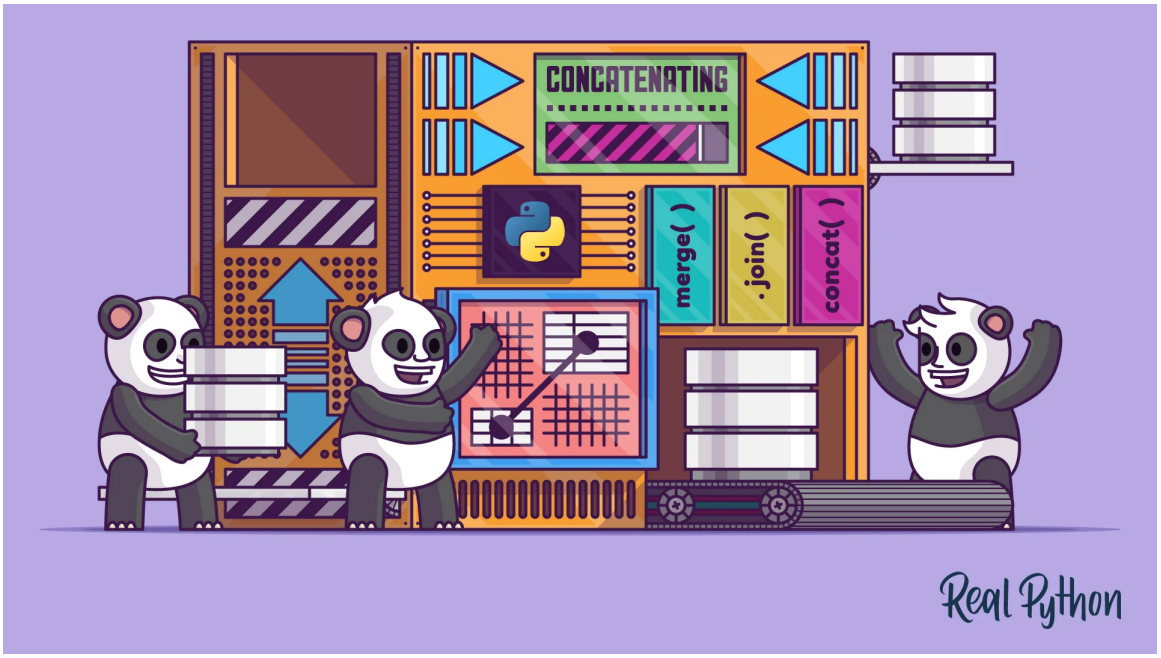
The Ultimate Pandas Guide: Simplifying Data Operations



Created by:

A Rahmawati

https://github.com/Data-Portofolio/The-Ultimate-Pandas-Guide-Simplifying-Data-Operations

Purpose	Method	Describe
Creation and Loading	**pd.DataFrame(data, index=index, columns=columns)**	Creates a new dataframe
	• **`pd.read_csv()`** • **`pd.read_excel()`** • **`pd.read_sql()`**	Loads data into a dataframe from various file formats and sources
Viewing and Inspecting	• **`df.head()`, `df.tail()`**	Display the first/last N rows of the dataframe.
	• **`df.info()`**	Provides basic information about the dataframe.
	• **`df.describe()`**	Generates summary statistics.
	• **`df.shape`**	Returns the number of rows and columns.
	• **`df.columns`**	Returns the column names.
Filtering and Sorting	• **`df['column_name']**	Selects a single column.
	• **`df.isin()** `filtered_df = df[df['Usia'].isin([30, 35]) & df['Kota'].isin(['Chicago', 'New York'])]`	DataFrame is contained in the specified list of values
	• **`df[['col1', 'col2']]`**	Selects multiple columns.
	• **`df.loc[df['Age'] < 30]`**	Allows for label-based indexing.
	• `df.iloc[1:4]`	Allows for integer-based indexing.
	• `df.query("Age < 30 and City == 'New York'")`	Allows for SQL-like queries.
	• **`df[df['column'] > value]`**	Filter rows based on a condition.

https://github.com/Data-Portofolio/The-Ultimate-Pandas-Guide-Simplifying-Data-Operations

	• **`df.sort_values()`**	Sorts the dataframe based on one or more columns.

Data Cleaning	• **`df.drop()`**
	`df.drop(labels=['col1',col2'], axis=1)`
	
	`df.drop(index=[0, 1, 2], axis=0)`
	
	• **`df.fillna()`**
	`df_filled = df.fillna(df.mean())`
	
	• **`df.replace()`**
	`df['B'] = df['B'].replace('Y', 'W', limit=1)`
	
	• **`df.drop_duplicates()`**
	`df_no_duplicates_b = df.drop_duplicates(subset=['B'], keep='last')`
		Removes specified rows or columns.
		
		Fills missing values.
		
		Replaces specific values
		
		Removes duplicate rows.

Aggregation and Grouping	`df.groupby('Product').agg({'Price': 'mean', 'Quantity': 'sum'})`
	
	`grouped = df.groupby(['Toko', 'Produk'])['Harga'].mean()`
	
	`grouped_data = df.groupby(['Toko', 'Produk']).agg( Rata_rata_Harga=('Harga', 'mean'), Total_Kuantitas=('Kuantitas', 'sum'))`
		Groups the dataframe by one or more columns

Joining and Combining	• **`pd.concat()`**
	`result = pd.concat([df1, df2], ignore_index=True)`
	
	• **`df.merge()`**
	`result = orders.merge(customers, on='CustomerID')`
	
	• **`df.join()`**
	`result = df1.join(df2)`
		Combines dataframes vertically
		
		Performs SQL-style joins on dataframes
		
		Joins dataframes using index/columns.

Pivoting and Reshaping	• **`df.pivot()`** `pivot_df = melted_df.pivot(index='Tahun',`	Reshape data

https://github.com/Data-Portofolio/The-Ultimate-Pandas-Guide-Simplifying-Data-Operations

	columns='Produk', values='Penjualan') • **`df.melt()`** melted_df = df.melt(id_vars=['Tahun'], var_name='Produk', value_name='Penjualan')	
Converting Column Data Types	• `df['col'] = df['col'].**astype('int64')**` • `df['date'] = pd.**to_datetime(df['date'])**` • `df['cat'] = df['cat'].**astype('category')**` • `df['col'] = pd.**to_numeric(df['col'], errors='coerce')**` • `df['col'] = df['col'].**astype(bool)**` • `df['col'] = df['col'].**astype(str)**`	