TP 3 – Measuring the Environmental Impact of File Formats (CSV vs Parquet)

Case Study: Books & Reviews Data Processing

---

1. Learning Objectives

At the end of this practical session, you will be able to:

1. Understand how storage formats influence the efficiency and environmental impact of data pipelines.
2. Measure the carbon emissions of reading and writing operations using **CodeCarbon**.
3. Compare **CSV** and **Parquet** formats in terms of execution time, storage footprint, and CO_2 emissions.
4. Design greener ETL workflows by making informed format choices.

---

2. Context

You are a **Data Engineer** in a company that processes millions of book reviews.
Your current workflow stores all datasets in **CSV**, which is simple but inefficient.

Your manager asks you to evaluate whether using **Parquet**, a binary and compressed columnar format, could **reduce both runtime and energy consumption** in your pipelines.

You will therefore reproduce the same *Books & Reviews* data-processing pipeline from **TP2**, but compare the performance and environmental cost of **CSV vs Parquet** operations.

---

3. Datasets Overview

You will use the same datasets as in TP2:

books.csv

Column	Description
Title	Book title
Description	Short summary
Authors	List of authors
Publisher	Publishing company
PublishedDate	Date of publication
Categories	Book categories
RatingsCount	Number of ratings

reviews.csv

Column	Description
Id	Book ISBN or unique ID
Title	Title of the book
Price	Price (optional)
User_id	Reviewer ID
profileName	Reviewer name
review/score	Rating (1–5)
review/time	Timestamp
review/summary	Short summary
review/text	Full review content

---

4. Experimental Design

Goal

Compare the environmental performance of two identical data pipelines differing only in the **file format used for storage**.

You will implement the same transformations as in TP2, but executed on two separate workflows:

- **Pipeline A:** operates on CSV files.
- **Pipeline B:** operates on Parquet files.

Pipeline Steps (identical for both formats)

1. **Load** `books` and `reviews` datasets from storage.
2. **Clean** data:
    - Handle missing values.
    - Normalize authors and categories.
3. **Join** datasets on the `Title` column.
4. **Compute metrics:**
    - Average rating per author.
    - Number of reviews per publisher.
    - Top 10 most-reviewed categories.
5. **Text processing:** compute average review length and most frequent keywords.
6. **Save** results to disk in the corresponding format (CSV or Parquet).

---

5. Tasks

Task 1 — CSV Baseline

1. Implement the complete data pipeline using **Pandas**.

Instrument the code with **CodeCarbon** to measure runtime and CO_2 emissions:

```
 from codecarbon import EmissionsTracker
import pandas as pd

tracker = EmissionsTracker(project_name="csv_pipeline")
tracker.start()

books = pd.read_csv("books.csv")
reviews = pd.read_csv("reviews.csv")

df = (
    reviews.merge(books, on="Title", how="inner")
)
df.to_csv("merged_books_reviews.csv", index=False)

tracker.stop()
```

2. Export the emission logs as `emissions_csv.csv`.

Task 2 — Parquet Pipeline

Convert the initial CSV datasets to Parquet using **Snappy** compression:

```
 books = pd.read_csv("books.csv")
reviews = pd.read_csv("reviews.csv")

books.to_parquet("books.parquet", compression="snappy")
reviews.to_parquet("reviews.parquet", compression="snappy")
```

1.

Re-run the same ETL process but reading/writing **Parquet** files:

```
 tracker = EmissionsTracker(project_name="parquet_pipeline")
tracker.start()

books = pd.read_parquet("books.parquet")
reviews = pd.read_parquet("reviews.parquet")
```

```
df = (
    reviews.merge(books, on="Title", how="inner")
    .dropna(subset=["review/score", "Authors"])
)
df.to_parquet("merged_books_reviews.parquet", compression="snappy")

tracker.stop()
```

2. Log results in `emissions_parquet.csv`.

Task 3 — Comparison and Analysis

Create a summary table:

Step	Format	Duration (s)	Energy (kWh)	CO_2 (kg)	Output Size (MB)
Load	CSV				
Load	Parquet				
Save	CSV				
Save	Parquet				

Then:

- Plot bar charts comparing **runtime** and **CO_2 emissions** per format.
- Discuss:
    - Which format performs faster?
    - Which one emits less CO_2?
    - How much storage is saved using Parquet?
    - Does compression always reduce emissions?

---

6. Task 4 — Eco-Design Experiment

Choose one optimization and re-measure emissions:

- Filter out unused columns before saving.
- Change compression codec (`gzip`, `brotli`, `snappy`).
- Reduce dataset size (sample 50%).
- Cache intermediate results (optional in PySpark version).

Compare before/after CO_2 emissions and explain differences.

---

7. Deliverables

Each group must submit:

- `tp_codecarbon_parquet.ipynb`
- `emissions_csv.csv` and `emissions_parquet.csv`
- A figure comparing runtime and emissions for both formats
- A **10-line reflection** discussing trade-offs between readability, performance, and sustainability of file formats.

---

8. Expected Outcomes

After completing this TP, you should observe that:

- Parquet files are smaller (≈ 5–10 × reduction).
- Reading and writing Parquet is significantly faster and greener than CSV.

- Choosing the right file format can reduce **storage cost**, **I/O load**, and **CO_2 emissions** simultaneously.