Week 6: File Ingestion and Schema Validation

1. Objective

The goal of this assignment was to:

- Ingest a large CSV dataset (credit card transactions) using multiple frameworks: **Pandas, Dask, Modin**.
- Clean column names and validate them against a predefined **YAML schema**.
- Write the cleaned data to **pipe-separated compressed (.gz) files**.
- Compare computational efficiency and output file sizes.

---

2. Dataset

- **File:** creditcard.csv
- **Rows:** 284,807
- **Columns:** 31

---

3. Methodology

1. **Reading the file**
    - **Pandas:** standard single-threaded read.
    - **Dask:** parallel read with assume_missing=True to handle dtype inference.
    - **Modin (Ray backend):** parallelized Pandas.
2. **Column cleaning**
    - Removed whitespace and special characters.
    - Replaced spaces with underscores.
3. **Schema validation**
    - Created a **YAML file** containing column names and CSV separator.
    - Validated ingested data for both **row count** and **column names**.
4. **File writing**
    - Wrote **pipe-separated (|) .gz compressed files** for all three frameworks.

---

4. Results

CSV Ingestion and Output Summary:

Framework	Rows	Columns	Time (s)	File Size (MB)
Pandas	284,807	31	3.87	65.41
Dask	284,807	31	2.57	65.48
Modin (Ray)	284,807	31	3.09	65.41

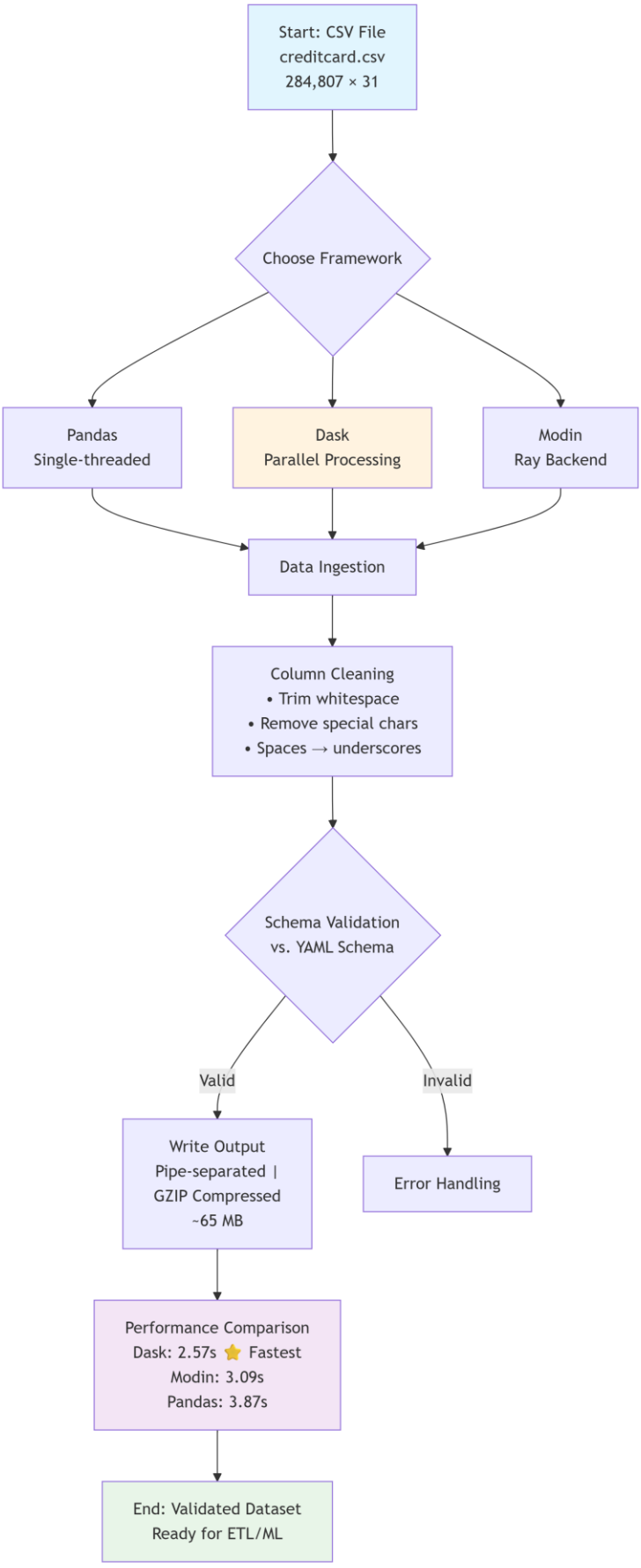
Observations:

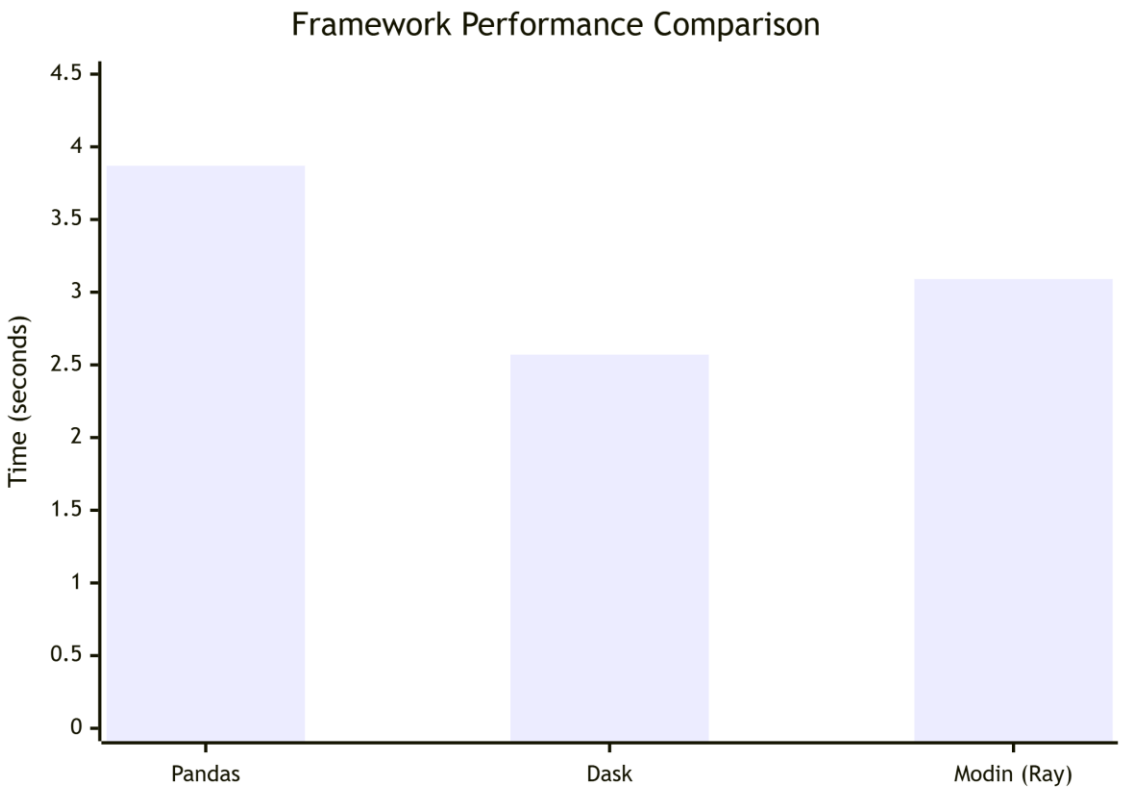
- **Dask** was the fastest due to parallel computation.
- **Modin** provided an easy-to-use parallel Pandas alternative with similar results.
- All frameworks produced **validated and consistent outputs**.
- Compressed pipe-separated files are approximately **65 MB**, suitable for downstream processing.

---

5. Conclusion

- The assignment demonstrates **efficient file ingestion, cleaning, schema validation, and writing large datasets** using multiple frameworks.
- **Dask** was the most efficient for ingestion, while **all frameworks produced identical outputs**.
- The workflow can easily scale to **larger datasets**, making it practical for real-world ETL tasks.


Start: CSV File
creditcard.csv
284,807 × 31
Choose Framework
Pandas
Single-threaded
Dask
Parallel Processing
Modin
Ray Backend
Data Ingestion
Column Cleaning
• Trim whitespace
• Remove special chars
• Spaces → underscores
Schema Validation
vs. YAML Schema
Valid
Invalid
Write Output
Pipe-separated |
GZIP Compressed
~65 MB
Error Handling
Performance Comparison
Dask: 2.57s ⭐ Fastest
Modin: 3.09s
Pandas: 3.87s
End: Validated Dataset
Ready for ETL/ML


Framework Performance Comparison
Time (seconds)
4.5
4
3.5
3
2.5
2
1.5
1
0.5
0
Pandas
Dask
Modin (Ray)