

LAPORAN TEORI DATA WAREHOUSE



Nama : Michael Christia Putra
NIM : 202331203
Kelas : F
Dosen : Dr. Dra. Dwina Kuswardani, M.Kom
No. PC : 21
Asisten : 1. Alfian Riswandi
2. Maurien Andriesta
3. Tarissa Nurhapsari Laksono
4. Muhammad Rifqi Apriansyah

**INSTITUT TEKNOLOGI PLN
TEKNIK INFORMATIKA
2024/2025**

Pada praktikum kali ini kita akan membuat program web scarping dan proses ETL yang mendasari pelaksanaan kuliah praktisi untuk proses ETL (Extract, Transform, Load) data cuaca dari Badan Meteorologi, Klimatologi, dan Geofisika (BMKG). Pembahasan mencakup arsitektur proses ETL, teknologi antarmuka aplikasi (API) sebagai sumber data, sistem manajemen database PostgreSQL sebagai tujuan penyimpanan data, serta ekosistem Python dan Jupyter Notebook sebagai platform implementasi.

Pengertian ETL (Extract, Transform, Load)

ETL, yang merupakan akronim dari *Extract*, *Transform*, *Load*, adalah sebuah paradigma fundamental dalam rekayasa data yang merujuk pada proses tiga tahap untuk mengintegrasikan data dari berbagai sumber ke dalam sebuah penyimpanan data terpusat, yang umumnya berupa *data warehouse* atau *data mart*. Tujuan utama dari proses ETL adalah untuk mengkonsolidasikan data yang heterogen—baik dari segi format, struktur, maupun platform—menjadi sebuah himpunan data yang terstandardisasi, bersih, konsisten, dan siap guna untuk keperluan analisis, *business intelligence* (BI), pelaporan, maupun aplikasi *machine learning*. Dalam lingkungan bisnis dan saintifik modern, data seringkali tersebar di berbagai sistem operasional: database transaksional (OLTP), aplikasi pihak ketiga yang diekspos melalui API, *file log* dari server, dokumen semi-terstruktur seperti JSON dan XML, hingga *spreadsheet* konvensional. Proses ETL berfungsi sebagai tulang punggung (*backbone*) yang menjembatani kesenjangan antara sistem-sistem sumber ini dengan sistem analitik, memastikan bahwa para pengambil keputusan memiliki akses ke data yang akurat, terpercaya, dan komprehensif sebagai dasar analisis strategis.

Setiap tahapan dalam proses ETL memiliki fungsi dan tantangan teknisnya masing-masing. Tahap pertama, **Extract**, berfokus pada akuisisi atau pengambilan data dari satu atau lebih sistem sumber. Proses ini tidak sekadar menyalin data, melainkan melibatkan konektivitas ke berbagai platform dengan protokol yang berbeda, validasi data awal untuk memastikan data yang ditarik tidak korup, serta penentuan strategi ekstraksi—apakah melakukan penarikan penuh (*full extraction*) atas seluruh data atau hanya penarikan inkremental (*incremental extraction*) terhadap data yang baru atau berubah sejak proses terakhir. Dalam praktikum ini, tahap Ekstraksi diwujudkan melalui interaksi dengan *endpoint* REST API yang disediakan oleh BMKG, di mana data mentah dalam format JSON ditarik menggunakan protokol HTTP. Tahap kedua, **Transform**, adalah inti dari proses ETL di mana data mentah diolah menjadi data yang bernilai. Transformasi dapat mencakup berbagai operasi, seperti *cleansing* (membersihkan data dari anomali, nilai null, atau inkonsistensi), *filtering* (memilih kolom atau baris yang relevan), *standardization* (menyamakan format data, seperti mengonversi string tanggal menjadi tipe data *TIMESTAMP* atau menyeragamkan unit pengukuran), *enrichment* (memperkaya data dengan menggabungkannya dengan data dari sumber lain), dan *structuring* (mengubah format data, misalnya dari JSON yang bersifat hirarkis menjadi format tabular yang datar). Pada praktikum ini, contoh transformasi yang dilakukan adalah mem-parsing struktur JSON, menyeleksi atribut-atribut cuaca yang relevan, serta menangani nilai koordinat yang kosong sebelum dimuat ke database. Tahap terakhir, **Load**, adalah proses memasukkan data yang telah ditransformasi ke dalam sistem penyimpanan target. Proses ini harus dirancang secara efisien untuk menangani volume data yang besar dan memastikan integritas data tetap terjaga, misalnya dengan menggunakan mekanisme *bulk loading*, mengelola *primary key* dan *unique constraints*, serta menerapkan strategi pembaruan data seperti *insert-update* (upsert). Dalam implementasi proyek ini, data

dimuat ke dalam tabel-tabel pada database PostgreSQL, dengan memanfaatkan perintah ON CONFLICT DO NOTHING untuk menangani duplikasi data secara idempoten.

Apa Itu API dan REST API

Application Programming Interface (API) secara fundamental adalah sebuah kontrak atau antarmuka komputasi yang mendefinisikan interaksi antara beberapa perantara perangkat lunak. API menetapkan serangkaian aturan, protokol, dan *tools* yang memungkinkan aplikasi atau sistem yang berbeda untuk saling berkomunikasi dan bertukar data tanpa harus mengetahui kompleksitas implementasi internal masing-masing. Analogi yang sering digunakan adalah menu di sebuah restoran; pelanggan (klien) tidak perlu tahu bagaimana dapur (server) memasak hidangan, mereka hanya perlu memesan dari menu (API) sesuai dengan aturan yang ada untuk mendapatkan hidangan (data atau layanan) yang diinginkan. Dalam konteks rekayasa perangkat lunak, API memungkinkan modularitas dan abstraksi, di mana pengembang dapat memanfaatkan fungsionalitas yang disediakan oleh aplikasi lain—seperti layanan peta, sistem pembayaran, atau dalam kasus ini, data cuaca dari BMKG—melalui serangkaian *endpoint* yang terdokumentasi dengan baik. Keberadaan API mendorong inovasi dan interoperabilitas, memungkinkan terciptanya ekosistem aplikasi yang saling terhubung dan lebih kaya fungsionalitas.

Di antara berbagai jenis arsitektur API, *Representational State Transfer* (REST) telah menjadi standar *de facto* untuk layanan berbasis web karena kesederhanaan, skalabilitas, dan kemampuannya untuk berjalan di atas protokol HTTP yang sudah ada di mana-mana. REST bukanlah sebuah protokol yang kaku, melainkan sebuah gaya arsitektur perangkat lunak yang dirancang oleh Roy Fielding. Sebuah API dianggap "RESTful" jika mematuhi serangkaian batasan (constraints) arsitektural, di antaranya adalah arsitektur klien-server, komunikasi *stateless* (setiap permintaan dari klien ke server harus berisi semua informasi yang diperlukan untuk memahami dan memproses permintaan tersebut, tanpa server perlu menyimpan konteks klien), *cacheability* (respons dari server dapat di-cache oleh klien), dan antarmuka yang seragam (*uniform interface*). Antarmuka seragam ini dicapai dengan memanfaatkan metode HTTP standar sebagai representasi operasi, yaitu GET untuk mengambil data, POST untuk membuat data baru, PUT/PATCH untuk memperbarui data, dan DELETE untuk menghapus data. Dalam praktikum ini, kita secara ekstensif berinteraksi dengan REST API yang disediakan oleh BMKG. Metode GET digunakan untuk mengekstrak data prakiraan cuaca dari *endpoint* seperti `/api/v1/public/weather/weekly-temperature`, sementara metode POST digunakan untuk berinteraksi dengan *endpoint* `/ndf/cgms/weather/forward` untuk mendapatkan data artikel.

Data yang dipertukarkan melalui REST API umumnya diformat dalam *JavaScript Object Notation* (JSON), sebuah format pertukaran data yang ringan, mudah dibaca oleh manusia, dan mudah diurai oleh mesin. Struktur JSON berbasis pada pasangan kunci-nilai (*key-value pairs*) dan tipe data terurut seperti *array*, yang secara langsung memetakan ke struktur data yang umum digunakan di hampir semua bahasa pemrograman modern. Keunggulan JSON dibandingkan format sebelumnya seperti XML adalah verbositasnya yang lebih rendah, sehingga menghasilkan ukuran *payload* yang lebih kecil dan transfer data yang lebih cepat. Dalam konteks praktikum ini, seluruh respons dari *endpoint* API BMKG diterima dalam format JSON. Pustaka `requests` pada Python digunakan untuk melakukan panggilan HTTP dan menerima respons ini, kemudian data JSON tersebut diurai (di-*parse*) menjadi objek kamus

(*dictionary*) dan daftar (*list*) Python. Kemudahan dalam memanipulasi struktur data ini di Python menjadi fondasi penting dalam tahap Transformasi, di mana data yang diekstrak kemudian dapat dengan mudah dibersihkan, difilter, dan disusun ulang sebelum dimuat ke dalam database PostgreSQL.

PostgreSQL

PostgreSQL, yang sering disebut sebagai "Postgres", adalah sebuah sistem manajemen database relasional objek (*Object-Relational Database Management System* atau ORDBMS) sumber terbuka (*open-source*) yang canggih dan sangat andal. Dikenal luas karena kepatuhannya yang kuat terhadap standar SQL dan arsitekturnya yang kokoh, PostgreSQL telah membangun reputasi selama lebih dari 30 tahun sebagai database yang stabil, tangguh, dan kaya fitur. Arsitekturnya didasarkan pada prinsip-prinsip ACID (*Atomicity, Consistency, Isolation, Durability*), yang menjamin keandalan transaksi dan integritas data, sebuah karakteristik krusial baik untuk sistem pemrosesan transaksi online (OLTP) maupun untuk sistem analitik seperti *data warehouse*. Didukung oleh komunitas pengembang global yang aktif, PostgreSQL terus berevolusi dengan penambahan fitur-fitur modern, menjadikannya pilihan yang sangat kompetitif dibandingkan dengan sistem database komersial. Skalabilitasnya, baik secara vertikal (dengan menambah sumber daya pada satu server) maupun horizontal (melalui berbagai teknik replikasi dan *sharding*), membuatnya cocok untuk menangani volume data dari skala kecil hingga sangat besar.

Dalam konteks proyek ETL dan *data warehousing*, PostgreSQL menawarkan sejumlah fitur unggulan yang menjadikannya pilihan yang sangat tepat sebagai sistem penyimpanan target. Pertama, dukungannya yang ekstensif terhadap berbagai tipe data, jauh melampaui tipe data primitif. Selain tipe standar seperti VARCHAR, INTEGER, TIMESTAMP, dan NUMERIC (yang kita gunakan untuk memastikan presisi data koordinat), PostgreSQL juga memiliki dukungan kelas satu untuk tipe data semi-terstruktur melalui tipe JSON dan JSONB. Tipe JSONB (JSON biner) sangat efisien karena menyimpan data JSON dalam format terurai yang memungkinkan pembuatan indeks dan kueri langsung ke dalam atribut-atribut JSON, sebuah kemampuan yang sangat berharga saat menangani data mentah dari API. Kedua, kapabilitas kueri analitiknya yang kuat. PostgreSQL mendukung fungsi jendela (*window functions*), *Common Table Expressions* (CTEs), kueri rekursif, dan beragam fungsi agregasi yang kompleks, yang semuanya merupakan perangkat esensial untuk melakukan transformasi dan agregasi data dalam tahap analitik. Dalam praktikum ini, PostgreSQL dipilih sebagai destinasi data karena keandalannya. Skema tabel dirancang dengan tipe data yang spesifik untuk setiap atribut data cuaca, dan batasan-batasan seperti PRIMARY KEY dan UNIQUE ditegakkan untuk memastikan integritas dan mencegah duplikasi data, di mana ON CONFLICT digunakan sebagai strategi pemuatan data yang idempoten.

Ekosistem PostgreSQL yang kaya dan dapat diperluas (*extensible*) juga menjadi nilai tambah yang signifikan. PostgreSQL memungkinkan pengguna untuk mendefinisikan tipe data, fungsi, dan bahkan metode indeks mereka sendiri. Selain itu, terdapat banyak ekstensi (*extensions*) yang matang dan kuat yang dapat diintegrasikan untuk memperluas fungsionalitasnya. Contoh yang paling relevan dengan data BMKG adalah PostGIS, sebuah ekstensi yang mengubah PostgreSQL menjadi database spasial berfitur lengkap, memungkinkan penyimpanan, pengindeksan, dan kueri data geografis yang kompleks (seperti mencari semua stasiun dalam radius tertentu). Ekstensi lain seperti TimescaleDB

mengoptimalkan PostgreSQL untuk beban kerja data deret waktu (*time-series*), yang sangat cocok untuk data pengamatan cuaca yang dicatat dari waktu ke waktu. Kemampuan ini menunjukkan bahwa PostgreSQL tidak hanya berfungsi sebagai "gudang" data statis, tetapi juga sebagai platform analitik yang aktif dan dapat beradaptasi dengan kebutuhan analisis data yang semakin kompleks.

Jupyter Notebook & Python

Python telah mengukuhkan posisinya sebagai bahasa pemrograman dominan dalam ranah ilmu data, rekayasa data, dan komputasi ilmiah. Popularitas ini didorong oleh beberapa faktor utama: sintaksisnya yang bersih dan mudah dipelajari, sifatnya sebagai bahasa interpretatif yang memfasilitasi pengembangan iteratif, dan yang terpenting, ekosistem pustaka pihak ketiga yang sangat luas dan matang. Bagi seorang insinyur data, Python berfungsi sebagai "lem" serbaguna yang dapat menghubungkan berbagai sistem dan teknologi dengan mudah. Untuk proyek ETL ini, beberapa pustaka kunci menjadi tulang punggung implementasi. Pustaka `requests` menyederhanakan proses interaksi dengan REST API, mengabstraksikan kompleksitas permintaan HTTP GET dan POST serta penanganan respons JSON. Pustaka `psycopg2` berfungsi sebagai *driver* database yang andal dan efisien, menyediakan antarmuka untuk terhubung ke PostgreSQL, mengeksekusi perintah SQL untuk membuat tabel (`CREATE TABLE`), memuat data (`INSERT`), dan memodifikasi skema (`ALTER TABLE`). Terakhir, pustaka `pandas` menyediakan struktur data berkinerja tinggi yang disebut `DataFrame`, yang merupakan alat fundamental untuk tahap Transformasi. `DataFrame` memungkinkan manipulasi data tabular—seperti pembersihan, pemfilteran, dan agregasi—dengan cara yang ekspresif dan efisien.

Jupyter Notebook melengkapi kekuatan Python dengan menyediakan lingkungan pengembangan komputasi interaktif berbasis web. Berbeda dengan lingkungan pengembangan terintegrasi (IDE) tradisional di mana kode ditulis dalam skrip panjang dan dieksekusi sekaligus, Jupyter menggunakan model "sel". Setiap sel dapat berisi kode, teks naratif (dalam format Markdown), atau visualisasi, dan dapat dieksekusi secara independen. Paradigma ini sangat cocok untuk alur kerja ETL dan analisis data. Pengembang dapat menulis dan menjalankan kode untuk tahap Extract dalam satu sel, kemudian langsung memeriksa data mentah yang diterima. Di sel berikutnya, mereka dapat menulis kode untuk tahap Transform dan segera menampilkan hasilnya dalam bentuk tabel `DataFrame` untuk memverifikasi bahwa transformasi telah berjalan dengan benar. Proses yang terkotak-kotak dan interaktif ini secara drastis mengurangi waktu siklus pengembangan-debug, karena kesalahan dapat diisolasi dan diperbaiki pada setiap langkah kecil, bukan setelah seluruh skrip besar gagal. Dalam praktikum ini, pendekatan sel-demi-sel ini terbukti sangat krusial, terutama saat melakukan *debugging* error seperti numeric field overflow atau invalid input syntax, di mana kita dapat memeriksa data yang akan dimuat tepat sebelum proses *Load* dieksekusi.

Sinergi antara Python dan Jupyter Notebook menciptakan platform yang ideal untuk pengembangan dan pembuatan prototipe alur kerja ETL. Interaktivitas Jupyter memungkinkan eksplorasi data dan eksperimen logika transformasi secara *real-time*, sementara kekuatan pustaka-pustaka Python menangani tugas-tugas berat seperti komunikasi jaringan, interaksi database, dan manipulasi data bervolume besar. Kemampuan untuk menyisipkan dokumentasi Markdown di antara sel-sel kode, seperti yang dilakukan dalam laporan ini, juga mempromosikan pembuatan "notebook yang dapat dieksekusi" (*executable notebooks*), yang

berfungsi sebagai artefak yang dapat direproduksi, didokumentasikan, dan dibagikan. Notebook ini tidak hanya berisi kode, tetapi juga narasi tentang apa yang dilakukan kode tersebut, mengapa dilakukan, dan apa hasilnya. Dengan demikian, kombinasi Python dan Jupyter Notebook tidak hanya menjadi alat implementasi, tetapi juga menjadi medium yang kuat untuk komunikasi dan kolaborasi dalam proyek-proyek yang berpusat pada data.

LAPORAN PRAKTIKUM

DATA WAREHOUSE



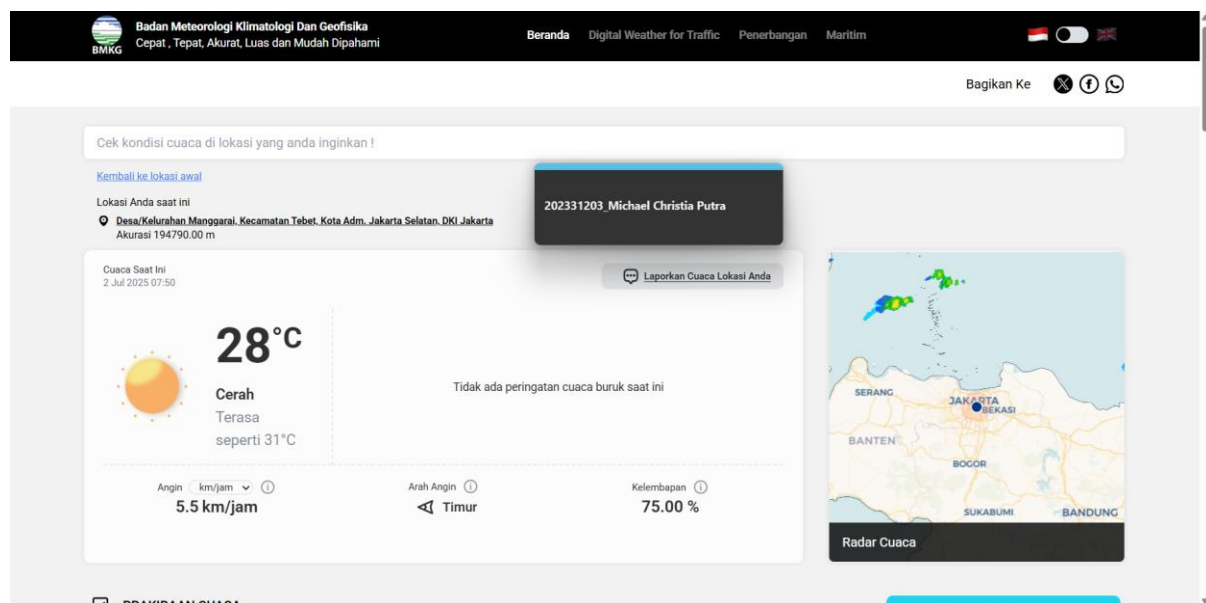
Nama : Michael Christia Putra
NIM : 202331203
Kelas : F
Dosen : Dr. Dra. Dwina Kuswardani, M.Kom
No. PC : 21
Asisten : 1. Alfian Riswandi
2. Maurien Andriesta
3. Tarissa Nurhapsari Laksono
4. Muhammad Rifqi Apriansyah

INSTITUT TEKNOLOGI PLN
TEKNIK INFORMATIKA

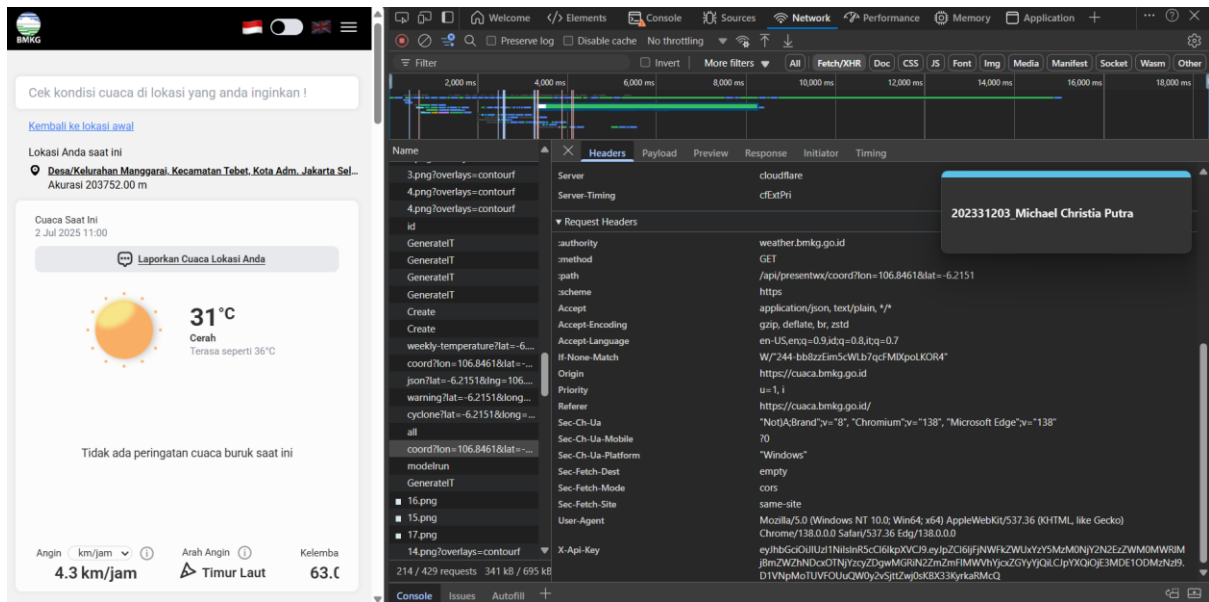
2024/2025

Deskripsi Web (Data)

Sumber data utama untuk proyek ETL ini adalah situs web resmi Badan Meteorologi, Klimatologi, dan Geofisika (BMKG), yang dapat diakses melalui domain cuaca.bmkg.go.id. Situs ini dipilih karena merupakan sumber data cuaca yang paling otoritatif dan relevan untuk wilayah Indonesia, menyajikan informasi yang bersifat real-time dan dapat diakses oleh publik. Data yang menjadi target untuk diekstraksi mencakup berbagai parameter cuaca, seperti prakiraan temperatur mingguan, prakiraan cuaca per jam untuk lokasi spesifik, data observasi untuk penerbangan, serta konten informatif berupa video dan artikel prospek cuaca. Data-data ini disajikan secara dinamis pada antarmuka web, yang mengindikasikan bahwa situs ini memuat data melalui mekanisme pemanggilan data di latar belakang, bukan data statis yang tertanam pada halaman HTML.



Untuk mengidentifikasi bagaimana cara mendapatkan data dinamis tersebut, dilakukan proses inspeksi jaringan (*network inspection*) menggunakan *Developer Tools* yang tersedia pada peramban web modern (misalnya, Google Chrome atau Mozilla Firefox). Dengan membuka tab "Network" dan memfilter permintaan jenis "Fetch/XHR", seluruh komunikasi data antara peramban (klien) dan server BMKG dapat dipantau. Dari pemantauan ini, teridentifikasi beberapa *endpoint* API (Application Programming Interface) yang digunakan oleh situs web untuk mengambil data cuaca. Proses ini mengungkapkan bahwa data tidak diambil dengan men-scraping konten HTML, melainkan dengan meniru panggilan API yang dilakukan oleh *frontend* situs web. Lebih lanjut, inspeksi pada bagian *Request Headers* dari setiap panggilan API yang berhasil menunjukkan adanya sebuah header otentikasi kustom, yaitu *x-api-key*, yang berisi sebuah token JWT (*JSON Web Token*). Penemuan ini sangat krusial, karena menandakan bahwa API tersebut terproteksi dan setiap permintaan dari skrip Python harus menyertakan header *x-api-key* ini agar berhasil diautorisasi oleh server BMKG.



Program Python

Implementasi keseluruhan proses ETL dilakukan menggunakan bahasa pemrograman Python dalam lingkungan Jupyter Notebook. Pendekatan ini dipilih karena sifatnya yang interaktif, memungkinkan setiap tahapan proses—mulai dari koneksi database hingga ekstraksi, transformasi, dan pemuatan data untuk setiap API—dapat dieksekusi dan diverifikasi secara modular dalam sel-sel terpisah. Berikut adalah penjelasan alur program untuk setiap sel kode.

Sel 1: Impor Pustaka

Sel pertama berfungsi untuk mengimpor semua pustaka Python yang diperlukan. requests digunakan untuk melakukan panggilan HTTP ke API BMKG. psycopg2 adalah driver untuk menghubungkan Python dengan database PostgreSQL. pandas menyediakan struktur data DataFrame yang sangat esensial untuk tahap transformasi. datetime dari pustaka standar digunakan untuk manipulasi data tanggal dan waktu, sementara display dari IPython.display digunakan untuk menampilkan DataFrame dengan format yang lebih baik di dalam notebook.

```
[25]: import requests
import psycopg2
import pandas as pd
from psycopg2 import sql
from datetime import datetime
from IPython.display import display

print("Pustaka berhasil diimpor.")
#202331203_Michael Christia Putra

Pustaka berhasil diimpor.
```

202331203_Michael Christia Putra

Sel 2 & 3: Konfigurasi dan Koneksi Database

Sel kedua mendefinisikan detail koneksi database dalam sebuah dictionary DB_SETTINGS, memisahkan konfigurasi dari logika utama. Sel ketiga menggunakan

konfigurasi ini untuk membuat koneksi ke server PostgreSQL. Di sel ini juga dieksekusi perintah-perintah DDL (Data Definition Language) CREATE TABLE IF NOT EXISTS. Perintah ini berfungsi untuk menyiapkan skema database, memastikan semua tabel yang dibutuhkan telah tersedia sebelum proses pemuatan data dimulai. Penggunaan klausa IF NOT EXISTS membuat skrip ini aman untuk dijalankan berulang kali tanpa menimbulkan error.

```
DB_SETTINGS = {  
    "dbname": "weather_db",  
    "user": "postgres",  
    "password": "1234",  
    "host": "localhost",  
    "port": "5432"  
}  
  
print("Pengaturan database telah dikonfigurasi.")  
#202331203_Michael Christia Putra  
  
Pengaturan database telah dikonfigurasi.  
  
try:  
    conn = psycopg2.connect(**DB_SETTINGS)  
    cur = conn.cursor()  
    print("Koneksi ke database PostgreSQL berhasil.")  
except psycopg2.OperationalError as e:  
    print(f"Error: Gagal terhubung ke database. Periksa pengaturan koneksi Anda.")  
    print(f"Detail: {e}")  
    conn = None  
  
#202331203_Michael Christia Putra
```

Sel 4 s/d 8: Proses ETL untuk Setiap API

Lima sel berikutnya merupakan inti dari alur kerja ETL, di mana setiap sel bertanggung jawab untuk satu sumber data API:

1. Tahap Ekstraksi (Extract): Kode pada setiap sel memulai proses dengan mendefinisikan URL endpoint target dan headers yang berisi x-api-key. Fungsi requests.get() atau requests.post() kemudian digunakan untuk mengambil data mentah dari API.
2. Tahap Transformasi (Transform): Setelah respons JSON berhasil diterima, data tersebut diurai. Logika transformasi kemudian diterapkan: menyeleksi kunci-kunci yang relevan, membersihkan data (seperti mengubah string kosong menjadi nilai None untuk kolom numerik), mengonversi tipe data (misalnya, string tanggal menjadi objek datetime), dan menyusun ulang data ke dalam struktur tabular menggunakan Pandas DataFrame. DataFrame ini kemudian ditampilkan untuk verifikasi visual.
3. Tahap Pemuatan (Load): Kode selanjutnya melakukan iterasi pada setiap baris DataFrame dan mengeksekusi perintah SQL INSERT untuk memasukkan data ke dalam tabel PostgreSQL yang sesuai. Klausa ON CONFLICT DO NOTHING digunakan untuk menangani integritas data secara efisien, mencegah duplikasi jika data yang sama diekstraksi kembali.

```

try:
    response = requests.get(url, headers=headers)
    response.raise_for_status()

    data = response.json().get('data', {}).get('daily', [])
    print(f"Ekstraksi data dari {url} berhasil.")

    records = []
    for record in data:
        records.append({
            "forecast_date": datetime.fromisoformat(record['date'].replace('Z', '+00:00')).date(),
            "temperature_celsius": record['temperature'],
            "latitude": lat,
            "longitude": lon
        })

    df_temp = pd.DataFrame(records)
    print("\nData yang akan dimasukkan (Transform):")
    display(df_temp)

    if not df_temp.empty:
        for index, row in df_temp.iterrows():
            cur.execute(
                """
                INSERT INTO weekly_temperature (forecast_date, temperature_celsius, latitude, longitude)
                VALUES (%s, %s, %s, %s) ON CONFLICT (forecast_date) DO NOTHING;
                """
                , (row['forecast_date'], row['temperature_celsius'], row['latitude'], row['longitude']))
            conn.commit()
        print(f"\nLoad berhasil: {len(df_temp)} baris data dimasukkan/dilewati ke tabel 'weekly_temperature'.")

    except Exception as e:
        print(f"Terjadi error: {e}")

```

202331203_Michael Christia Putra

--- Memulai ETL 1: Temperatur Mingguan ---

Ekstraksi data dari <https://cuaca.bmkg.go.id/api/v1/public/weather/weekly-temperature?lat=-6.2151&lon=106.8461> berhasil.

Data yang akan dimasukkan (Transform):

	forecast_date	temperature_celsius	latitude	longitude
0	2025-07-02	26.088803	-6.2151	106.8461
1	2025-07-03	25.918488	-6.2151	106.8461
2	2025-07-04	25.344962	-6.2151	106.8461
3	2025-07-05	25.569670	-6.2151	106.8461
4	2025-07-06	25.290600	-6.2151	106.8461
5	2025-07-07	26.914716	-6.2151	106.8461
6	2025-07-08	26.154286	-6.2151	106.8461

202331203_Michael Christia Putra

Load berhasil: 7 baris data dimasukkan/dilewati ke tabel 'weekly_temperature'.

--- Memulai ETL 2: Prakiraan Cuaca per Jam ---

Ekstraksi data dari <https://cuaca.bmkg.go.id/api/df/v1/forecast/coord?lon=106.8461&lat=-6.2151> berhasil.

Data yang akan dimasukkan (Transform):

	datetime_utc	temperature_celsius	weather_description	desa
0	2025-07-02 03:00:00+00:00	30	Cerah	Manggarai
1	2025-07-02 04:00:00+00:00	31	Cerah	Manggarai
2	2025-07-02 05:00:00+00:00	31	Cerah	Manggarai
3	2025-07-02 06:00:00+00:00	31	Hujan Sedang	Manggarai
4	2025-07-02 07:00:00+00:00	29	Hujan Sedang	Manggarai

202331203_Michael Christia Putra

Load berhasil: 14 baris data dimasukkan/dilewati ke tabel 'hourly_forecast'.

202331203_Michael Christia Putra

```
--- Memulai ETL 3: Observasi Penerbangan ---  
Ekstraksi data dari https://cuaca.bmkg.go.id/api/v1/aviation/latest/observation.json berhasil.  
  
Data yang akan dimasukkan (Transform):  


|   | icao_id | observed_time       | station_name                             | latitude | longitude | weather         | temperature_celsius | pressure_hpa |
|---|---------|---------------------|------------------------------------------|----------|-----------|-----------------|---------------------|--------------|
| 0 | WITN    | 2025-07-02 09:00:00 | Maimun Saleh - Sabang                    | 5.876667 | 95.340000 | Berawan         | 28                  | 1011         |
| 1 | WITT    | 2025-07-02 09:00:00 | Sultan Iskandar Muda - Banda Aceh        | 5.520000 | 95.420833 | Berawan         | 26                  | 1009         |
| 2 | WITC    | 2025-07-02 09:00:00 | Cut Nyak Dien Nagan Raya - Meulaboh, NAD | 4.045000 | 96.249444 | Berawan         | 26                  | 1011         |
| 3 | WIMA    | 2025-07-02 09:00:00 | Malikus Saleh - Lhokseumawe              | 5.225833 | 96.948611 | Cerah - berawan | 29                  | 1009         |
| 4 | WIMB    | 2025-07-02 09:00:00 | Binaka - Gunung Sitoli                   | 1.166389 | 97.704444 | Cerah - berawan | 28                  | 1011         |

  
Load berhasil: 119 baris data dimasukkan/dilewati ke tabel 'aviation_observation'.  
  
--- Memulai ETL 4: Video Cuaca Terbaru ---  
Ekstraksi data dari https://cuaca.bmkg.go.id/api/v1/public/weather/video-latest?hashtag=infobmkgpws berhasil.  
  
Data yang akan dimasukkan (Transform):  

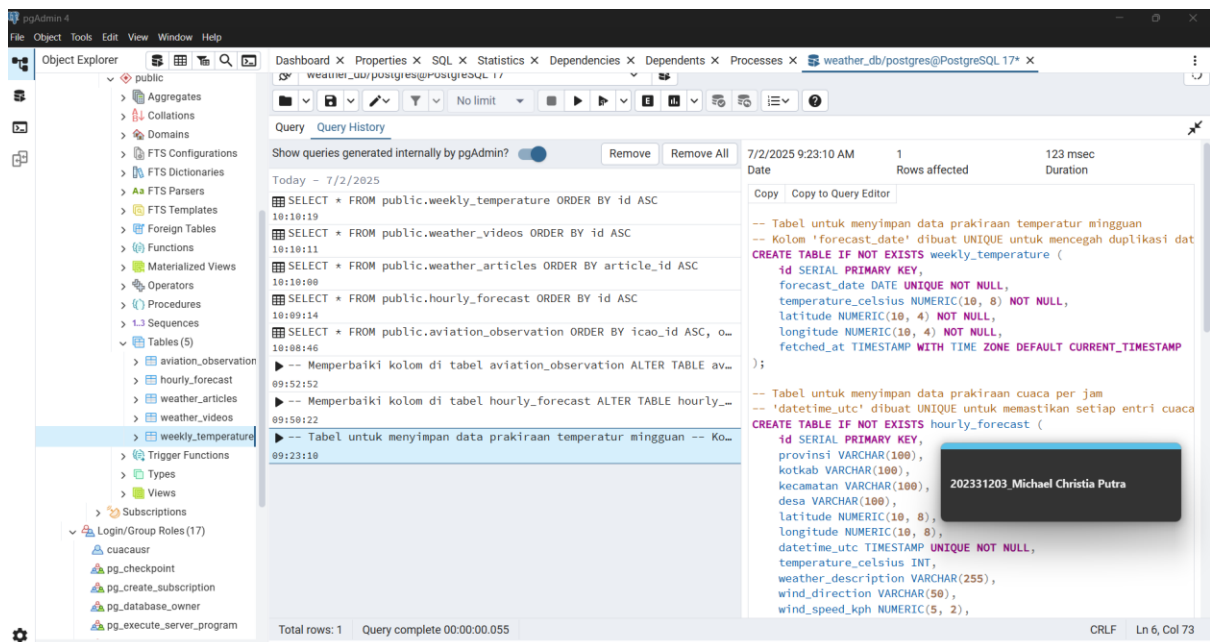

|   | title                                            | video_url                                   |
|---|--------------------------------------------------|---------------------------------------------|
| 0 | Prakiraan Cuaca Esok Hari, Rabu, 02 Juli 2025.   | https://www.youtube.com/watch?v=PjhSUK8LprQ |
| 1 | Prakiraan Cuaca Esok Hari, Selasa, 01 Juli 2025  | https://www.youtube.com/watch?v=IX5FFHpwave |
| 2 | Prakiraan Cuaca Esok Hari, Senin, 30 Juni 2025.  | https://www.youtube.com/watch?v=LGB3bBt7ajc |
| 3 | Prakiraan Cuaca Esok Hari, Minggu, 29 Juni 2025. | https://www.youtube.com/watch?v=cD7vwjRVRUc |
| 4 | Prakiraan Cuaca Esok Hari Sabtu, 28 Juni 2025    | https://www.youtube.com/watch?v=YYnDXzyWEf0 |
| 5 | Prakiraan Cuaca Esok Hari, Jumat, 27 Juni 2025   | https://www.youtube.com/watch?v=g8lvWXNSBrA |

  
Load berhasil: 6 baris data dimasukkan/dilewati ke tabel 'weather_videos'.  
  
if 'conn' in locals() and conn:  
    cur.close()  
    conn.close()  
    print("Koneksi ke database telah ditutup. Proses ETL selesai!")  
  
Koneksi ke database telah ditutup. Proses ETL selesai!
```

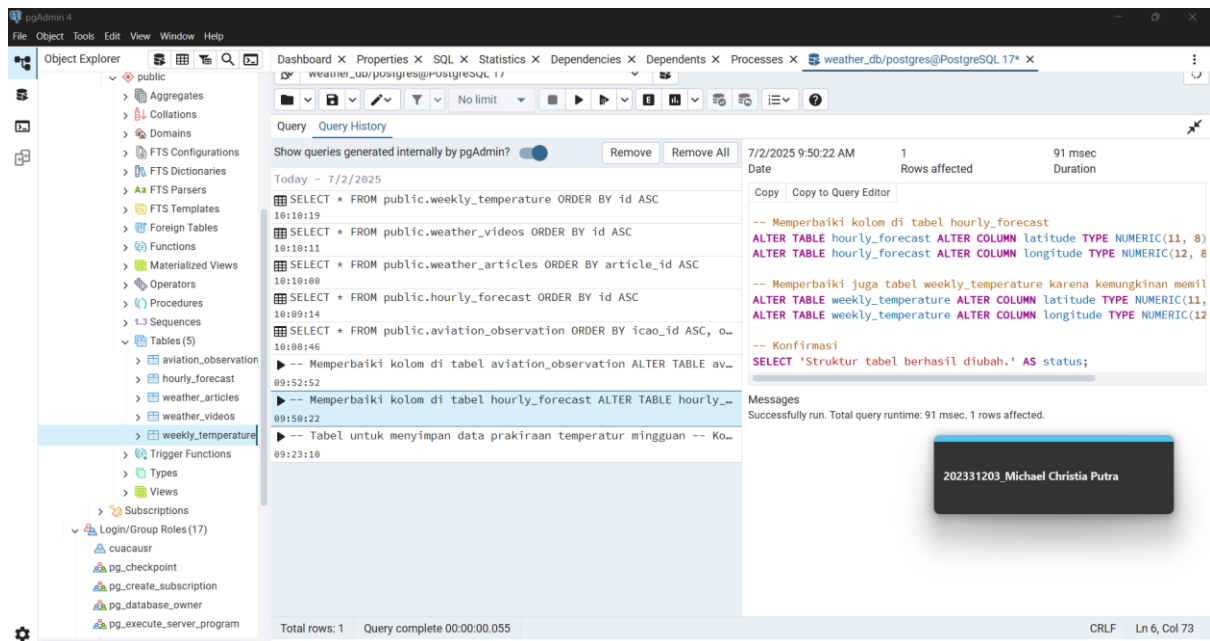
202331203_Michael Christia Putra

Integrasi Database

Integrasi database dalam proyek ini mencakup persiapan skema di PostgreSQL hingga verifikasi data yang telah berhasil dimuat. PostgreSQL dipilih sebagai sistem manajemen database karena keandalannya, dukungannya terhadap tipe data yang kaya, dan kemampuannya menangani kueri yang kompleks. Langkah pertama dalam persiapan database adalah pembuatan tabel. Skema untuk setiap tabel dirancang dengan cermat untuk merepresentasikan data yang diekstrak dari API. Tipe data kolom dipilih sesuai dengan sifat datanya, misalnya `TIMESTAMP` untuk data waktu, `VARCHAR` untuk data teks, `INTEGER` untuk bilangan bulat, dan `NUMERIC` dengan presisi dan skala tertentu untuk data desimal seperti koordinat. Batasan-batasan seperti `PRIMARY KEY` dan `UNIQUE` juga didefinisikan untuk menjaga integritas dan keunikan data. Perintah-perintah `CREATE TABLE` ini dieksekusi secara otomatis oleh skrip Python pada Sel 3.

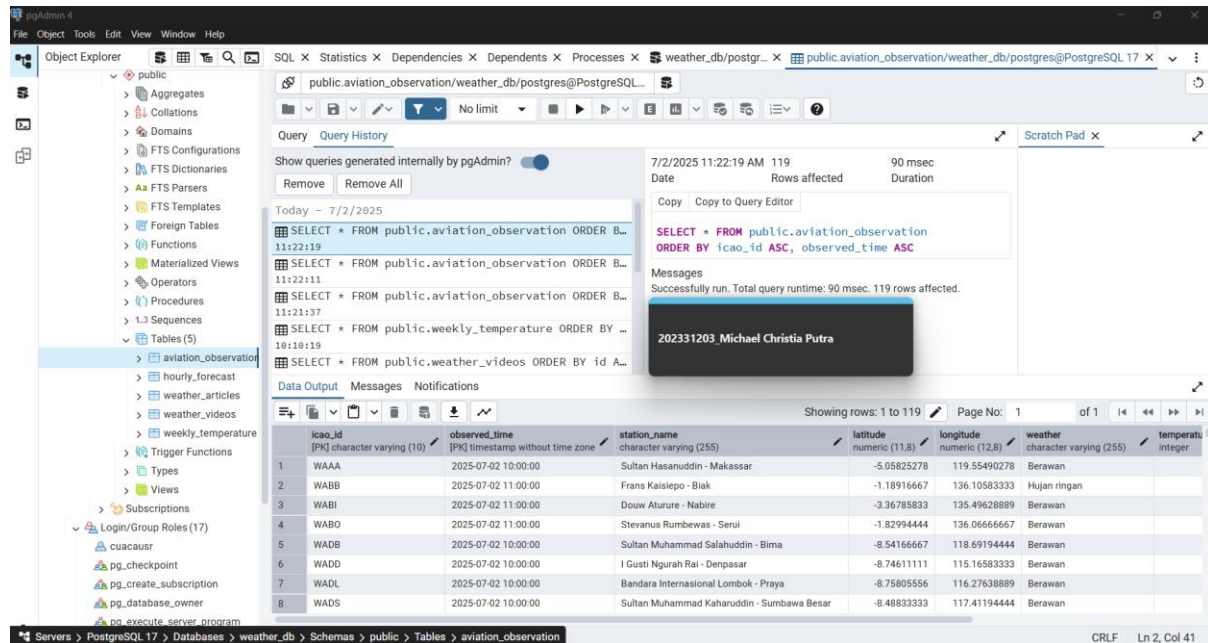


Selama proses pengembangan, ditemukan bahwa definisi awal untuk kolom koordinat (NUMERIC(10,8)) tidak memadai dan menyebabkan error numeric field overflow. Masalah ini diatasi dengan memodifikasi skema tabel yang sudah ada menggunakan perintah ALTER TABLE. Kolom latitude dan longitude diubah menjadi NUMERIC(11,8) dan NUMERIC(12,8), yang memberikan ruang lebih untuk digit di depan koma. Proses *debugging* dan modifikasi skema ini merupakan bagian realistis dari siklus pengembangan proyek data.



Setelah seluruh proses ETL berhasil dijalankan, langkah terakhir adalah melakukan verifikasi untuk memastikan data telah tersimpan dengan benar di dalam database. Verifikasi ini dilakukan dengan dua cara utama: pertama, dengan menggunakan fitur "View Data" pada tool GUI seperti pgAdmin yang menampilkan isi tabel secara visual. Kedua, dengan menjalankan

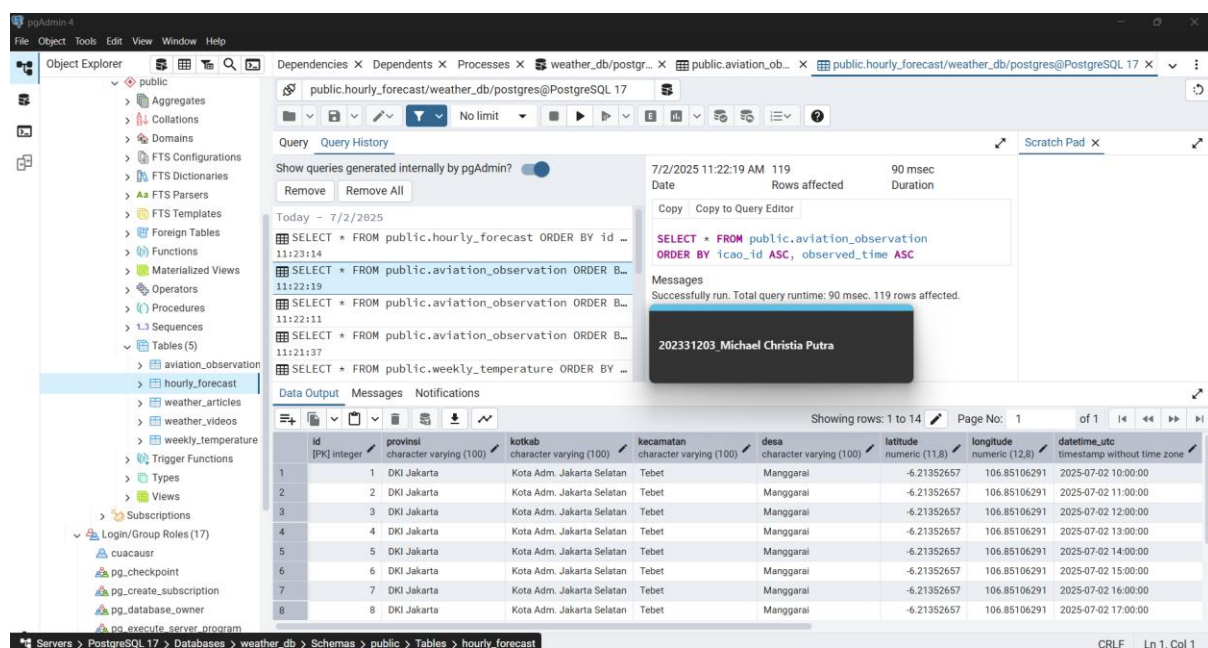
kueri `SELECT * FROM nama_tabel` langsung dari dalam Jupyter Notebook menggunakan fungsi `pandas.read_sql_query()`. Hasil dari kueri ini ditampilkan sebagai `DataFrame`, mengonfirmasi bahwa data telah berhasil diekstrak dari sumber, ditransformasi sesuai logika, dan dimuat ke dalam destinasi database dengan sukses.



Query: `SELECT * FROM public.aviation_observation ORDER BY icao_id ASC, observed_time ASC`

Messages: Successfully run. Total query runtime: 90 msec. 119 rows affected.

icao_id	observed_time	station_name	latitude	longitude	weather	temperatu
WAAA	2025-07-02 10:00:00	Sultan Hasanuddin - Makassar	-5.05825278	119.55490278	Berawan	
WABB	2025-07-02 11:00:00	Frans Kaisiepo - Biak	-1.18916667	136.10583333	Hujan ringan	
WABI	2025-07-02 11:00:00	Douw Aturure - Nabire	-3.36785833	135.49628889	Berawan	
WABO	2025-07-02 11:00:00	Stevanus Rumbewas - Serui	-1.82994444	136.06666667	Berawan	
WADB	2025-07-02 10:00:00	Sultan Muhammad Salahuddin - Bima	-8.54166667	118.69194444	Berawan	
WADL	2025-07-02 10:00:00	I Gusti Ngurah Rai - Denpasar	-8.74611111	115.16583333	Berawan	
WADS	2025-07-02 10:00:00	Bandara Internasional Lombok - Praya	-8.75805556	116.27638889	Berawan	
		Sultan Muhammad Kaharuddin - Sumbawa Besar	-8.48833333	117.41194444	Berawan	



Query: `SELECT * FROM public.hourly_forecast ORDER BY id ...`

Messages: Successfully run. Total query runtime: 90 msec. 119 rows affected.

id	provinsi	kotab	kecamatan	desa	latitude	longitude	datetime_utc
1	DKI Jakarta	Kota Adm. Jakarta Selatan	Tebet	Manggarai	-6.21352657	106.85106291	2025-07-02 10:00:00
2	DKI Jakarta	Kota Adm. Jakarta Selatan	Tebet	Manggarai	-6.21352657	106.85106291	2025-07-02 11:00:00
3	DKI Jakarta	Kota Adm. Jakarta Selatan	Tebet	Manggarai	-6.21352657	106.85106291	2025-07-02 12:00:00
4	DKI Jakarta	Kota Adm. Jakarta Selatan	Tebet	Manggarai	-6.21352657	106.85106291	2025-07-02 13:00:00
5	DKI Jakarta	Kota Adm. Jakarta Selatan	Tebet	Manggarai	-6.21352657	106.85106291	2025-07-02 14:00:00
6	DKI Jakarta	Kota Adm. Jakarta Selatan	Tebet	Manggarai	-6.21352657	106.85106291	2025-07-02 15:00:00
7	DKI Jakarta	Kota Adm. Jakarta Selatan	Tebet	Manggarai	-6.21352657	106.85106291	2025-07-02 16:00:00
8	DKI Jakarta	Kota Adm. Jakarta Selatan	Tebet	Manggarai	-6.21352657	106.85106291	2025-07-02 17:00:00

pgAdmin 4

Object Explorer

- public
 - Aggregates
 - Collations
 - Domains
 - FTS Configurations
 - FTS Dictionaries
 - FTS Parsers
 - FTS Templates
 - Foreign Tables
 - Functions
 - Materialized Views
 - Operators
 - Procedures
 - Sequences
 - Tables (5)
 - aviation_observation
 - hourly_forecast
 - weather_articles
 - weather_videos**
 - weekly_temperature
 - Trigger Functions
 - Types
 - Views
 - Subscriptions
- Login/Group Roles (17)
 - cuacausr
 - pg_checkpoint
 - pg_create_subscription
 - pg_database_owner
 - pg_execute_server_program

Query

public.weather_videos/weather_db/postgres@PostgreSQL 17

Query History

Show queries generated internally by pgAdmin? ☒

Remove Remove All

Today - 7/2/2025

```
SELECT * FROM public.weather_videos ORDER BY id ASC
11:23:43
SELECT * FROM public.hourly_forecast ORDER BY id ASC
11:23:14
SELECT * FROM public.aviation_observation ORDER BY id ASC
11:22:19
SELECT * FROM public.aviation_observation ORDER BY id ASC
11:22:11
SELECT * FROM public.aviation_observation ORDER BY id ASC
11:22:11
```

Messages

Successfully run. Total query runtime: 151 msec. 14 rows affected.

202331203_Michael Christia Putra

Data Output

Showing rows: 1 to 6 Page No: 1 of 1

id	title	video_url	fetch_at
[PK] integer	character varying (255)	character varying (255)	timestamp with time zone
1	Prakiraan Cuaca Esok Hari, Rabu, 02 Juli 2025.	https://www.youtube.com/watch?v=PjHsUK8LpRQ	2025-07-02 09:57:20.889871+07
2	Prakiraan Cuaca Esok Hari, Selasa, 01 Juli 2025	https://www.youtube.com/watch?v=IX5FFHpwavo	2025-07-02 09:57:20.889871+07
3	Prakiraan Cuaca Esok Hari, Senin, 30 Juni 2025.	https://www.youtube.com/watch?v=LG83bBt7ajc	2025-07-02 09:57:20.889871+07
4	Prakiraan Cuaca Esok Hari, Minggu, 29 Juni 2025	https://www.youtube.com/watch?v=cD7vWjRVRUc	2025-07-02 09:57:20.889871+07
5	Prakiraan Cuaca Esok Hari Sabtu, 28 Juni 2025	https://www.youtube.com/watch?v=YnDZyWE...	2025-07-02 09:57:20.889871+07
6	Prakiraan Cuaca Esok Hari, Jumat, 27 Juni 2025	https://www.youtube.com/watch?v=g8lvWXSBrA	2025-07-02 09:57:20.889871+07

Servers > PostgreSQL 17 > Databases > weather_db > Schemas > public > Tables > weather_videos

CRLF Ln 1, Col 1

pgAdmin 4

Object Explorer

- public
 - Aggregates
 - Collations
 - Domains
 - FTS Configurations
 - FTS Dictionaries
 - FTS Parsers
 - FTS Templates
 - Foreign Tables
 - Functions
 - Materialized Views
 - Operators
 - Procedures
 - Sequences
 - Tables (5)
 - aviation_observation
 - hourly_forecast
 - weather_articles
 - weather_videos
 - weekly_temperature**
 - Trigger Functions
 - Types
 - Views
 - Subscriptions
- Login/Group Roles (17)
 - cuacausr
 - pg_checkpoint
 - pg_create_subscription
 - pg_database_owner
 - pg_execute_server_program

Query

public.weekly_temperature/weather_db/postgres@PostgreSQL 17

Query History

Show queries generated internally by pgAdmin? ☒

Remove Remove All

Today - 7/2/2025

```
SELECT * FROM public.weekly_temperature ORDER BY id ASC
11:24:08
SELECT * FROM public.weather_videos ORDER BY id ASC
11:23:43
SELECT * FROM public.hourly_forecast ORDER BY id ASC
11:23:14
SELECT * FROM public.aviation_observation ORDER BY id ASC
11:22:19
SELECT * FROM public.aviation_observation ORDER BY id ASC
11:22:11
```

Messages

Successfully run. Total query runtime: 114 msec. 6 rows affected.

202331203_Michael Christia Putra

Data Output

Showing rows: 1 to 7 Page No: 1 of 1

id	forecast_date	temperature_celsius	latitude	longitude	fetch_at
[PK] integer	date	numeric (10,8)	numeric (11,8)	numeric (12,8)	timestamp with time zone
1	2025-07-02	26.08880324	-6.21510000	106.84610000	2025-07-02 09:43:43.519602+07
2	2025-07-03	25.91848848	-6.21510000	106.84610000	2025-07-02 09:43:43.519602+07
3	2025-07-04	25.34496226	-6.21510000	106.84610000	2025-07-02 09:43:43.519602+07
4	2025-07-05	25.56966979	-6.21510000	106.84610000	2025-07-02 09:43:43.519602+07
5	2025-07-06	25.29060005	-6.21510000	106.84610000	2025-07-02 09:43:43.519602+07
6	2025-07-07	26.91471577	-6.21510000	106.84610000	2025-07-02 09:43:43.519602+07
7	2025-07-08	26.15428573	-6.21510000	106.84610000	2025-07-02 09:43:43.519602+07

Total rows: 7 Query complete 00:00:00.149

CRLF Ln 1, Col 1