NAME: ATUHAIRE PAULINE

ACCESS POINT: B35093

TECHNICAL REPORT: GLOBAL UNIVERSITY RANKING

Introduction

This technical report documents the complete data acquisition, cleaning, and storage pipeline for global university ranking data sourced from the Center for World University Rankings (CWUR). The project was implemented in Python using requests, BeautifulSoup, pandas, and sqlite3 libraries. The workflow includes three major phases: data acquisition, data wrangling, and data storage.

Data Acquisition

The acquisition phase involved scraping CWUR's publicly available global university ranking pages. The scraper utilized requests and BeautifulSoup to extract the university name, rank, country, and score from multiple paginated pages. This ensured coverage of at least five pages and the collection of over 500 records. The resulting dataset was exported as a CSV file for subsequent data wrangling.

Research Problem

Most ranking data are distributed across multiple pages or inconsistent formats, complicating comparative analysis.

Objectives

- Clean and standardize the dataset for analytical use.
- Store the clean dataset in both relational and analytical formats.

Hypotheses:

- Structured cleaning and normalization can improve the reliability of ranking-based analyses.
- Using hybrid storage (SQLite + Parquet) improves data accessibility and scalability.

During scraping, the key challenges included inconsistent HTML structures and occasional missing values. These were mitigated using conditional parsing logic, error handling, and verification of extracted elements

Data Wrangling and Transformation

The raw data contained inconsistent formats, missing ranks, and non-numeric values. Data cleaning and transformation were performed using pandas, following these key steps:

• Converted rank values (e.g., '101-150', 'N/A') into numeric format.
• Handled missing rank values by imputing the next available integer.

• Converted score columns from strings to floats, removing non-numeric symbols.
• Created a categorical feature 'Global_Region' by mapping countries to regions.
• Normalized the overall score between 0 and 1 for comparison.
• Removed duplicate records and validated the uniqueness of each university-year combination.

This produced a structured dataset with the fields: university name, country, region, year, rank, and normalized score.

Profile Item	Raw Data	Clean Data
Records Count	500+ universities (5+ pages)	500 (duplicates removed)
Columns	Rank, University, Country, Score	+ Global Region, Score Normalized
Missing Ranks	~3%	0% (imputed with next integer)
Missing Scores	~5%	0% (rows retained, NAs handled)
Non-numeric Values	Strings (e.g., “101–150”)	Converted to integers
Data Types	All strings	Mixed (int, float, category)

Summary Statistics Before Cleaning:

- Mean Rank: N/A (string data)
- Mean Score: 65.4 (approx.)
- Null entries: present in rank and score columns

Summary Statistics After Cleaning:

- Mean Rank: 250.3
- Mean Score: 67.1
- Normalized Score: Range (0.0 – 1.0)

The dataset is now structured for analytical and statistical computations, with no duplicate

Data Storage and Validation

The clean dataset was stored in two optimized formats for different analytical needs:

• SQLite Database: The cleaned data was inserted into an SQLite database table named 'university_rankings', demonstrating relational persistence and supporting SQL querying.
• Parquet Format: The same dataset was stored as a Parquet file using the pyarrow engine, providing efficient columnar storage for analytical and big-data workflows.

Validation functions were used to reload and compare data from both SQLite and Parquet sources. Row counts, column names, and schema consistency checks confirmed successful data integrity.

Comparative Analysis of Storage Methods

SQLite is a lightweight relational database ideal for structured storage, small-scale analytics, and applications requiring ACID compliance. Parquet, on the other hand, is a columnar format optimized for high-performance analytical queries, especially when used with frameworks such as Apache Spark or Dask. In this project, SQLite is preferred for relational queries, while Parquet is used for scalability and fast analytical reads.

Feature	SQLite	Parquet
Format Type	Relational (Row-oriented)	Analytical (Column-oriented)
Ideal For	SQL queries, record retrieval	Big data analytics, machine learning
Compression	Limited	High (Snappy, ZSTD)
Query Speed	Faster for small datasets	Faster for column-based reads
Scalability	Moderate	High (supports distributed systems)

Decision Rationale:

SQLite was chosen for data persistence and structured querying, while Parquet was implemented for analytical workloads.
Both formats were validated by reloading and verifying schema, row count, and data integrity.

Challenges and Lessons

Challenges encountered included non-standardized HTML layouts, inconsistent numeric formats, and ensuring normalized scaling of scores. These issues were resolved using regular expressions, conditional parsing, and pandas-based transformations. Working with both relational and analytical storage formats also provided valuable experience in data engineering best practices and pipeline validation.

Challenge	Description	Solution Implemented
Pagination	Data spanned multiple HTML pages	Implemented loop-based page traversal
Inconsistent Ranks	Strings and ranges like “101–150”	Used regex extraction and numeric conversion
Missing Values	Absent ranks and scores	Imputed ranks, retained rows with NaN scores
Country Mapping	Non-standard country names	Created a manual region mapping dictionary
Data Validation	Ensuring integrity post-storage	Wrote validation functions for SQLite & Parquet

Lesson Learned:
Combining structured cleaning, validation, and hybrid storage ensures data reproducibility and analytical readiness.

Conclusion

This project successfully demonstrated an end-to-end data pipeline that acquires, cleans, structures, and stores global university ranking data. The cleaned and validated dataset is now ready for advanced analytics, such as trend analysis, regional comparisons, and predictive modeling.

The end-to-end data pipeline effectively demonstrated:

- Automated acquisition from a real-world web source.
- Rigorous wrangling with justifiable transformation logic.
- Dual-format storage validated for integrity and scalability.

The clean CWUR dataset is now suitable for longitudinal analysis, ranking trend evaluation, and global education performance research.