**Data Creation and Importing**

* **pd.Series()**: Creates a one-dimensional labeled array capable of holding any data type.
* **pd.DataFrame()**: Creates a two-dimensional, size-mutable, and potentially heterogeneous tabular data structure.
* **pd.read\_csv()**: Reads a comma-separated values (CSV) file into a DataFrame.
* **pd.read\_excel()**: Reads an Excel file into a DataFrame.
* **pd.read\_sql()**: Reads a SQL query or database table into a DataFrame.
* **pd.read\_json()**: Reads a JSON string or file into a DataFrame.
* **pd.read\_html()**: Parses an HTML table and returns a DataFrame.
* **pd.read\_pickle()**: Loads a pickled object file into a DataFrame.
* **pd.DataFrame.from\_dict()**: Constructs a DataFrame from a dictionary of array-like or dicts.

**Data Inspection**

* **df.head()**: Returns the first n rows of the DataFrame.
* **df.tail()**: Returns the last n rows of the DataFrame.
* **df.info()**: Provides a concise summary of the DataFrame.
* **df.describe()**: Generates descriptive statistics.
* **df.dtypes**: Returns the data types of each column.
* **df.shape**: Returns a tuple representing the dimensionality of the DataFrame.
* **df.memory\_usage()**: Returns the memory usage of each column in bytes.

**Indexing and Selecting Data**

* **df.set\_index()**: Sets the DataFrame index using existing columns.
* **df.reset\_index()**: Resets the index of the DataFrame.
* **df.iloc[]**: Purely integer-location-based indexing for selection by position.
* **df.loc[]**: Accesses a group of rows and columns by labels or a boolean array.
* **df.at[]**: Accesses a single value for a row/column label pair.
* **df.iat[]**: Accesses a single value for a row/column pair by integer position.
* **df.query()**: Queries the DataFrame with a boolean expression.
* **df.filter()**: Subsets the DataFrame rows or columns according to labels.
* **df.sort\_values()**: Sorts the DataFrame by the values along either axis.
* **df.sort\_index()**: Sorts the DataFrame by its index.
* **df.nlargest()**: Returns the first n rows ordered by columns in descending order.
* **df.nsmallest()**: Returns the first n rows ordered by columns in ascending order.

**Data Cleaning**

* **df.isnull()**: Detects missing values.
* **df.notnull()**: Detects non-missing values.
* **df.dropna()**: Removes missing values.
* **df.fillna()**: Fills missing values using a specified method.
* **df.interpolate()**: Interpolates values according to different methods.
* **df.drop\_duplicates()**: Removes duplicate rows.
* **df.replace()**: Replaces values given in the DataFrame.
* **df.astype()**: Casts a DataFrame to a specified dtype.

**Data Manipulation**

* **df.assign()**: Assigns new columns to a DataFrame.
* **df.rename()**: Renames labels in the DataFrame.
* **df.insert()**: Inserts a column into the DataFrame.
* **df.pop()**: Removes and returns a column.
* **df.update()**: Modifies a DataFrame in place using non-NA values from another DataFrame.
* **df.apply()**: Applies a function along an axis of the DataFrame.
* **df.applymap()**: Applies a function element-wise across the whole DataFrame.
* **df.map()**: Maps values from a Series according to input correspondence.

**String Methods**

* **df.str.lower()**: Converts strings in Series/Index to lowercase.
* **df.str.upper()**: Converts strings in Series/Index to uppercase.
* **df.str.strip()**: Removes leading and trailing spaces.
* **df.str.contains()**: Tests if a pattern or regex is contained within a string.
* **df.str.replace()**: Replaces occurrences of a pattern/regex with another string.
* **df.str.extract()**: Extracts capture groups in the regex pat as columns.
* **df.str.split()**: Splits strings around a given delimiter.

**Grouping and Aggregation**

* **df.groupby()**: Groups DataFrame using a mapper or by a Series of columns.
* **df.agg()**: Aggregates using one or more operations over the specified axis.
* **df.aggregate()**: Aggregates using one or more operations over the specified axis.
* **df.transform()**: Applies a function to each group, returning a DataFrame with the same shape.
* **df.filter()**: Filters groups based on a group-wise computation.
* **df.sum()**: Computes the sum of each group.
* **df.mean()**: Computes the mean of each group.
* **df.median()**: Computes the median of each group.
* **df.var()**: Computes the variance of each group.
* **df.min()**: Computes the minimum of each group.
* **df.max()**: Computes the maximum of each group.
* **df.std()**: Computes the standard deviation of each group.
* **df.count()**: Computes the number of non-NA values in each group.
* **df.size()**: Computes the size of each group.
* **df.describe()**: Generates descriptive statistics for each group.
* **df.cumsum()**: Cumulatively sums each group.
* **df.cumprod()**: Cumulatively multiplies each group.
* **isin**: Filters DataFrame rows based on a boolean condition.

**Merging, Joining, and Concatenating**

* **pd.concat()**: Concatenates pandas objects along a particular axis.
* **pd.merge()**: Merges DataFrames by columns or indexes.
* **df.join()**: Joins columns of another DataFrame.

**Reshaping and Pivoting**

* **df.pivot()**: Reshapes data based on column values.
* **df.pivot\_table()**: Creates a spreadsheet-style pivot table.
* **df.stack()**: Stacks the prescribed level(s) from columns to index.
* **df.unstack()**: Unstacks the prescribed level(s) from index to columns.
* **df.transpose()**: Transposes the index and columns of the DataFrame.
* **df.T**: Transposes the index and columns of the DataFrame.
* **df.explode()**: Transforms each element of a list-like to a row.

**Time Series**

* **pd.to\_datetime()**: Converts an argument to datetime.
* **df.asfreq()**: Converts DataFrame to a specified frequency.
* **df.resample()**: Resamples time-series data.
* **df.shift()**: Shifts index by the desired number of periods.
* **df.tshift()**: Shifts the time index without realigning data.
* **df.rolling()**: Provides rolling window calculations.
* **df.expanding()**: Provides expanding window calculations.
* **df.ewm()**: Provides exponentially weighted window calculations.

**Input/Output**

* **df.to\_csv()**: Writes DataFrame to a CSV file.
* **df.to\_excel()**: Writes DataFrame to an Excel file.
* **df.to\_sql()**: Writes DataFrame to a SQL database.
* **df.to\_json()**: Writes DataFrame to a JSON file.
* **df.to\_html()**: Renders DataFrame as an HTML table.
* **df.to\_clipboard()**: Copies DataFrame to the system clipboard.
* **df.to\_pickle()**: Serializes DataFrame to a pickle object.

**Visualization**

* **df.plot()**: Plots the DataFrame.
* **df.hist()**: Plots a histogram.
* **df.boxplot()**: Plots a boxplot.
* **df.plot.scatter()**: Plots a scatter plot.
* **df.plot.bar()**: Plots a bar chart.
* **df.plot.pie()**: Plots a pie chart.
* **df.plot.line()**: Plots a line chart.
* **df.plot.area()**: Plots an area chart.