| **Category** | **Operation** | **Code Example** |
| --- | --- | --- |
| **Read Data** | **Read CSV File** | **df = pd.read\_csv('filename.csv')** |
|  | **Read Excel File** | **df = pd.read\_excel('filename.xlsx')** |
|  | **Read SQL Database** | **df = pd.read\_sql(query, connection)** |
| **Explore Data** | **Display Top Rows** | **df.head()** |
|  | **Display Bottom Rows** | **df.tail()** |
|  | **Display Data Types** | **df.dtypes** |
|  | **Summary Statistics** | **df.describe()** |
|  | **Info (Index, Columns, Data)** | **df.info()** |
| **Missing Values** | **Check for Missing Values** | **df.isnull().sum()** |
|  | **Fill Missing Values** | **df.fillna(value)** |
|  | **Drop Missing Values** | **df.dropna()** |
| **Column Operations** | **Rename Columns** | **df.rename(columns={'old\_name': 'new\_name'})** |
|  | **Drop Columns** | **df.drop(columns=['column\_name'])** |
| **Apply & Group** | **Apply Function** | **df['column'].apply(lambda x: function(x))** |
|  | **Group By and Aggregate** | **df.groupby('column').agg({'column': 'sum'})** |
|  | **Pivot Table** | **df.pivot\_table(index='column1', values='column2', aggfunc='mean')** |
| **Merging & Concatenation** | **Merge DataFrames** | **pd.merge(df1, df2, on='column')** |
|  | **Concatenate DataFrames** | **pd.concat([df1, df2])** |
| **Visualization** | **Histogram** | **df['column'].hist()** |
|  | **Boxplot** | **df.boxplot(column=['column1', 'column2'])** |
|  | **Scatter Plot** | **df.plot.scatter(x='col1', y='col2')** |
|  | **Line Plot** | **df.plot.line()** |
|  | **Bar Chart** | **df['column'].value\_counts().plot.bar()** |
|  | **Heatmap (Correlation)** | **sns.heatmap(df.corr(), annot=True)** |
|  | **Pairplot (Seaborn)** | **sns.pairplot(df)** |
|  | **Custom Plot** | **df.plot(); plt.show()** |
|  | **Customize Style** | **plt.style.use('ggplot')** |
| **Statistics** | **Correlation Matrix** | **df.corr()** |
|  | **Covariance Matrix** | **df.cov()** |
|  | **Value Counts** | **df['column'].value\_counts()** |
|  | **Unique Values** | **df['column'].unique()** |
|  | **Number of Unique Values** | **df['column'].nunique()** |
| **Selection** | **Select Column** | **df['column']** |
|  | **Select Multiple Columns** | **df[['col1', 'col2']]** |
|  | **Select Rows by Position** | **df.iloc[0:5]** |
|  | **Select Rows by Label** | **df.loc[0:5]** |
|  | **Conditional Selection** | **df[df['column'] > value]** |
| **Data Types & String Ops** | **Convert Data Type** | **df['column'].astype('type')** |
|  | **String to Lowercase** | **df['column'].str.lower()** |
|  |  |  |
|  | **Datetime Conversion** | **pd.to\_datetime(df['column'])** |
| **Indexing** | **Setting Index** | **df.set\_index('column')** |
|  | **Set Datetime Index** | **df.set\_index(pd.to\_datetime(df['date']))** |
|  | **Reset Index** | **df.reset\_index(drop=True)** |
| **Resampling & Rolling** | **Resampling** | **df.resample('M').mean()** |
|  | **Rolling Window** | **df.rolling(window=5).mean()** |
| **Export Data** | **To CSV** | **df.to\_csv('filename.csv')** |
|  | **To Excel** | **df.to\_excel('filename.xlsx')** |
|  | **To SQL** | **df.to\_sql('table\_name', connection)** |
| **Advanced Functions** | **Lambda** | **df.apply(lambda x: x + 1)** |
|  | **Melt (Longer Format)** | **df.melt(id\_vars=['col1'])** |
|  | **Stack/Unstack** | **df.stack(), df.unstack()** |
|  | **Cross Tab** | **pd.crosstab(df['col1'], df['col2'])** |
| **Joins** | **Outer Join** | **pd.merge(df1, df2, on='column', how='outer')** |
|  | **Inner Join** | **pd.merge(df1, df2, on='column', how='inner')** |
|  | **Left Join** | **pd.merge(df1, df2, on='column', how='left')** |
|  | **Right Join** | **pd.merge(df1, df2, on='column', how='right')** |
| **Duplicates** | **Find Duplicates** | **df.duplicated()** |
|  | **Remove Duplicates** | **df.drop\_duplicates()** |
| **Memory Optimization** | **Memory Usage** | **df.memory\_usage(deep=True)** |
|  | **Change to Categorical** | **df['column'].astype('category')** |
| **Categorical & Sparse** | **Working with Categorical** | **df['column'].astype('category')** |
|  | **Sparse Data** | **pd.arrays.SparseArray(df['column'])** |
| **Performance** | **Swifter Apply** | **df['column'].swifter.apply(lambda x: func(x))** |
|  | **Parallel with Dask** | **dd.from\_pandas(df, npartitions=10)** |
| **Multicolumn Grouping & Transform** | **Group by Multiple Columns** | **df.groupby(['col1', 'col2']).mean()** |
|  | **Aggregate Multiple Functions** | **df.groupby('col').agg(['mean', 'sum'])** |
|  | **Transform Function** | **df.groupby('col').transform(lambda x: x - x.mean())** |
| **String Operations** | **Contains** | **df[df['column'].str.contains('substring')]** |
|  | **Split** | **df['column'].str.split(' ', expand=True)** |
|  | **Regex Extract** | **df['column'].str.extract(r'(regex)')** |
| **Normalization** | **Min-Max Normalization** | **(df['column'] - df['column'].min()) / (df['column'].max() - df['column'].min())** |
|  | **Z-Score Standardization** | **(df['column'] - df['column'].mean()) / df['column'].std()** |
| **File Types** | **Read JSON** | **df = pd.read\_json('filename.json')** |
|  | **Write JSON** | **df.to\_json('filename.json')** |
|  | **Read XML** | **df = pd.read\_xml('filename.xml')** |
|  | **Read CSV with Delimiter** | **df = pd.read\_csv('filename.csv', delimiter=';')** |
|  | **Read HTML** | **dfs = pd.read\_html('http://example.com')** |
| **SQL & Web Integration** | **Connect SQL (SQLAlchemy)** | **create\_engine('sqlite:///db.sqlite')** |
|  | **Read SQL Table** | **df = pd.read\_sql('SELECT \* FROM table\_name', engine)** |
| **Validation** | **Assert for Nulls** | **assert df.notnull().all().all(), "Missing values"** |