DSCI 551 Project Proposal: Data2App

Student: Amey Chede
Group Size: 1 person
Section: MW
Date: September 19, 2025

---

1. Chosen Project Option

I have chosen the **SQL option** for this project.

Rationale: The SQL option is more suitable for a one-person project because:

- CSV parsing is more straightforward than nested JSON parsing
- DataFrame operations have clear, well-defined semantics
- The assignment provides concrete pandas examples to reference
- I can focus more time on building a polished application rather than debugging complex parsing

---

2. Implementation Plan

2.1 Core Components

Parsing Component:

- Develop a custom CSV parser similar to `pd.read_csv()`
- Support basic functionality: file reading, column separator specification
- Handle edge cases: quoted fields, escaped characters, empty cells
- Return data in a custom DataFrame-like structure

DataFrame Structure:

- Implement custom DataFrame class using dictionary storage (column name → list of values)
- Implement `__getitem__` method for bracket-style data access (e.g., `df['column_name']`)
- Support basic data types and handle missing values

Required Functions:

1. **Filtering:** Select rows based on conditions (e.g., `df[df.column > value]`)
2. **Projection:** Select specific columns (e.g., `df[['col1', 'col2']]`)
3. **Group By:** Group data by column values
4. **Aggregation:** Compute statistics (sum, mean, max, min, count) for grouped data
5. **Join:** Merge two DataFrames on specified columns

2.2 Application Development

Application Type: Olympic Performance Analytics Dashboard

Dataset: Olympic Games Historical Data (120+ years of Olympics)

(https://www.kaggle.com/datasets/heesoo37/120-years-of-olympic-history-athletes-and-results?resource=download)

- **Primary dataset**: `athlete_events.csv` (~270K athletes, 40MB)
  - Columns: Name, Sex, Age, Height, Weight, Team, NOC, Games, Year, Season, City, Sport, Event, Medal
- **Secondary dataset**: `noc_regions.csv` (230 rows, country code mappings)
  - Columns: NOC (country code), region (country name), notes
- **Join key**: NOC column connects athletes to their countries
- **Size**: Large-scale real-world dataset perfect for demonstrating scalability

Application Features:

- **Data Loading**: Parse both CSV files and display basic dataset statistics
- **Filtering Examples**:
  - Find all gold medalists from specific countries
  - Athletes competing in multiple Olympics
  - Medal winners by age ranges or sports
- **Projection Examples**: Extract athlete names and their medal counts
- **Grouping & Aggregation**:
  - Medal counts by country and sport
  - Average athlete age/height/weight by sport
  - Historical medal trends by decade
- **Join Operations**: Connect athlete data with country names for readable reports
- **Summary Reports**: Generate insights like "Top 10 Countries by Gold Medals" or "Most Successful Athletes"

---

3. Timeline and Milestones

Week	Dates	Tasks	Deliverables
Week 1	Sep 16-22	• ✅ Submit project proposal (Sep 19)<br>• Design DataFrame class structure<br>• Begin CSV parser implementation	**Proposal (Sep 19)**
Week 2-3	Sep 23 - Oct 6	• Complete CSV parser with basic functionality<br>• Implement DataFrame class with `__getitem__` method<br>• Test parsing with sample Olympic datasets	
Week 4-5	Oct 7-20	• Implement filtering and projection functions<br>• Implement groupby functionality<br>• Test with medium-sized data subsets	**Midterm Report (Oct 17)**
Week 6-7	Oct 21 - Nov 3	• Implement aggregation functions (sum, mean, max, min, count)<br>• Implement join functionality for athlete-country data<br>• Begin application development and user interface	
Week 8-9	Nov 4-17	• Complete Olympic Analytics Dashboard application<br>• Integrate all functions into the application<br>• Test with full 40MB dataset and optimize performance<br>• Handle edge cases and error scenarios	
Week 10	Nov 18-24	• Finalize code and comprehensive documentation<br>• Prepare demo presentation with interesting Olympic insights	**Code Submission (Nov 23)**<br>**Live Demo (Nov 24/25)**
Finals Week	Dec 15	• Complete final project report<br>• Submit comprehensive documentation	**Final Report (Dec 15)**

---

4. Group Formation and Division of Labor

Group Member: [Your Name] - Solo project

Individual Responsibilities:

- **Parsing & Data Structure (Weeks 1-3):** Design and implement CSV parser and DataFrame class with dictionary-based storage
- **Core Functions (Weeks 4-7):** Implement all required data manipulation functions (filter, project, groupby, aggregate, join)
- **Application Development (Weeks 8-9):** Build end-to-end Olympic Analytics Dashboard demonstrating all functions
- **Testing & Documentation (Week 10):** Comprehensive testing with large dataset and documentation

---

5. Technical Implementation Details

DataFrame Internal Structure:

```
# Dictionary-based storage as specified in assignment
data = {
    'Name': ['Michael Phelps', 'Usain Bolt', ...],
    'NOC': ['USA', 'JAM', ...],
    'Medal': ['Gold', 'Gold', ...]
}
```

Key Technical Challenges:

- Efficient parsing of 40MB CSV file
- Memory-efficient dictionary operations
- Fast filtering and grouping algorithms
- Clean join implementation between athlete and country data

Success Metrics:

- Parse full Olympic dataset (270K+ rows) successfully
- Demonstrate all 5 required operations working correctly
- Build functional analytics dashboard
- Generate meaningful insights from real Olympic data

---

6. Risk Management

Potential Challenges:

- Large dataset size (40MB) may require memory optimization
- CSV parsing edge cases (quoted fields, different encodings)
- Complex join operations with large datasets

Mitigation Strategies:

- Start with subset of data, gradually scale up
- Implement robust CSV parsing with proper error handling
- Use efficient algorithms for dictionary operations
- Maintain regular backups and version control
- Test incrementally rather than building entire system first

---

7. Learning Objectives

Through this project, I expect to gain:

- Deep understanding of DataFrame/pandas internals and implementation
- Experience with efficient data structure design using Python dictionaries
- Skills in parsing and large-scale data manipulation algorithms
- Knowledge of building data-driven applications from scratch
- Insights into performance optimization for data processing

Note: This proposal represents my committed plan for the SQL option. I understand that changing options after proposal submission is not permitted, and I am confident in my ability to deliver all components on schedule.