# Import necessary libraries

import os

import psycopg2

from sodapy import Socrata

from psycopg2.extras import execute\_values

from dotenv import load\_dotenv

from datetime import datetime

# Step 1: Set up environment variables (API Token and DB credentials)

# Set your environment variables manually in Google Colab

os.environ["APP\_TOKEN"] = "your\_socrata\_app\_token" # Replace with your Socrata App Token

os.environ["DB\_NAME"] = "your\_db\_name" # Replace with your PostgreSQL DB name

os.environ["DB\_USER"] = "your\_db\_user" # Replace with your PostgreSQL username

os.environ["DB\_PASS"] = "your\_db\_password" # Replace with your PostgreSQL password

os.environ["DB\_HOST"] = "localhost" # Replace with your PostgreSQL host (e.g., localhost or cloud DB IP)

# Step 2: Fetch data from the API

def fetch\_data(limit=1000):

try:

client = Socrata("data.melbourne.vic.gov.au", os.getenv("APP\_TOKEN"))

# Fetch data from Socrata API with the provided limit

results = client.get("b2ak-trbp", limit=limit) # Dataset ID: b2ak-trbp

if not results:

print("No data found.")

return []

return results

except Exception as e:

print(f"Error fetching data: {e}")

return []

# Step 3: Database Connection Setup

def get\_connection():

return psycopg2.connect(

dbname=os.getenv("DB\_NAME"),

user=os.getenv("DB\_USER"),

password=os.getenv("DB\_PASS"),

host=os.getenv("DB\_HOST"),

port=5432

)

# Step 4: Load the fetched data into PostgreSQL

def load\_data(records):

formatted\_data = []

for r in records:

# Extract relevant fields from the API response

formatted\_data.append((

r.get("sensor\_id"), # Replace with the correct field if necessary

float(r.get("temperature", 0)), # Default to 0 if temperature is missing

float(r.get("humidity", 0)), # Default to 0 if humidity is missing

r.get("timestamp"),

f"POINT({r['location']['longitude']} {r['location']['latitude']})"

))

insert\_query = """

INSERT INTO weather\_data (sensor\_id, temperature, humidity, timestamp, location)

VALUES %s

ON CONFLICT (sensor\_id, timestamp) DO NOTHING;

"""

with get\_connection() as conn:

with conn.cursor() as cur:

# Insert data into the PostgreSQL database using `execute\_values` for bulk inserts

execute\_values(cur, insert\_query, formatted\_data)

print("Data inserted successfully!")

# Step 5: Run unit tests for fetching data (optional)

def test\_fetch\_data():

data = fetch\_data(limit=10)

assert isinstance(data, list)

assert len(data) <= 10

print(f"Fetched {len(data)} records.")

# Step 6: End-to-End Test for data fetching and loading

def end\_to\_end\_test():

print("Fetching data...")

data = fetch\_data(limit=10) # Fetch 10 records

print(f"Fetched {len(data)} records.")

if data:

print("Loading data into the database...")

load\_data(data)

print("Data loading complete!")

else:

print("No data to load.")

# Run tests and load data

end\_to\_end\_test()