Manipulasi Data: Mengurutkan DataFrame (`.sort_values()`)

Salah satu tugas paling dasar dalam analisis data adalah mengurutkan data untuk melihat nilai tertinggi, terendah, atau untuk mengatur data agar lebih mudah dibaca. Di Pandas, kita menggunakan metode `.sort_values()`.

Mengurutkan Berdasarkan Satu Kolom

Cara paling sederhana adalah mengurutkan berdasarkan nilai di satu kolom. Secara *default*, Pandas akan mengurutkan dari yang terkecil ke terbesar (*ascending*).

Contoh: Mengurutkan data anjing berdasarkan berat badan.

```
# Mengurutkan dari yang paling ringan ke paling berat

dogs.sort_values("weight_kg")
```

Mengurutkan Secara Menurun (*Descending*)

Jika Anda ingin melihat data dari yang terbesar ke terkecil, Anda bisa menambahkan argumen `ascending=False`.

Contoh: Mengurutkan dari yang paling berat ke paling ringan.

```
dogs.sort_values("weight_kg", ascending=False)
```

Mengurutkan Berdasarkan Beberapa Kolom

Terkadang, Anda perlu mengurutkan berdasarkan satu kolom, dan jika ada nilai yang sama, Anda ingin mengurutkannya lagi berdasarkan kolom kedua. Caranya adalah dengan memberikan sebuah **list nama kolom**.

Contoh: Mengurutkan berdasarkan berat, lalu jika ada yang beratnya sama, urutkan lagi berdasarkan tinggi badan.

```
# Pandas akan mengurutkan 'weight_kg' dulu, baru 'height_cm'
dogs.sort_values(["weight_kg", "height_cm"])
```

Anda bahkan bisa menentukan urutan *ascending* yang berbeda untuk setiap kolomnya:

```
# Urutkan berat secara menaik (True), dan tinggi secara
menurun (False)
dogs.sort_values(["weight_kg", "height_cm"], ascending=[True,
False])
```

Latihan Singkat

Gunakan `df_penjualan` dari proyek kita sebelumnya. Tuliskan kode untuk:

1. Menampilkan transaksi yang diurutkan berdasarkan '**Jumlah Terjual**' dari yang paling sedikit ke paling banyak.
2. Menampilkan transaksi yang diurutkan berdasarkan '**Total Pendapatan**' dari yang paling besar ke paling kecil.

Manipulasi Data: Memilih Sebagian Data (Subsetting)

Subsetting adalah proses mengambil sebagian kecil data dari DataFrame Anda, baik itu kolom tertentu, baris tertentu, atau keduanya.

1. Memilih Kolom

Ada dua cara utama untuk memilih kolom:

a. Memilih Satu Kolom Gunakan satu pasang kurung siku `[]` dengan nama kolom di dalamnya. Hasilnya adalah sebuah **Pandas Series**.

```
# Memilih kolom 'merek' saja
df_penjualan['merek']
```

b. Memilih Beberapa Kolom Untuk memilih lebih dari satu kolom, gunakan dua pasang kurung siku `[[]]`. Anda memberikan sebuah **list nama kolom** di dalamnya. Hasilnya adalah sebuah **DataFrame** baru.

```
# Memilih kolom 'Produk' dan 'Total Pendapatan'
kolom_pilihan = ['Produk', 'Total Pendapatan']
df_penjualan[kolom_pilihan]
```

2. Memilih Baris (Filtering)

Ini adalah kelanjutan dari apa yang sudah kita pelajari. Kita membuat sebuah kondisi boolean untuk memfilter baris yang kita inginkan.

a. Berdasarkan Kondisi Tunggal

```
# Memilih penjualan elektronik saja
df_penjualan[df_penjualan['Kategori'] == 'Elektronik']
```

b. Berdasarkan Beberapa Kondisi (`&` dan `|`) Saat menggabungkan beberapa kondisi, sangat penting untuk membungkus setiap kondisi di dalam tanda kurung `()`.

- `&` untuk **DAN** (kedua kondisi harus benar)
- `|` untuk **ATAU** (salah satu kondisi harus benar)

```
# Memilih produk elektronik DAN jumlah terjual lebih dari 5
df_penjualan[(df_penjualan['Kategori'] == 'Elektronik') &
(df_penjualan['Jumlah Terjual'] > 5)]
```

c. Menggunakan `.isin()` untuk Beberapa Nilai Jika Anda ingin memfilter berdasarkan beberapa nilai dalam satu kolom (misalnya, produk 'Laptop' ATAU 'Mouse'), menggunakan `.isin()` jauh lebih ringkas daripada menggunakan `|` berulang kali.

```
# Memilih semua transaksi untuk produk 'Laptop' atau 'Mouse'
produk_dicari = ['Laptop', 'Mouse']
df_penjualan[df_penjualan['Produk'].isin(produk_dicari)]
```

Latihan Singkat

Gunakan `df_penjualan` lagi.

1. Buat DataFrame baru bernama `info_produk` yang hanya berisi kolom `'Produk'` dan `'Harga Satuan'`.
2. Tampilkan semua transaksi di mana `'Jumlah Terjual'` lebih dari 7 **DAN** `'Harga Satuan'` di bawah 10.000.000.

Manipulasi Data: Menambah Kolom Baru

Seringkali, data mentah yang kita miliki tidak cukup. Kita perlu membuat kolom baru yang berisi informasi turunan dari data yang sudah ada untuk mendapatkan wawasan baru. Misalnya, menghitung total pendapatan dari harga dan jumlah, atau menghitung BMI dari tinggi dan berat.

Cara Paling Umum: Operasi Antar Kolom

Cara termudah dan paling efisien untuk membuat kolom baru adalah dengan melakukan operasi matematika (atau lainnya) pada kolom yang sudah ada. Pandas secara otomatis akan melakukan operasi ini untuk setiap baris (ini disebut **vektorisasi**).

Struktur: `df['nama_kolom_baru'] = df['kolom_A'] + df['kolom_B']`

Contoh: Membuat kolom tinggi anjing dalam satuan meter dari sentimeter.

```
# Membuat kolom 'height_m' dari 'height_cm'
dogs['height_m'] = dogs['height_cm'] / 100
```

Contoh Lebih Kompleks: Menghitung BMI

Anda juga bisa membuat kolom baru dari hasil perhitungan beberapa kolom yang ada.

Contoh: Menghitung *Body Mass Index* (BMI) untuk anjing.

```
# Membuat kolom 'bmi' dari 'weight_kg' dan 'height_m'
dogs['bmi'] = dogs['weight_kg'] / dogs['height_m'] ** 2
```

Setelah kolom baru dibuat, Anda bisa langsung menggunakannya untuk analisis lebih lanjut, seperti *sorting* atau *filtering*.

```
# Memfilter anjing dengan BMI di atas 100
dogs[dogs['bmi'] > 100]
```

Latihan Singkat

Gunakan `df_penjualan` dari latihan kita.

1. Buatlah kolom baru bernama `'Total Pendapatan'` yang berisi hasil perkalian dari kolom `'Harga Satuan'` dan `'Jumlah Terjual'`.
2. Tampilkan 5 baris pertama dari DataFrame setelah kolom baru ditambahkan.

Tantangan Ekstra

Setelah menambahkan kolom `'Total Pendapatan'`, urutkan DataFrame berdasarkan kolom baru tersebut dari yang terbesar ke terkecil.