Airline-focused AI, API, Emission Example Project

王馬丁

25/08

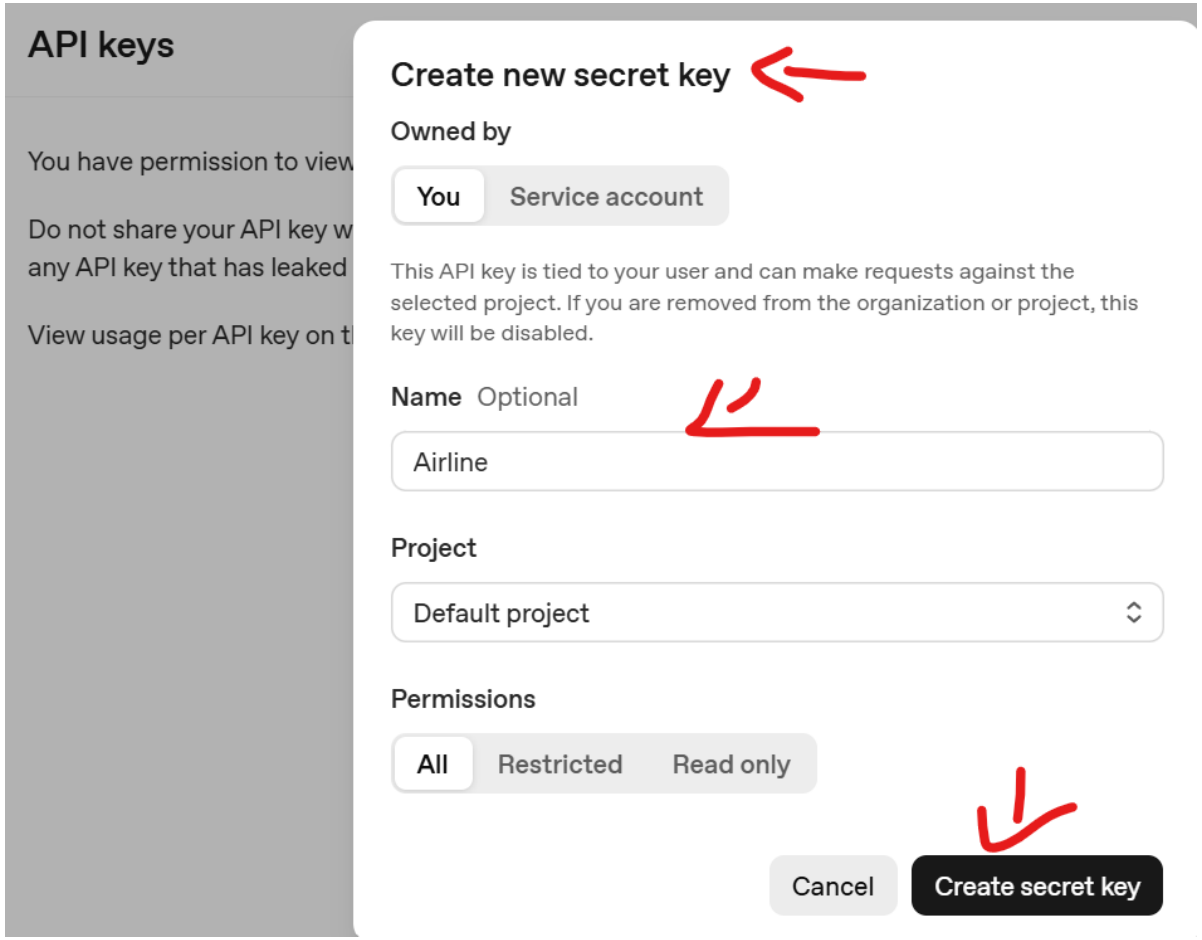
Plan

1. Generating AI Dataset with **OpenAI API** and **Python**.
2. Cybersecurity Scan with **Microsoft Defender**.
3. Analyze and manage data using **MS SQL.**
4. Result visualization with **Tableau**.

Concept

1. AI-synthetic Airline dataset based on Taiwanese Carriers via API.
2. Validated for file safety to ensure defense against cyber threats.
3. Benchmark Carrier Efficiency using CO_2 emission and energy usage, to outline environmental considerations.

1A Create API Key on OpenAI Developer



Important to save the OpenAI API Key securely, e.g. it should not leak publicly for others to use.

Add to credit balance

Make sure there is enough credit for the project in-hand to complete.

1B Using Visual Code to write Python code

generate_dataset.py

Create .py file which will allow to access OpenAI API and generated synthetic dataset.

Opted for GPT-4o-mini version, which is one of the most cost-saving options.

The API-related cost of this project is only 0.01 USD.

1C Adding Key, Ensuring OpenAI is installed

```
import os
import sys
import json
import csv
import subprocess
import random
from pathlib import Path

API_KEY = "sk-"
```

Adding API-key generated on OpenAI Developer.

```
# Ensure openai is installed for this interpreter
try:
    from openai import OpenAI
except ModuleNotFoundError:
    subprocess.check_call([sys.executable, "-m", "pip", "install", "openai"])
    from openai import OpenAI
```

Make sure OpenAI module is installed for VS Code.

1D Writing a clear AI Prompt for Data Creation

```
def build_prompt():
    return f"""
Return ONLY valid JSON as:
{{"rows":[{{"airline": str, "route": "domestic"|"international",
"flight_hours": number, "energy_consumption_mwh": number,
"passenger_load": integer, "co2_emissions_tonnes": number}} ...]}}

Rules:
- Try to output {TARGET_ROWS} objects in "rows".
- "airline" ∈ {AIRLINES}. Tigerair Taiwan must be international only.
- "route": domestic is rare; most are international.
- "flight_hours": domestic ~0.5–1.5; international ~2–6 (regional) or 8–13 (long-haul).
- "energy_consumption_mwh" ~ proportional to flight_hours.
- "passenger_load": domestic 80–160; regional 120–260; long-haul 220–360.
- "co2_emissions_tonnes" correlates with energy_consumption_mwh.
- Numbers must be numeric (not strings). No explanations—JSON only.
"""
```

Being very clear and precise. Excluding any unnecessary context.

```
def call_api():
    client = OpenAI(api_key=require_key())
    resp = client.chat.completions.create(
        model=MODEL,
        response_format={"type": "json_object"},
        temperature=0.2,
        messages=[
            {"role": "system", "content": "You are a strict JSON generator for data analysis demos."},
            {"role": "user", "content": build_prompt()},
        ],
    )
    return resp.choices[0].message.content
```

Calling OpenAI API and prompting it to be a strict JSON generator.

1E Forcing output of 100 rows

```python
def parse_and_pad(text):
    data = json.loads(text)
    rows = data.get("rows", [])
    if not isinstance(rows, list):
        raise ValueError("Model did not return JSON with a 'rows' array.")
    if len(rows) == 0:
        raise ValueError("Model returned zero rows.")
    # If fewer than TARGET_ROWS, duplicate random rows until length is met
    while len(rows) < TARGET_ROWS:
        rows.append(random.choice(rows))
    # If more, truncate
    rows = rows[:TARGET_ROWS]
```

Most OpenAI models are optimized to minimize risk of error and stay within token budgets. It may result in undershooting. Thus, enforcing right output via Python is an essential step.

Making sure that the JSON schema makes the AI return exactly the number of rows we ask for, with the right columns and valid values.

1F Calling GPT to Output final CSV dataset

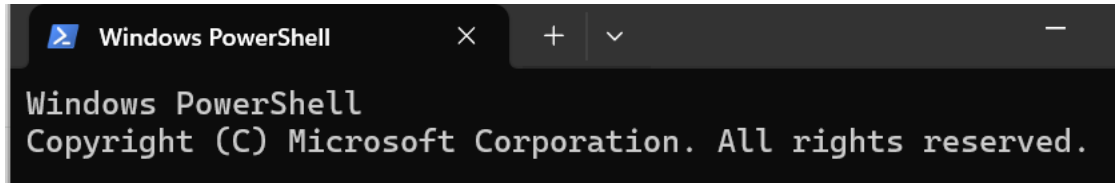
```
/Documents/python/Airline/generate_dataset.py
Calling gpt-4o-mini… aiming for 100 rows
```

May take couple of seconds to a minute to call GPT-4o-mini and generate the dataset.

	A	B	C	D	E	F
1	airline	route	flight_hou	energy_co	passenger	co2_emissions_tonnes
2	EVA Air	international	8.5	85	300	60
3	Starlux	international	6	60	250	42
4	China Air	international	10	100	350	70
5	Tigerair T	international	3	30	200	20
6	EVA Air	international	9	90	320	63
7	Starlux	international	5.5	55	240	38
8	China Air	international	12	120	360	80
9	Tigerair T	international	4	40	180	25
10	EVA Air	international	7	70	280	50
11	Starlux	international	2.5	25	150	15
12	China Air	international	11	110	340	75
13	Tigerair T	international	3.5	35	190	22
14	EVA Air	domestic	1	10	120	8

Complete dataset in CSV format, with exactly 100 rows.

2A Scan CSV with Microsoft Defender

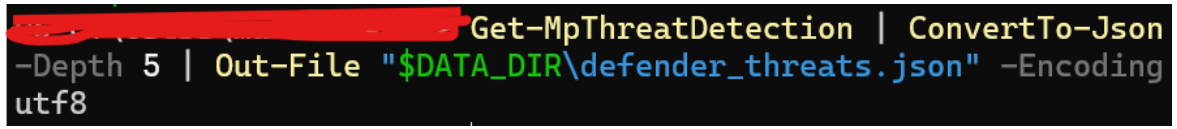



```
$DATA_DIR  = Split-Path $DATA
```

Opening and Using Windows PowerShell.
Then setting data path.

Running a custom scan on just the CSV.

Start-MpScan is Defender's built-in cmdlet; it runs the Defender engine.



Exporting detections as JSON.

2B Sanity Check JSON for Threats

This command queries Defender's Detection List. The Count result is 0.

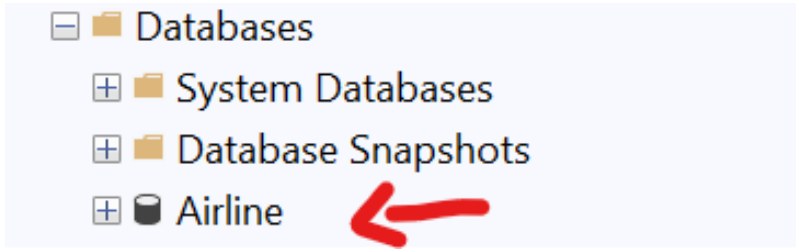
Therefore, nothing was detected.

Data security is essential because even simple CSV files can carry hidden threats if sourced online.

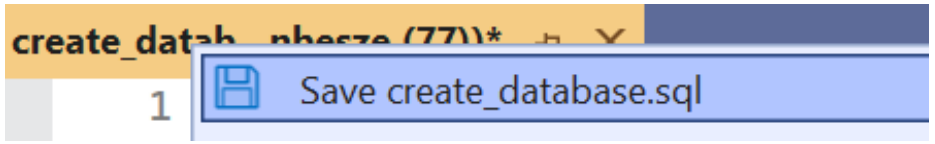
Running a scan with Microsoft Defender ensures that AI-generated or downloaded datasets are clean before storing

3A Creating Database in MS SQL via SSMS

```
1   USE master;
2   GO
3   CREATE DATABASE [Airline];
4   GO
```

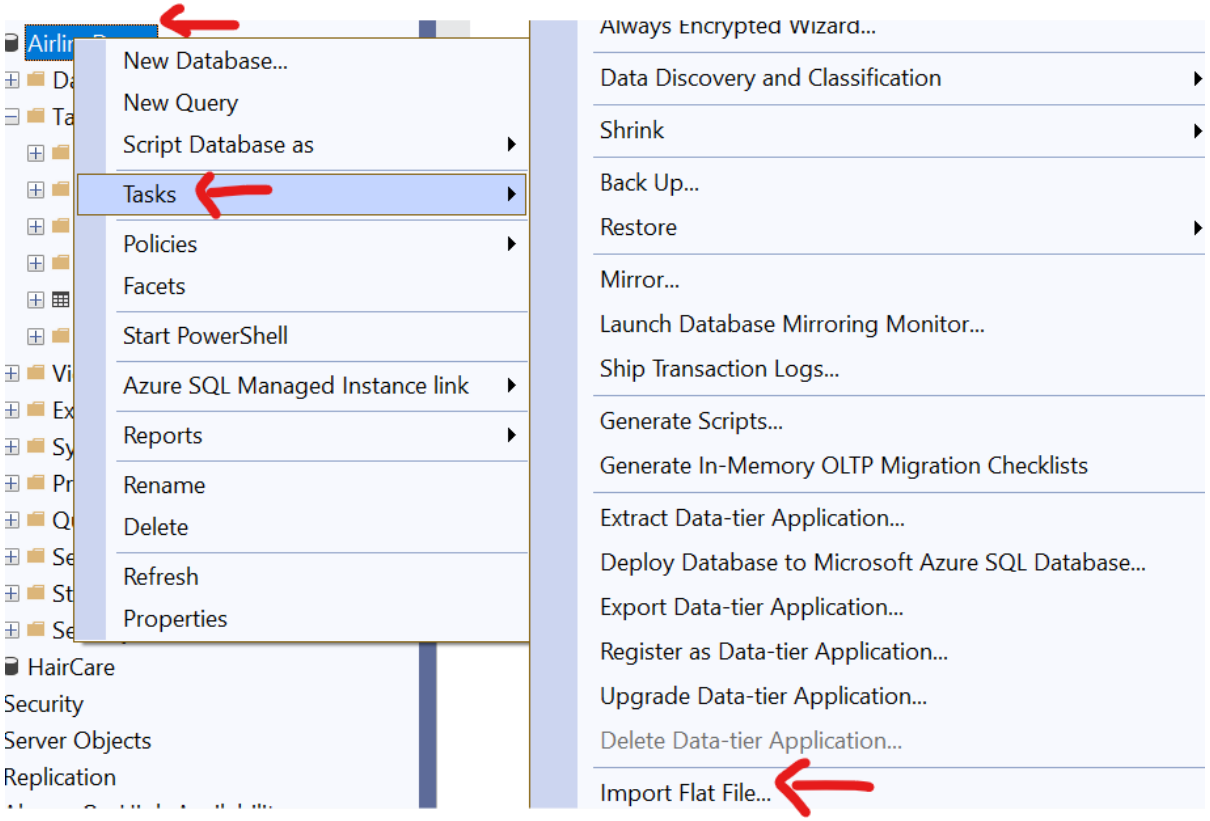


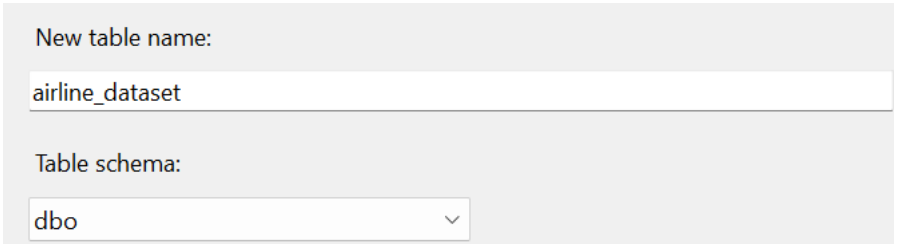
Creating Airline Database for use.

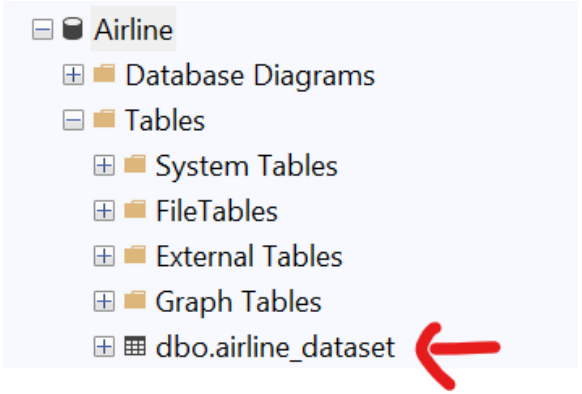


Saving SQL query as .sql file to our device for future use.

3B Import Secure CSV and create table







airline	route	flight_hours	energy_consumption_mwh	passenger_load	co2_emissions_tonnes
EVA Air	international	8.5	85	300	60
Starlux	international	6	60	250	42
China Airlines	international	10	100	350	70

3C Create Primary Key flight_id

```
USE Airline;
GO
ALTER TABLE dbo.airline_dataset
ADD flight_id INT IDENTITY(1,1) PRIMARY KEY;
```

Adding new column, which is auto-numbering rows starting from 1.

Then it is defined as PK.

sults | Messages

airline	route	flight_hours	energy_consumption_mwh	passenger_load	co2_emissions_tonnes	flight_id
EVA Air	international	8.5	85	300	60	1
Starlux	international	6	60	250	42	2
China Airlines	international	10	100	350	70	3
Tigerair Taiwan	international	3	30	200	20	4
EVA Air	international	9	90	320	63	5
Starlux	international	5.5	55	240	38	6

3D Passenger Load by Airline

```sql
USE Airline;
SELECT
    airline,
    SUM(passenger_load) AS total_passengers
FROM dbo.airline_dataset
GROUP BY airline
ORDER BY total_passengers DESC;
```

	airline	total_passengers
1	EVA Air	8170
2	China Airlines	7880
3	Starlux	5050
4	Tigerair Taiwan	4650

Gives a ranking of airlines by passenger volume, using GROUP BY.

3E CO2 Emission per Passenger

```sql
USE Airline;
SELECT
    airline,
    SUM(co2_emissions_tonnes) AS total_co2,
    SUM(passenger_load) AS total_passengers,
    CAST(SUM(co2_emissions_tonnes) * 1.0 / NULLIF(SUM(passenger_load),0) AS DECIMAL(10,4))
        AS co2_per_passenger
FROM dbo.airline_dataset
GROUP BY airline
ORDER BY co2_per_passenger ASC;
```

	airline	total_co2	total_passengers	co2_per_passenger
1	Tigerair Taiwan	473	4650	0.1017
2	Starlux	732	5050	0.1450
3	EVA Air	1486	8170	0.1819
4	China Airlines	1630	7880	0.2069

Allows to evaluate which airline excels in reducing CO2 emissions.

Using CAST and NULLIF.

3F Emission compared to global average

```sql
WITH airline_stats AS (
    SELECT
        airline,
        SUM(passenger_load) AS total_passengers,
        SUM(co2_emissions_tonnes) AS total_co2,
        CAST(SUM(co2_emissions_tonnes) * 1.0 / NULLIF(SUM(passenger_load),0) AS DECIMAL(10,4)) AS co2_per_passenger
    FROM dbo.airline_dataset
    GROUP BY airline
)
SELECT
    airline,
    total_passengers,
    total_co2,
    co2_per_passenger,
    RANK() OVER (ORDER BY co2_per_passenger ASC) AS efficiency_rank,
    AVG(co2_per_passenger) OVER () AS global_avg_co2_per_passenger,
    co2_per_passenger - AVG(co2_per_passenger) OVER () AS diff_from_avg
FROM airline_stats
ORDER BY efficiency_rank;
```

AVG() OVER() calculates the global average once, without re-joining.

airline	total_passengers	total_co2	co2_per_passenger	efficiency_rank	global_avg_co2_per_passenger	diff_from_avg
Tigerair Taiwan	4650	473	0.1017	1	0.158875	-0.057175
Starlux	5050	732	0.1450	2	0.158875	-0.013875
EVA Air	8170	1486	0.1819	3	0.158875	0.023025
China Airlines	7880	1630	0.2069	4	0.158875	0.048025

4A Import CSV to Tableau Public

airline_dataset.csv
100 rows 6 fields

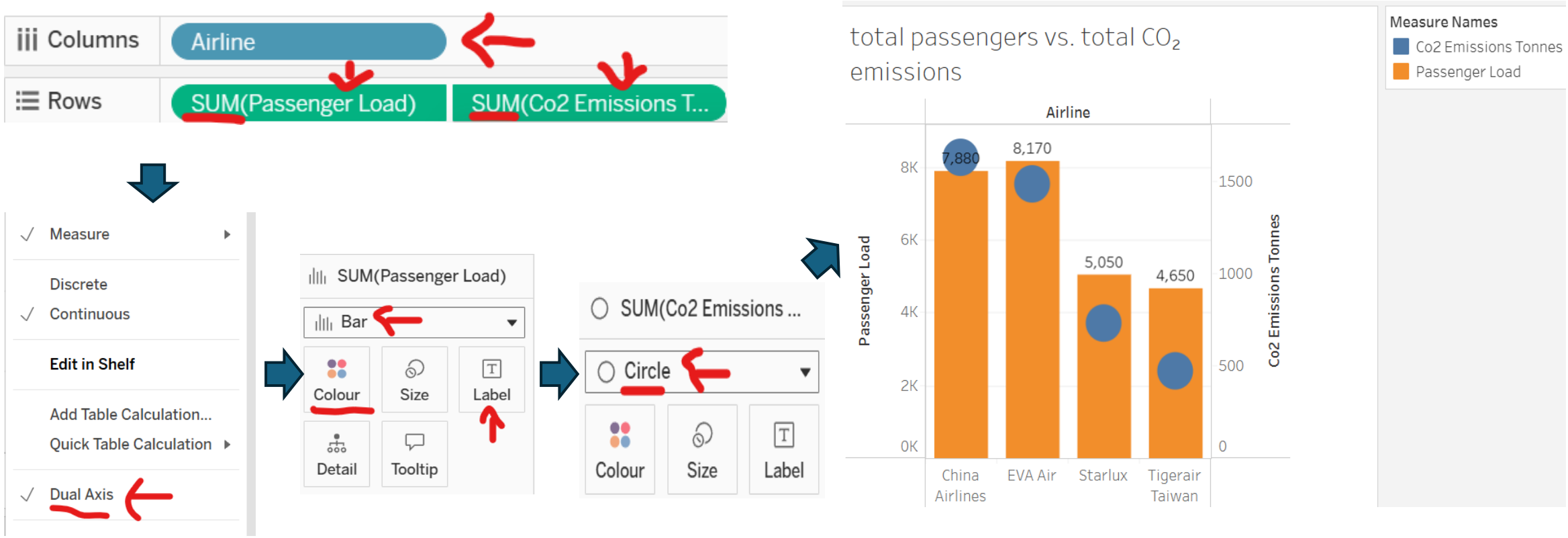
airline_dataset.c...

airline airline_dataset.csv	route airline_dataset.csv	co2_emissions_tonnes airline_dataset.csv	energy_consumption_mwh airline_dataset.csv	flight_hours airline_dataset.csv	passenger_load airline_dataset.csv
EVA Air	international	60.0000	85.000	8.5000	300
Starlux	international	42.0000	60.000	6.0000	250
China Airlines	international	70.0000	100.000	10.0000	350
Tigerair Taiwan	international	20.0000	30.000	3.0000	200
EVA Air	international	63.0000	90.000	9.0000	320

Tableau handles complex queries visually, makes comparisons intuitive, and allows clear, interactive storytelling.

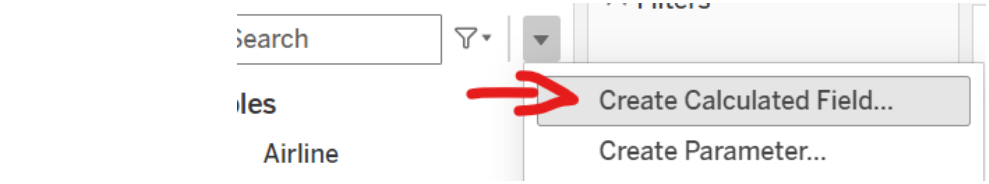
Using Tableau Public (Free version).

4B Dual Axis Chart to showcase CO2 emission



Bars showing total passengers, with circle showing CO_2 emissions.

4C Efficiency Quadrants: Calculated Fields

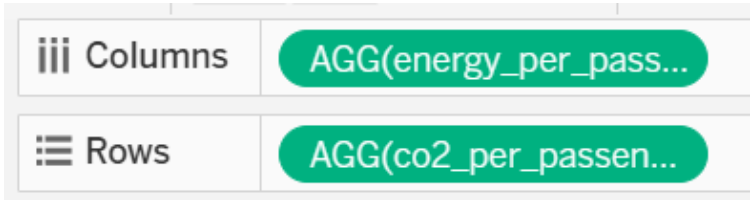


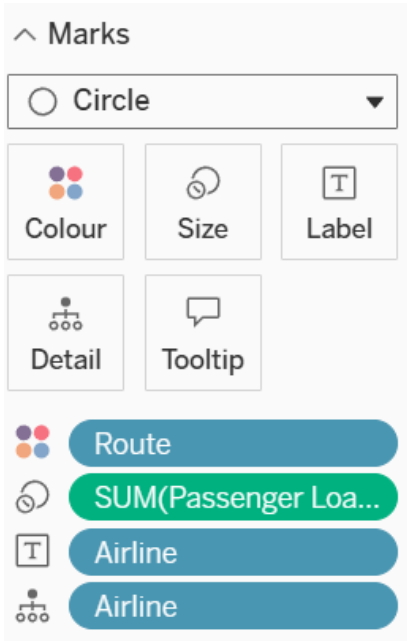
co2_per_passenger

```
[co2_emissions_tonnes]) / NULLIF(SUM([passenger_load]),0)
```

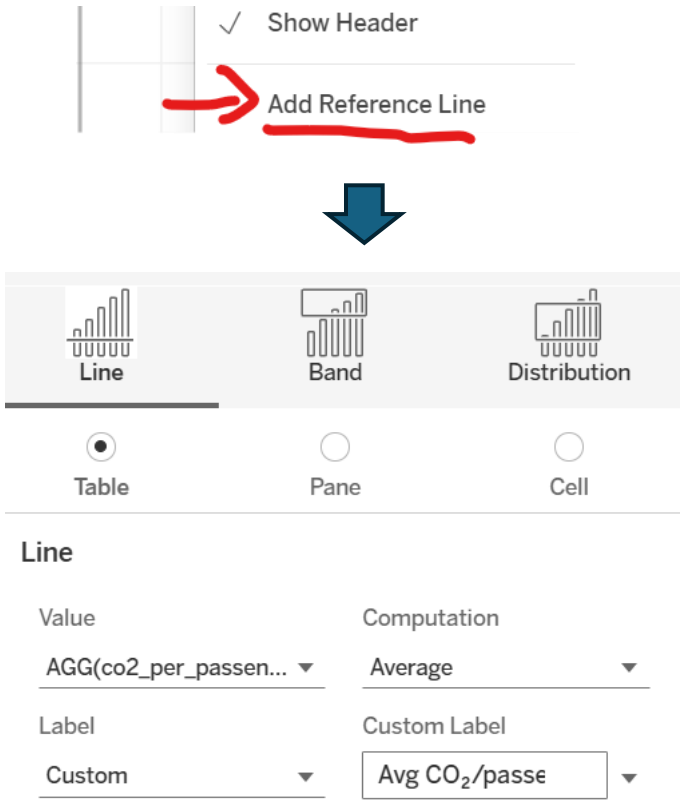
energy_per_passenger

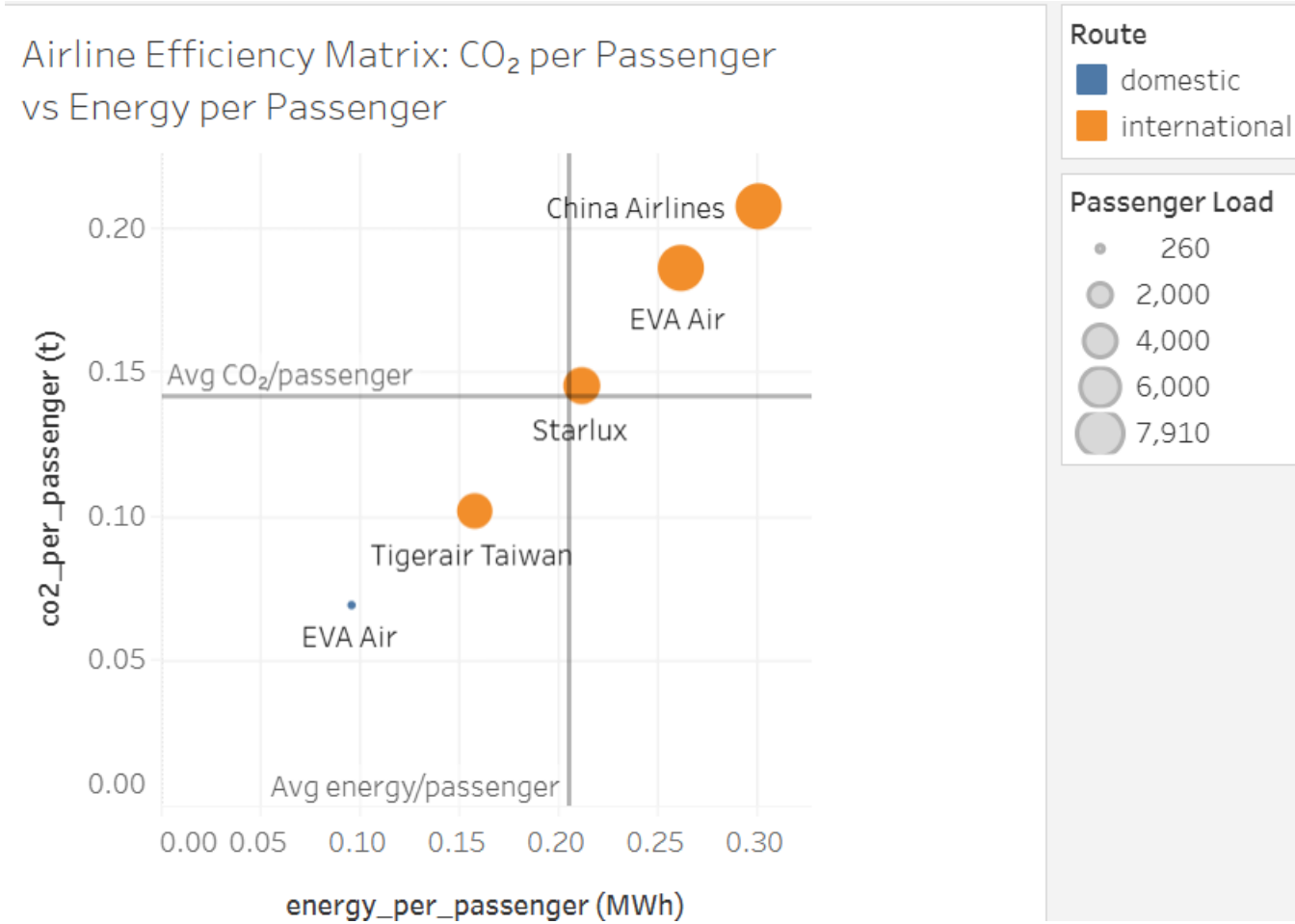
```
SUM([Energy Consumption Mwh]) / NULLIF(SUM([Passenger Loa
```





4D Efficiency Quadrants: Reference Lines





謝謝!