Initial Notes

Notes of what I need to do

Interest → understand the broader trends in the automotive industry.

In this task, I'm **going to create a data pipeline** to **acquire, process, and load data from various and public sources.** Basically ETL process.

I have 5 tasks ahead:

- **Data Acquisition**: First step is to **write scripts** that downloads the datasets from the provided public data sources:
    - Use the following data sources for this task:
        1. U.S. Department of Transportation - National Highway Traffic Safety Administration: Vehicle Complaints (link: https://www.nhtsa.gov/nhtsa-datasets-and-apis)
        2. U.S. Department of Energy: Alternative Fuel Stations (link: https://afdc.energy.gov/data_download)
        3. U.S. Environmental Protection Agency: Vehicle Fuel Economy Information (link: https://fueleconomy.gov/feg/ws/index.shtml)
- **Data Processing**: Clean and integrate these datasets. This should include, but not be limite to, handling missing values, duplicates and possible outliers. Basically the **Silver Stage**.
- **Data Transformation**: Transform the data into a format suitable for further analysis. Justify the choices you make during this process.
- **Data Loading**: Write a script to load the data into a hypothetical data storage system. While I can't actually load the data into Azure SQL Database or Databricks Delta Lake, you should simulate the process and include the relevant commands in your script.
- **Automation Suggestion**: Describe how you would automate this pipeline with a schedule interval you would choose and explain why.

Overall Goal

The overall goal is to create a repeatable pipeline (w/ scripts) that pulls data from US government sources, cleans and transforms it, loads it (simulation) and automates the process (suggestion (?)). Then, document everything in a presentation shared by GitHub. Simulating a real data engineering task.

Overview and Setup Context

- I'm looking for trends like fuel economy (efficiency standards), vehicle safety and alternative fuels. I'm also building a pipeline to ingest this data automatically, process it for analysis and store it hypothetically.
- **Constraints/"Agreements"**: Will use data provided, scripts will be in python (because it's more flexible). Advanced parts will be simulated (e.g. no real Azure or Databricks Access). Focus on handling common data engineering issues like incompleteness or inconsistencies.
- **What to submit into GitHub**: All scripts, processed data samples and a PowerPoint presentation.

Objective and Tasks: Step-by-Step

There 5 primordial objectives:

1. **Data Acquisition**
    a. By writing scripts (in Python in this case), I will download datasets from three sources. Include basic handling for issues like missing values, duplicates and outliers. **IMPORTANT (?):** The script should make downloads repeatable (e.g., via URLS/APIs).
    b. Will identify and address data quality issues early, like NaNs after the download.
    c. Will download into a data/raw folder to have the raw data save and log errors. Will make the project robust.
2. **Data Transformation**

a. Will transform the raw data into a suitable format for further analysis. Cleaning (fix formats, merge related data), standardizing (consistent units like MPG) and possibly aggregating (averages by vehicle type, for example). Here, every choice needs to be documented (things like “Dropped outliers > 3 SD because they skew trends”, for example).

b. The primordial objective here is to making messy data usable. Will need to decide on quality (impute missing values or drop them, for example) and why they’re appropriate for automative analysis (accurate fuel trends, for example).

c. Will build on acquired data (like merging epa fuel data with nhtsa safety by vehicle model or year, handle issues (like incomplete safety ratings), duplicates or outliers, clean csvs/parquets and perhaps create a unified dataset for querying (?) like this I can test creatively my work and efficiently.

3. **Data Loading**

   a. I want to write a script to “load” the transformed data into a hypothetical storage like Azure SQL Database or Databricks Delta Lake. Simulating it by saving to local files in a lake-like format for example. Including schema design (table structures) and error handling).

   b. Will show in this step how I’d integrate with cloud tools IRL.

   c. Maybe use pandas to save as Parquet (Delta Lake sim) or insert into local DB. Loading is important because we ensure data is queryable and scalable. Things like analytics dashboards.

   d. In azure, I’d use ADF (?)

4. **Automation**

   a. Describe (maybe script) how to automate the full pipeline on a schedule (e.g., daily) and explain why (weekly because data updates monthly, example).

   b. Making it ready for production

   c. Python schedulers or Azure functions (simulated (?))