSV files

```
#import
df  = pd.read_csv(path)

#export
ex = df.to_csv(path, index=False)
```

3) Attributes

```
#shape i.e a tuple representing (row, column)
df.shape

#returns number of elements in a df
df.size

#returns column labels of a dataframe
df.columns

#returns row labels of a df
df.index

#returns an array containing all values of df
df.values

#returns datatype of each column
df.dtypes

#Lists row and column indexes/labels
df.axes

#Returns whether dataframe is empty or not in boolean values
df.empty

#Returns number of dimensions
df.ndim
```

#Returns transpose
df.T

4) Functions

#Provides summary of df
info()

#Provides statistical summary of df
describe()

#Number of elements
count()

#Mean of numerical column
mean()

#Standard deviation of numerical column
std()

#Median
median()

#Minimum value
min()

#Maximum value
max()

Note:
min() and max() can be used for both numerical and categorical columns. When used for numerical columns, it returns the minimum value. But, in case of categorical columns, it returns the lexicographically smallest value in that column. In other words, it returns the value that appears first when the categorical values are sorted in alphabetical order.

5) Data viewing functions

#Returns first 5 rows of a df
df.head() #however we can specify number of rows that we need to display ex.df.head(3)

#Returns last 5 rows of a df
df.tail()

#Displaying random rows
df.sample(number) #bydefault axis=0

#Displaying random columns
df.sample(number, axis=1) #it displays specified number of columns

6) Indexing : loc and iloc

(i) loc : Label based indexing

#selecting single row
df.loc[row_index]

#selecting multiple rows
df.loc[[row_index1, row_index2, …]]

#selecting single column
df.loc[:, col_index]

#selecting multiple column (more than 2)
df.loc[:, [col_index1, col_index2]]

Modify a specific value
df.loc[row_index, col_index] = value

Select a range of rows (y is inclusive)
df.loc[x:y]

Select a range of rows with stepsize
df.loc[x:y:z]

Select a range of columns (y is inclusive)
df.loc[:, x:y]

Select a range of rows with stepsize
df.loc[:, x:y:z]

(i) iloc : Integer based indexing

All the methods of loc are applicable to iloc. Here column indexing starts from 0.
In case of slicing in iloc, the end index is inclusive.

7) Retreiving based on conditions

Ex: df[df[col] > val]

8) Creating copy of a df

df.copy()

9) Sorting

df.sort_values(by=col, ascending=False/True)

df.sort_index()

10) Insertion

i) Inserting a column
df[col] = list #col refers to column label

ii)Inserting a row
df.loc[len(df)] = list

11) Deletion

i) Deleting a column
df.drop(col, axis=1)

ii)Deleting a row
df.drop(row) #row refers to row index

iii) Deleting multiple columns
df.drop([col1, col2], axis=1)

iv) Deleting multiple rows
df.drop([row1, row2])

12) Data Type Conversion

#Converting df to dictionary
dic = df.to_dict()

#Converting df to string
st = df.to_string()

#Conversion of series to list
li = df[col].to_list()

#Converting datatype: int, float, object
df.astype(datatype)
df[col].astype(datatype)

13) Dealing with NaN values in csv

```
#Returns true if a value is null/NaN else False
df.isnull()

#Returns count of NaN values in each column
df.isnull().sum()

#Returns total number of NaN in df
df.isnull().sum().sum()

#Filling NaN values with something else
df.fillna(0) #Here NaN values filled with 0

#Dropping NaN values
df.dropna()
```

14) Checking for duplicate values

```
df.duplicated().sum()
```

15) Displaying all values

```
pd.set_option('display.max_columns',None)
pd.set_option('display.max_rows',None)
```

16) Getting unique values in a column

```
df[‘col’].unique()
```

17) Getting frequency of values in a column

```
df[‘col’].value_counts()
```

18) Concatenating two dataframe

```
pd.concat([df1, df2, …], axis=1)
```