```
import pandas as pd
```

## Membuat DataFrame Baru

```
# Membuat DataFrame dari dictionary
data = {'Nama': ['John', 'Anna', 'Peter', 'Linda'],
       'Usia': [28, 35, 40, 32],
       'Kota': ['New York', 'Paris', 'Berlin', 'London']}

# Memuat data ke dalam Dataframe
df = pd.DataFrame(data)

# Menampilkan DataFrame
df.head(10)
```

| | Nama | Usia | Kota |
|---|---|---|---|
| **0** | John | 28 | New York |
| **1** | Anna | 35 | Paris |
| **2** | Peter | 40 | Berlin |
| **3** | Linda | 32 | London |

## Memuat Data ke dalam Dataframe

```
# Memuat file ke dalam dataframe

df = pd.read_csv('List of Orders.csv')

df
```

| | Order ID | Order Date | CustomerName | State | City |
|---|---|---|---|---|---|
| **0** | B-25601 | 01-04-2018 | Bharat | Gujarat | Ahmedabad |
| **1** | B-25602 | 01-04-2018 | Pearl | Maharashtra | Pune |
| **2** | B-25603 | 03-04-2018 | Jahan | Madhya Pradesh | Bhopal |
| **3** | B-25604 | 03-04-2018 | Divsha | Rajasthan | Jaipur |
| **4** | B-25605 | 05-04-2018 | Kasheen | West Bengal | Kolkata |
| **...** | ... | ... | ... | ... | ... |
| **555** | NaN | NaN | NaN | NaN | NaN |
| **556** | NaN | NaN | NaN | NaN | NaN |
| **557** | NaN | NaN | NaN | NaN | NaN |
| **558** | NaN | NaN | NaN | NaN | NaN |
| **559** | NaN | NaN | NaN | NaN | NaN |

560 rows × 5 columns

## Attribut

Shape : Dimensi tabel (baris x kolom)

```
# Atribut .shape (jumlah baris, jumlah kolom)
df.shape
```

columns : jumlah kolom

```
# Atribut .columns (jumlah kolom)
df.columns
```

index : rentang index

```
# Atribut .index (rentang index yang digunakan)
df.index
```

Info : informasi rinci dari datafram

```
# Atribut .info (informasi rinci tentang DataFrame:
#                tipe data setiap kolom,
#                jumlah nilai non-null,
#                dan penggunaan memori)
df.info()
```

## Contoh Lengkap Implementasi

```
import pandas as pd

# Membuat DataFrame contoh
data = {'Nama': ['John', 'Anna', 'Peter', 'Linda'],
       'Usia': [28, 35, 40, 32],
       'Kota': ['New York', 'Paris', 'Berlin', 'London']}
df = pd.DataFrame(data)

# Mengidentifikasi atribut DataFrame
print("Dimensi DataFrame:")
print(df.shape)


print("\nLabel Kolom:")
print(df.columns)


print("\nIndeks Baris:")
print(df.index)


print("\nTipe data setiap kolom:")
print(df.dtypes)


print("\nInformasi DataFrame:")
print(df.info())
```

```
Dimensi DataFrame:
(4, 3)

Label Kolom:
Index(['Nama', 'Usia', 'Kota'], dtype='object')

Indeks Baris:
RangeIndex(start=0, stop=4, step=1)

Tipe data setiap kolom:
Nama    object
Usia     int64
Kota    object
dtype: object

Informasi DataFrame:
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 4 entries, 0 to 3
Data columns (total 3 columns):
 #   Column  Non-Null Count  Dtype
---  ------  --------------  -----
 0   Nama    4 non-null      object
 1   Usia    4 non-null      int64
 2   Kota    4 non-null      object
dtypes: int64(1), object(2)
memory usage: 224.0+ bytes
None
```

## Mengakses Dataframe

### Mengakses Kolom

```
# Mengakses satu kolom
# df['Nama_Kolom']

# Mengakses beberapa kolom
# df[['Nama_Kolom_1', 'Nama_Kolom_2']]


# Mengakses satu kolom berdasarkan indeks
# df.iloc[:, indeks_kolom]

# Mengakses beberapa kolom berdasarkan indeks
# df.iloc[:, [indeks_kolom_1, indeks_kolom_2]]



# Mengakses baris berdasarkan indeks
# df.iloc[indeks_baris]


# Mengakses baris berdasarkan rentang indeks
# df.iloc[indeks_awal:indeks_akhir]


# Mengakses baris terakhir
# df.iloc[-1]


# Mengakses baris berdasarkan kriteria
# df[df['Kolom_Kriteria'] == nilai_kriteria]


# Menggunakan operasi logika untuk masking
# df[(df['Kolom_1'] > nilai_1) & (df['Kolom_2'] < nilai_2)]


# Menggunakan fungsi loc untuk mengakses baris berdasarkan label indeks
# df.loc[label_indeks]

# Menggunakan fungsi iloc untuk mengakses baris berdasarkan label indeks
# df.iloc[label_index]
```

## Contoh Lengkap Implementasi

```
# Membuat DataFrame contoh
import pandas as pd

data = {'Nama': ['John', 'Anna', 'Peter', 'Linda'],
      'Usia': [28, 35, 40, 32],
      'Kota': ['New York', 'Paris', 'Berlin', 'London']}
df = pd.DataFrame(data)


# Mengakses kolom 'Nama'
print("# Mengakses kolom 'Nama'")
print(df['Nama'])


# Mengakses baris dengan indeks 0
print("# Mengakses baris dengan indeks 0")
print(df.iloc[0])


# Mengakses baris dengan kriteria 'Usia' lebih dari 38
print("# Mengakses baris dengan kriteria 'Usia' lebih dari 38")
print(df[df['Usia'] > 38])


# Mengakses beberapa kolom dengan indeks
print("# Mengakses beberapa kolom dengan indeks")
print(df.iloc[:, [0, 2]])
```

```
# Mengakses kolom 'Nama'
0     John
1     Anna
2    Peter
3    Linda
Name: Nama, dtype: object
# Mengakses baris dengan indeks 0
Nama        John
Usia          28
Kota    New York
Name: 0, dtype: object
# Mengakses baris dengan kriteria 'Usia' lebih dari 38
    Nama  Usia    Kota
2  Peter    40  Berlin
# Mengakses beberapa kolom dengan indeks
    Nama      Kota
0   John  New York
1   Anna     Paris
2  Peter    Berlin
3  Linda    London
```

```
sales_data = pd.read_csv('Order Details.csv')
sales_data
```

| | Order ID | Amount | Profit | Quantity | Category | Sub-Category |
|---|---|---|---|---|---|---|
| **0** | B-25601 | 1275.0 | -1148.0 | 7 | Furniture | Bookcases |
| **1** | B-25601 | 66.0 | -12.0 | 5 | Clothing | Stole |
| **2** | B-25601 | 8.0 | -2.0 | 3 | Clothing | Hankerchief |
| **3** | B-25601 | 80.0 | -56.0 | 4 | Electronics | Electronic Games |
| **4** | B-25602 | 168.0 | -111.0 | 2 | Electronics | Phones |
| **...** | ... | ... | ... | ... | ... | ... |
| **1495** | B-26099 | 835.0 | 267.0 | 5 | Electronics | Phones |
| **1496** | B-26099 | 2366.0 | 552.0 | 5 | Clothing | Trousers |
| **1497** | B-26100 | 828.0 | 230.0 | 2 | Furniture | Chairs |
| **1498** | B-26100 | 34.0 | 10.0 | 2 | Clothing | T-shirt |
| **1499** | B-26100 | 72.0 | 16.0 | 2 | Clothing | Shirt |

1500 rows × 6 columns

```
sales_data.iloc[:, :5]
```

| | Order ID | Amount | Profit | Quantity | Category |
|---|---|---|---|---|---|
| **0** | B-25601 | 1275.0 | -1148.0 | 7 | Furniture |
| **1** | B-25601 | 66.0 | -12.0 | 5 | Clothing |
| **2** | B-25601 | 8.0 | -2.0 | 3 | Clothing |
| **3** | B-25601 | 80.0 | -56.0 | 4 | Electronics |
| **4** | B-25602 | 168.0 | -111.0 | 2 | Electronics |
| **...** | ... | ... | ... | ... | ... |
| **1495** | B-26099 | 835.0 | 267.0 | 5 | Electronics |
| **1496** | B-26099 | 2366.0 | 552.0 | 5 | Clothing |
| **1497** | B-26100 | 828.0 | 230.0 | 2 | Furniture |
| **1498** | B-26100 | 34.0 | 10.0 | 2 | Clothing |
| **1499** | B-26100 | 72.0 | 16.0 | 2 | Clothing |

1500 rows × 5 columns

## Penggunaan Fungsi SUM

```
# Kategori produk apa yang memiliki jumlah penjualan paling banyak?
# Perintah sum digunakan untuk menghitung jumlah dari keseluruhan nilai data

# sales_data.groupby('Category').agg({'Quantity': 'sum'})
```

```
# Total profit per kategori produk.
# Perintah sum digunakan untuk menghitung jumlah dari keseluruhan nilai data.

sales_data.groupby('Category').agg({'Profit': 'sum'})
```

## Penggunaan Fungsi Nunique

```
# Jumlah transaksi per kategori
# menggunakan nunique untuk mengetahui jumlah data unik Order ID

sales_data.groupby('Category').agg({'Order ID': 'nunique'})
```

## Menggunakan fungsi mean

```
# Menampilkan semua nilai rata-rata data yang dikelompokkan berdasarkan kategori
sales_data[['Category', 'Amount',   'Profit',   'Quantity']].groupby('Category').mean().head()
```

## Penggunaan Fungsi Group By

```
# Total produk yang paling banyak dibeli grouped by kategori
sales_data.groupby(['Category', 'Sub-Category']).sum()[['Quantity']]
```

## Mengubah Nilai pada Kolom

```
# Mengubah nilai dalam satu kolom
# df['Kolom'] = nilai_baru

# Mengubah nilai dalam beberapa kolom
# df[['Kolom_1', 'Kolom_2']] = nilai_baru
```

## Menambahkan kolom baru dengan nilai statis

```
# Menambahkan kolom baru dengan nilai statis
# df['Kolom_Baru'] = nilai_baru

# Menambahkan kolom baru dengan hasil operasi kolom lain
# df['Kolom_Baru'] = df['Kolom_1'] + df['Kolom_2']
```

## Menghapus Kolom

```
# Menghapus satu kolom
# df.drop('Kolom', axis=1, inplace=True)

# Menghapus beberapa kolom
# df.drop(['Kolom_1', 'Kolom_2'], axis=1, inplace=True)
```

## Menambah baris dengan nilai tertentu

```
# Menambahkan baris baru dengan nilai
# df = df.append({'Kolom_1': nilai_1, 'Kolom_2': nilai_2}, ignore_index=True)
```

## Menghapus baris berdasarkan index

```
# Menghapus baris berdasarkan indeks
# df.drop(indeks_baris, inplace=True)

# Menghapus baris berdasarkan kriteria
# df = df[df['Kolom'] != nilai_kriteria]
```

## Mereset index

```
# Mereset indeks DataFrame
# df.reset_index(drop=True, inplace=True)
```

## Contoh

```
# Membuat DataFrame contoh
data = {'Nama': ['John', 'Anna', 'Peter', 'Linda'],
      'Usia': [28, 35, 40, 32],
      'Kota': ['New York', 'Paris', 'Berlin', 'London']}
df = pd.DataFrame(data)

print("Dataframe sebelum modifikasi")
print(df)
print("\n")


# Mengubah nilai dalam kolom 'Usia'
df['Usia'] = [30, 36, 41, 33]


print("Dataframe setelah perubahan usia")
print(df)
print("\n")


# Menambahkan kolom baru 'Jenis_Kelamin'
df['Jenis_Kelamin'] = ['Laki-laki', 'Perempuan', 'Laki-laki', 'Perempuan']


print("Dataframe setelah penambahan kolom baru")
print(df)
print("\n")


# Menghapus kolom 'Kota'
df.drop('Kota', axis=1, inplace=True)


print("Dataframe setelah menghapus kolom kota")
print(df)
print("\n")
```

```
# Membuat DataFrame contoh
data = {'Nama': ['John', 'Anna', 'Peter', 'Linda'],
       'Usia': [28, 35, 40, 32],
       'Nilai': [85, 92, 78, 90]}
df = pd.DataFrame(data)

print("\nDataFrame sebelum operasi lambda:")
print(df)

# Menggunakan lambda untuk menghitung kuadrat dari setiap elemen dalam kolom 'Usia'
df['Usia_Kuadrat'] = df['Usia'].apply(lambda x: x**2)


# Menggunakan lambda untuk mengonversi setiap elemen dalam kolom 'Nilai' menjadi huruf A jika lebih dari 80, dan B jika tidak
df['Nilai_Grade'] = df['Nilai'].map(lambda x: 'A' if x > 80 else 'B')


# Menggunakan lambda untuk memfilter baris berdasarkan kondisi tertentu dalam kolom 'Usia'
df_filtered = df[df['Usia'].apply(lambda x: x >= 30)]
```