



LEARNING PROGRESS REVIEW

Week 7

Entropy Team

A large, light gray semi-circle graphic located in the bottom right corner of the slide.

OUR TEAM

Entropy Team



Adhang Muntaha Muhammad

<https://www.linkedin.com/in/adhangmuntaha/>



Aziz Fauzi

<https://www.linkedin.com/in/aziz-fauzi-a6904711b/>



Iwan Wahyu

<https://www.linkedin.com/in/iwan-wahyu-setyawan-506809183>



Marcellina Alvita F

<https://www.linkedin.com/in/marcellina-alvita-faustina-63a284226>



Ramadhan Luthfan

<https://www.linkedin.com/in/luthfan-mahathir-91369b18b>

DAFTAR ISI

1.

Advanced Dataframe

Materi Pandas dataframe
(lanjutan)

2.

Database Programming

Pemrograman *database*
dengan Python

3.

Application Programming Interface

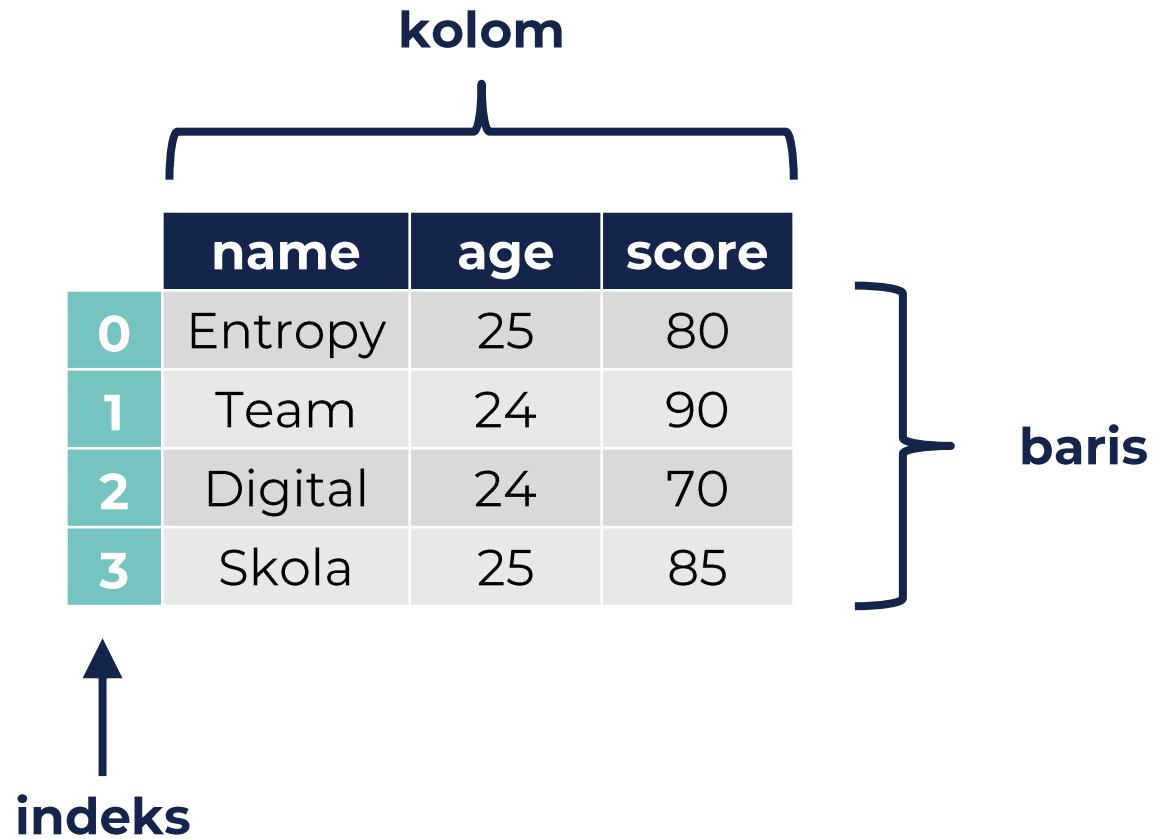
Pengenalan tentang
API

01

Advanced Dataframe

Materi Pandas dataframe
(lanjutan)

Indexing Dataframe



The diagram illustrates a DataFrame with four columns and four rows. The columns are labeled 'name', 'age', and 'score'. The rows are indexed from 0 to 3. An arrow labeled 'indeks' points to the index column. A bracket labeled 'kolom' spans the column headers. A bracket labeled 'baris' spans the row indices.

	kolom		
	name	age	score
0	Entropy	25	80
1	Team	24	90
2	Digital	24	70
3	Skola	25	85

↑ indeks

baris

- Indeks digunakan untuk **mengidentifikasi suatu baris**
- Secara **default**, indeks akan mulai dari **0 sampai n** (tergantung jumlah baris)

Kolom sebagai Indeks

```
in | data.set_index('name')
```

- **Kolom** tertentu dapat diubah **menjadi indeks** dari dataframe

	name	age	score
0	Entropy	25	80
1	Team	24	90
2	Digital	24	70
3	Skola	25	85



name	age	score
Entropy	25	80
Team	24	90
Digital	24	70
Skola	25	85

Mengatur Ulang Indeks

```
in | data.reset_index(drop=True)
```

- Indeks yang tidak beraturan dapat diubah **ke bentuk default** (urut dari 0 sampai n)

	name	age	score
3	Entropy	25	80
10	Team	24	90
2	Digital	24	70
15	Skola	25	85



	name	age	score
0	Entropy	25	80
1	Team	24	90
2	Digital	24	70
3	Skola	25	85

Menghapus Kolom

```
in | data.drop(['age', 'score'], axis=1)
```

- **Kolom** tertentu dapat **dihapus** dengan menentukan **namanya**
- Pastikan **axis = 1**

	name	age	score
0	Entropy	25	80
1	Team	24	90
2	Digital	24	70
3	Skola	25	85



	name
0	Entropy
1	Team
2	Digital
3	Skola

Menghapus Baris

```
in | data.drop([2,3], axis=0)
```

- **Baris** tertentu dapat **dihapus** dengan menentukan **indeksnya**
- Pastikan **axis = 0**

	name	age	score
0	Entropy	25	80
1	Team	24	90
2	Digital	24	70
3	Skola	25	85



	name	age	score
0	Entropy	25	80
1	Team	24	90

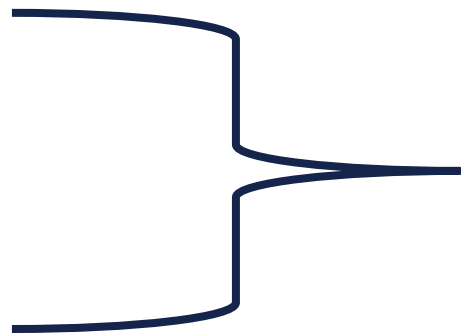
Join

```
in data_left.join(data_right,  
                  lsuffix='_1',  
                  rsuffix='_2',  
                  how='outer')
```

- Digunakan untuk **menggabungkan kolom berdasarkan indeksnya**
- Secara *default*, menggunakan *left join*

	name	age
0	Entropy	25
1	Team	24

	name	age
1	Team	24
2	Digital	24



	name_1	age_1	name_2	age_2
0	Entropy	25.0	NaN	NaN
1	Team	24.0	Team	24.0
2	NaN	NaN	Digital	24.0

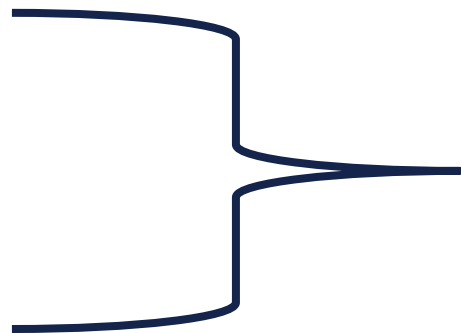
Concatenate – Axis 1

```
in | pd.concat([data_left, data_right], axis=1)
```

- Digunakan untuk **menggabungkan kolom berdasarkan indeksnya**
- Secara *default*, menggunakan *outer join*

	name	age
0	Entropy	25
1	Team	24

	name	age
1	Team	24
2	Digital	24



	name	age	name	age
0	Entropy	25.0	NaN	NaN
1	Team	24.0	Team	24.0
2	NaN	NaN	Digital	24.0

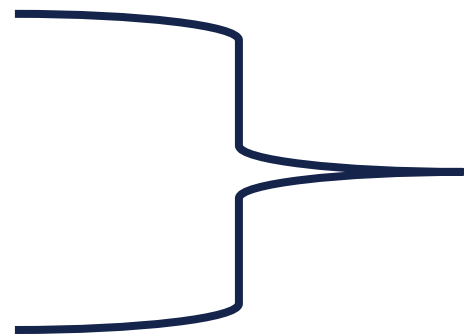
Concatenate – Axis 0

```
in | pd.concat([data_left, data_right], axis=0)
```

- Digunakan untuk **menggabungkan baris**
- Secara *default*, menggunakan *outer join*

	name	age
0	Entropy	25
1	Team	24

	name	age
1	Team	24
2	Digital	24



	name	age
0	Entropy	25
1	Team	24
1	Team	24
2	Digital	24

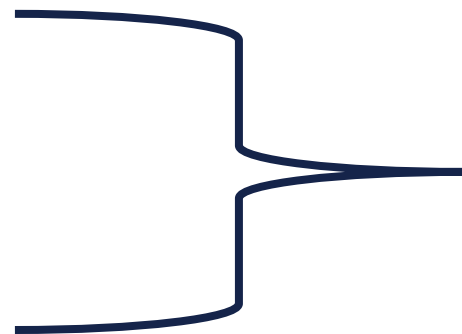
Append

```
in | data_left.append(data_right)
```

- Digunakan untuk **menggabungkan baris**
- Secara *default*, menggunakan *outer join*

	name	age
0	Entropy	25
1	Team	24

	name	age
1	Team	24
2	Digital	24



	name	age
0	Entropy	25
1	Team	24
1	Team	24
2	Digital	24

Merge vs Join vs Concat vs Append

Merge

Menggabungkan **kolom** berdasarkan **kolom tertentu** (konektor)

Concat

Menggabungkan dataframe menggunakan ***outer/ inner join***

- **Axis = 1**
Mirip join()
- **Axis = 0**
Mirip append()

Join

Menggabungkan **kolom** berdasarkan **indeksnya**

Append

Menggabungkan **baris** menggunakan ***outer join***

Pivot Table

```
in | pd.pivot_table(data,  
|                 values='score',  
|                 index='city',  
|                 aggfunc='mean')
```

- Mirip seperti *grouping*, yaitu untuk **mengelompokkan data** berdasarkan kolom tertentu sekaligus **melakukan agregasi**

	name	city	score
0	Entropy	Jogja	80
1	Team	Jakarta	90
2	Digital	Jogja	70
3	Skola	Jakarta	85



city	score
Jakarta	87.5
Jogja	75.0

Melt

```
in | pd.melt(data,  
           id_vars='id',  
           value_vars=['name', 'score'])
```

- Digunakan untuk **menggabungkan data**, di mana kolom tertentu digunakan sebagai **identifier** (id_vars), kolom lain sebagai **value** (value_vars)

	id	name	score
0	1	Entropy	80
1	2	Team	90



	id	variable	value
0	1	name	Entropy
1	2	name	Team
2	1	score	80
3	2	score	90

Fungsi Lambda pada Dataframe

```
in | data['status'] = data['score'].apply(lambda x: 'passed' if x > 70 else 'failed')
```

- Sama seperti fungsi lambda biasa, yaitu untuk membuat **fungsi yang ringkas**

	name	score
0	Entropy	90
1	Team	80
2	Digital	70
3	Skola	60



	name	score	status
0	Entropy	90	passed
1	Team	80	passed
2	Digital	70	failed
3	Skola	60	failed

02

Database Programming

Pemrograman *database*
dengan Python

Menulis File Teks

```
in | path = '/content/entropy.txt'
    |
    | with open(path, 'w') as my_file:
    |     my_file.write('Entropy Team\n')
    |     my_file.write('Data Science Batch 11')
```

- Fungsi `write()` digunakan untuk **membuat** *file* .txt dan **menulis** isinya

entropy.txt	
1	Entropy Team
2	Data Science Batch 11

Membaca File Teks – 1 Baris

```
in | path = '/content/entropy.txt'
    | with open(path, 'r') as my_file:
    |     print(my_file.readline())
out | Entropy Team
```

- Fungsi `readline()` digunakan untuk **membaca** *file* .txt sebanyak **1 baris**

entropy.txt	
1	Entropy Team
2	Data Science Batch 11

Membaca File Teks – Semua Baris (1)

```
in | path = '/content/entropy.txt'
    | with open(path, 'r') as my_file:
    |     print(my_file.readlines())
out | ['Entropy Team\n', 'Data Science Batch 11']
```

- Fungsi `readlines()` digunakan untuk membaca *file* .txt dan **menyimpan** setiap baris **dalam list**

entropy.txt	
1	Entropy Team
2	Data Science Batch 11

Membaca File Teks – Semua Baris (2)

```
in | path = '/content/entropy.txt'
    | with open(path, 'r') as my_file:
    |     print(my_file.read())
out | Entropy Team
    | Data Science Batch 11
```

- Fungsi `read()` digunakan untuk membaca *file* .txt dan **menyimpan** semua datanya **dalam string**

entropy.txt	
1	Entropy Team
2	Data Science Batch 11

Menambahkan Teks

```
in | path = '/content/entropy.txt'  
    |  
    | with open(path, 'a') as my_file:  
    |     my_file.write('\nDigital Skola')
```

- Data baru dapat **ditambahkan** (*append*) ke **baris terakhir** dari *file* .txt

	entropy.txt
1	Entropy Team
2	Data Science Batch 11



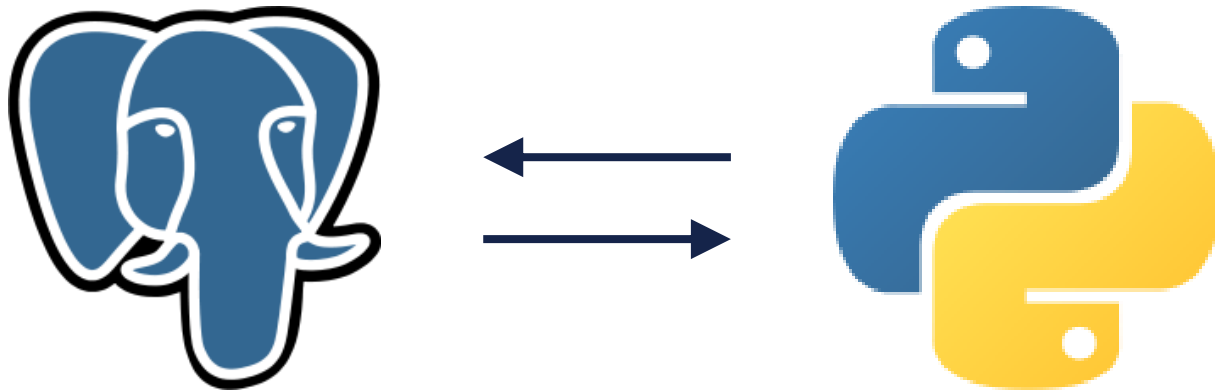
	entropy.txt
1	Entropy Team
2	Data Science Batch 11
3	Digital Skola

Parameter Membuka File Teks

Parameter	Deskripsi
'r'	Membuka file untuk dibaca
'w'	Membuka file untuk ditulis, menggantikan file sebelumnya jika nama file sama
'a'	Membuka file untuk menambahkan isinya (<i>appending</i>)
'x'	Membuka file untuk pembuatan eksklusif, akan gagal jika nama file sudah ada
'b'	Mode biner
't'	Mode teks (<i>default</i>)
'+'	Membuka file untuk memperbaruinya, dapat membaca dan menulis

Sumber: <https://docs.python.org/3/library/functions.html#open>

Database Programming



Psycopg2 merupakan *library* untuk **menjembatani** antara *database* **PostgreSQL** dengan **Python**

Membuat Koneksi

```
in | import psycopg2 as pg
    | import pandas as pd
    |
    | pg_connection = pg.connect(
    |     host = 'database_host',
    |     database = 'database_name',
    |     user = 'username',
    |     password = 'password'
    | )
    |
    | pg_cursor = pg_connection.cursor()
```

- *Method* `connect()` digunakan untuk membuat **koneksi ke database**
- Fungsi `cursor()` digunakan untuk membuat **tempat penyimpanan sementara** bagi data dari *database*

Melakukan Query Dasar

```
in | sql_script = """select name, score, status
    |         from my_table"""
    |
    | pg_cursor.execute(sql_script)
    | pg_cursor.fetchone()
out | ('Entropy',
    | 80,
    | 'passed')
```

- Method `execute()` digunakan untuk **menjalankan script query**
- Fungsi `fetchone()` digunakan untuk **mendapatkan satu baris** data dari hasil *query*

Melakukan Query dengan Pandas

```
in | sql_script = """select name, score, status
    |         from my_table"""
    |
    | data = pd.read_sql_query(sql_script,
    |                        pg_connection)
    |
    | display(data)
```

- Digunakan untuk mengubah tabel hasil *query* **menjadi Pandas dataframe**

	name	score	status
0	Entropy	90	passed
1	Team	80	passed
2	Digital	70	failed
3	Skola	60	failed

Melakukan Perubahan Database

```
in | pg_connection.excute(sql_script)
    | pg_connection.commit()
```

- *Method* commit() digunakan untuk **melakukan perubahan pada *database***
- Jika **tidak dipanggil**, maka semua manipulasi data yang telah dilakukan **tidak akan tersimpan** pada *database*

Menutup Koneksi

```
in | pg_cursor.close()  
   | pg_connection.close()
```

- *Method* `close()` digunakan untuk **menutup koneksi**



03

Application Programming Interface

Pengenalan tentang
API

Application Programming Interface



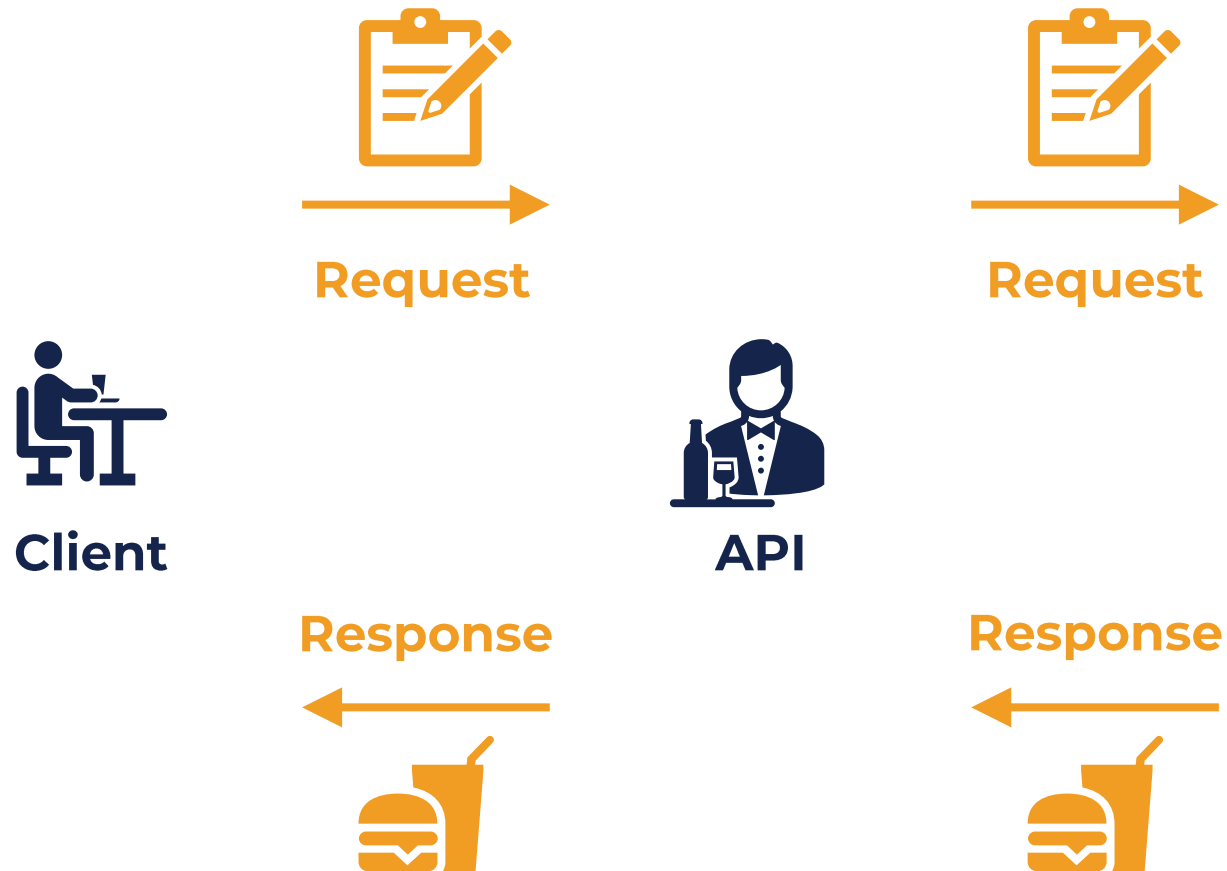
- *Application programming language* (API) merupakan sebuah **perantara software**
- API memungkinkan beberapa *software* **saling berkomunikasi**

Ilustrasi API



- Pelanggan **memesan** makanan melalui pelayan
- Pelayan akan **menyampaikan** ke koki
- Koki **menyerahkan** makanan ke pelayan
- Pelayan **mengantarkan** makanan ke pelanggan

Ilustrasi API



- *Client* melakukan **request** dengan API
- API akan menyampaikan **request** ke server
- Server menyerahkan **response** ke API
- API menyampaikan **response** ke *client*

Keuntungan Penggunaan API

- **Otomatisasi dan efisiensi**
Komputer dapat menjalankan *task* secara otomatis, sehingga lebih efisien
- **Fleksibilitas**
Lebih adaptif terhadap perubahan dan lebih fleksibel
- **Cakupan lebih banyak**
Cakupan distribusi informasi pengguna lebih luas
- **Integrasi**
Suatu komponen lebih mudah untuk disematkan/ diintegrasikan
- **Personalisasi**
Layanan atau konten disesuaikan dengan perilaku *user*

Sumber: <https://www.bbvaapimarket.com/en/api-world/8-advantages-apis-developers/>

Format Data



JSON

- Format seperti dictionary, yaitu pasangan **key-value**
- **{key : value}**
- {"nama" : "Entropy"}



XML

- Format data menggunakan **tag**
- **<key> value </key>**
- <nama> Entropy </nama>

Sumber gambar: <https://iconscout.com/>

API dalam Python



- Terdapat beberapa *library* yang dapat digunakan untuk membuat dan meminta API
- Contoh: flask, fastapi, django

Komponen dari API

- **Endpoint**
Path (URL) yang merujuk ke suatu *resource*
- **Method**
 - Menentukan bagaimana kita akan berinteraksi (CRUD) dengan suatu *endpoint*
- **Body (data)**
 - Tempat menyimpan data ketika menggunakan *method* yang melibatkan perubahan data, seperti POST dan PUT
 - REST API menggunakan JSON sebagai format data
- **Header**
Tempat menyimpan *metadata*

Sumber:

<https://www.nylas.com/blog/use-python-requests-module-rest-apis>

<https://developer.cybersource.com/api/developer-guides/dita-gettingstarted/RESTComponents.html>

Method dalam REST API

HTTP Verb	CRUD	Deskripsi
POST	Create	Menambahkan data baru
GET	Read	Membaca data tertentu
PUT	Update	Mengubah data tertentu secara keseluruhan (<i>replace</i>)
PATCH	Update	Mengubah data tertentu secara parsial (<i>modify</i>)
DELETE	Delete	Menghapus data tertentu

CRUD = Create, Read, Update, Delete

Status Code

- *Status code* digunakan untuk **melihat *response* dari *endpoint*** ketika kita membuat suatu *request*. Status code dibagi menjadi 5 kelompok:
 - **1xx – *information response***
Request didapatkan, proses dilanjutkan
 - **2xx – *successful***
Request berhasil didapat, dipahami, dan diterima
 - **3xx – *redirection***
Tindakan lebih lanjut perlu dilakukan untuk menyelesaikan *request*
 - **4xx – *client error***
Request berisikan *bad syntax* atau tidak dapat dipenuhi
 - **5xx – *server error***
Server gagal memenuhi *request* yang terlihat valid

THANKS

Entropy Team

CREDITS: This presentation template was originally created by **Slidesgo**, including icons by **Flaticon**, and infographics & images by **Freepik**