DuckDB-based CSV Search Pipeline

High-Performance Data Processing Documentation

1 Overview

Project Summary

This project is a scalable and efficient data processing pipeline built in Python, designed to:

- Convert large CSV files to optimized Parquet format
- Load and index the data into DuckDB
- Perform high-speed searches using indexed fields
- Export filtered results in multiple formats

2 Dependencies

Required Dependencies:

- Python 3.8+
- `duckdb`
- `pandas`
- `pyarrow`
- `tabulate`

Installation:

```
python -m venv .venv
source .venv/bin/activate
pip install -r requirements.txt
```

Listing 1: Environment Setup

3 Pipeline Steps

3.1 1. CSV to Parquet Conversion

Script: `convert_to_parquet.py`

Features:

- Converts large CSV files into Parquet format using chunked streaming.
- Automatically creates directories and logs the process.

Usage:

```
python convert_to_parquet.py --csv data/sample.csv \
                             --parquet data/sample.parquet \
                             --chunk 1000000 \
                             --load yes
```

Listing 2: CSV to Parquet Conversion

3.2 2. Data Loading and Indexing

Script: `load_and_index.py`

Functionality:

- Loads Parquet file into DuckDB.
- Creates a persistent table and optionally recreates it.

Note: No CLI arguments are required for this script.

3.3 3. Search and Export

Script: `search_and_export.py`

Capabilities:

- Searches indexed or filterable fields using range or equality conditions.
- Exports results in CSV, JSON, or Parquet.

Usage Examples:

```
# Equality match
python search_and_export.py --equals col_2=foo col_3=bar \
                            --columns col_2 col_3 col_4 \
                            --format csv
```

Listing 3: Equality Match Search

```
# Range filter
python search_and_export.py --range col_0 10 50 \
                            --columns col_0 col_1 \
                            --format json
```

Listing 4: Range Filter Search

3.4 4. Logging

Logging System

Each script writes a timestamped log file under the `logs/` directory, and also streams logs to the terminal.

4 Error Handling

The pipeline includes validations and exception handling for:

- Missing or invalid files
- Invalid search fields or columns
- DuckDB connection issues
- Empty query results

All errors are logged in the `logs/` directory for audit and debugging purposes.

5 Logging Format

Log Structure

Each run produces a detailed log file that contains:

- Timestamp of execution
- CLI arguments passed
- Number of rows processed
- Any warnings or errors encountered

Sample Log Output:

```
2025-06-23 14:30:12 [INFO] Running query: SELECT col_1 FROM data WHERE col_0
    BETWEEN 10 AND 50
2025-06-23 14:30:12 [INFO] Found 3,920 rows.
2025-06-23 14:30:12 [INFO] Exported to data/search_output.csv
```

Listing 5: Example Log Output

6 Folder Structure

```
.
        data/
            sample.csv
            sample.parquet
            engine.duckdb
        logs/
            *.log
        convert_to_parquet.py
        load_and_index.py
        search_and_export.py
        requirements.txt
```

Listing 6: Project Directory Structure

7 Appendix: Sample Schema

Dataset Schema

The test dataset contains 50 columns named `col_0` to `col_49`, with a mix of types:

- `col_0, col_1, col_2` are indexed (INTEGER)
- `col_3` to `col_20` are FLOAT
- `col_21` to `col_45` are VARCHAR
- `col_46` to `col_49` may include TEXT/BLOB data

Author Information

Harsh Prakash

Email: `harshprakash06@gmail.com`