Delivery Delay Pipeline

> *So, you're halfway through your training? It's the perfect time to challenge yourself with a real-world scenario closely tied to your daily professional life!*

Context

At SuperCourrier, our Data Science team is building a predictive model to anticipate delivery delays.

Issue Problem: They need a clean, structured dataset... by tomorrow!

As a Data Engineer, your task is to create a simple yet effective pipeline to generate a ready-to-use dataset by combining and transforming data from various sources.

Your Mission (2 hours max!)

Develop a Python script simulating data extraction, transformation, and loading (ETL) for delivery data.

The final result should be a CSV file immediately usable by Data Scientists.

Data Sources to Simulate

1. Logistics Database (simulated with SQLite):
   - Basic information about deliveries
   - Types of packages and delivery zones
2. Weather Data (generated):
   - Weather conditions by date and time
3. Tracking Logs (generated):
   - Timestamps of delivery events

Tasks to Accomplish

1. Project Structure

- Create a main Python script
- Define functions for each step (extraction, transformation, loading)
- Implement a logging structure

2. Generating Source Data

- Create a SQLite database with a simplified delivery table

- Generate a dictionary of weather conditions (JSON format)
- Simulate tracking logs with start and end timestamps

3. Data Transformation

- Join data from various sources
- Calculate delivery durations
- Enrich with weather information
- Determine if a delivery is delayed based on the provided algorithm
- Handle missing values

4. Export and Validation

- Export the final dataset in CSV format
- Perform basic validations (no outliers)
- Generate simple dataset statistics

5. Documentation and Finalization

- Document the code (docstrings, comments)
- Prepare a brief presentation of the pipeline

Expected Data Structure

The final dataset must include these columns:

- `Delivery_ID` : unique identifier
- `Pickup_DateTime` : date and time of pickup
- `Weekday` : day of the week (Monday, Tuesday, etc.)
- `Hour` : hour of the day (0-23)
- `Package_Type` : category (Small, Medium, Large, Extra Large, Special)
- `Distance` : distance in km (simulated)
- `Delivery_Zone` : type of area (Urban, Suburban, Rural, etc.)
- `Weather_Condition` : weather during delivery
- `Actual_Delivery_Time` : delivery time in minutes
- `Status` : final status (On-time, Delayed)

Delay Calculation Formula

A package is considered delayed if the actual delivery time exceeds a threshold calculated as:

- Base theoretical time = 30 + distance × 0.8 minutes
- Adjustment factors:

    - Package type: small (×1), medium (×1.2), large (×1.5), extra large (×2), special (×2.5)
    - Zone: urban (×1.2), suburban (×1), rural (×1.3), industrial (×0.9), shopping center (×1.4)
    - Weather: sunny (×1), cloudy (×1.05), rainy (×1.2), snowy (×1.8), etc.
    - Peak hours: morning (×1.3), evening (×1.4)
    - Day: Monday/Friday (×1.2), weekend (×0.9)
- Delay threshold = adjusted theoretical time × 1.2

Expected Deliverables

1. Working Python ETL pipeline script
2. Generated CSV dataset file (minimum 1000 deliveries)
3. README briefly explaining how the pipeline works and the code structure

Resources Provided

- Snippet for creating an SQLite database
- Example JSON structure for weather data
- Basic function for calculating theoretical delivery time

Last-minute Advice

- **Simplicity first!** A simple, functioning pipeline is better than an ambitious but incomplete project
- Use pandas for easy data manipulation
- Feel free to make reasonable assumptions if necessary
- Comment your code as you go

Ready? Set... Go! ⏱️