Đại học Quốc gia Thành phố Hồ Chí Minh
Trường Đại học Bách Khoa
Khoa Khoa học và Kỹ thuật Máy tính

LẬP TRÌNH NÂNG CAO (CO2039)

Bài Tập Lớn

XÂY DỰNG APP TRA CỨU SAO KÊ

GVHD: Lê Đình Thuận
Lớp-Nhóm: L01-503
SVTH: Võ Trần Phi Phong
MSSV: 2420006

Contents

1 Nội Dung Đề Tài

Xây dựng một web/app tra cứu dữ liệu sao kê của MTTQ VN với khoảng 200.000 bản ghi. Ứng dụng cần có giao diện (web/app/CLI) có thể tra cứu thông tin sao kê theo: số tiền, tên người gửi, nội dung.

2 Phương Pháp Tiếp Cận

2.1 Ngôn Ngữ

Python rất phù hợp để tìm kiếm dữ liệu trong tệp CSV có khoảng 200.000 bản ghi. Mặc dù 200.000 bản ghi là một lượng dữ liệu đáng kể, nhưng các thư viện của Python có thể xử lý một cách hiệu quả.

1. Pandas: là một thư viện tuyệt vời để làm việc với các tập dữ liệu lớn. Nó có thể đọc nhanh các tệp CSV vào DataFrame, cho phép thao tác và tìm kiếm dữ liệu nhanh chóng. Pandas có thể xử lý dữ liệu với hàng triệu hàng và các tối ưu hóa như sử dụng `dtype` có thể cải thiện hiệu suất.

2. CSV module: là một thư viện built-in của Python, làm việc với dữ liệu CSV thô mà không cần tải toàn bộ tập dữ liệu vào bộ nhớ bằng cách đọc dữ liệu theo từng dòng. Phương pháp này sử dụng ít bộ nhớ hơn nhưng có thể chậm hơn nếu tìm kiếm nhiều điều kiện hoặc thực hiện các truy vấn phức tạp.

3. SQLite: để có chức năng tìm kiếm nâng cao hơn và tối ưu hóa, tải dữ liệu CSV vào cơ sở dữ liệu SQLite. SQLite nhẹ và cho phép truy vấn SQL nhanh.

2.2 Thư Viện

CSV module là một thư viện Python tích hợp được sử dụng để đọc và ghi tệp CSV.

- `csv.DictReader`: đọc tệp CSV và trả về mỗi hàng dưới dạng dictionary, trong đó key là tên cột và value là giá trị hàng. Điều này hữu ích khi xử lý dữ liệu CSV một cách linh động.

OS module cung cấp một cách thức để tương tác với hệ điều hành, chẳng hạn như đọc đường dẫn file, thư mục và các chức năng khác ở cấp độ hệ điều hành.

- `os.path.dirname` và `os.path.abspath`: tìm đường dẫn tuyệt đối của thư mục hiện tại và phục vụ tệp `index.html` một cách chính xác.

2.2 Thư Viện

Flask là một web framework nhẹ cho Python được sử dụng để tạo các ứng dụng web. Trong ứng dụng này, Flask được sử dụng để tạo các API endpoint, xử lý các HTTP request, xử lý dữ liệu từ file CSV và trả kết quả cho người dùng. Flask đóng vai trò như một backbone của máy chủ web, quản lý các yêu cầu từ người dùng và trả về các phản hồi phù hợp.

- `Flask`: core class, được sử dụng để khởi tạo ứng dụng
- `request`: xử lý dữ liệu HTTP request (ví dụ: tham số truy vấn)
- `jsonify`: chuyển đổi các đối tượng Python (như dictionary) thành kiểu dữ liệu JSON
- `send_from_directory`: phục vụ các tệp tĩnh (static), trong trường hợp này là `index.html`

3 Cấu trúc dữ liệu

Một file CSV (Comma Separated Values) là tệp văn bản chứa dữ liệu theo định dạng có cấu trúc, trong đó mỗi giá trị được phân cách bằng dấu phẩy. Mỗi dòng biểu diễn một bản ghi và mỗi giá trị trong dòng tương ứng với một trường trong bản ghi.

Ví dụ data từ file CSV: `date_time,trans_no,credit,debit,detail` là các header.

```
date_time,trans_no,credit,debit,detail
03/09/2024_5216.65140,9,200000,0,120167.030924.100642.NGUYEN
   HOAI NAM ung ho
03/09/2024_5017.43849,13,9000,0,888828.030924.121121.NGUYEN
   THI KIEU OANH chuyen tien
04/09/2024_5240.80637,55,500000,0,MBVCB.6944619812.chung tay
   gop suc .CT tu 0351000801710 TRAN DANH VU toi
   001100193241 8 MAT TRAN TO QUOC VN - BAN CUU TRO TW
```

3 Cấu trúc dữ liệu

Đối tượng `csv.DictReader` là một iterator đọc file CSV và ánh xạ từng hàng vào một dictionary trong đó key là tên cột và value là dữ liệu tương ứng trong hàng đó. Ngoài ra, đối tượng `fieldnames` trả về danh sách tên cột xuất hiện ở hàng đầu tiên của file CSV.

Dữ liệu được đọc trong Python:


```
  {
    "date_time": "03/09/2024_5216.65140",
    "trans_no": "9",
    "credit": "200000",
    "debit": "0",
    "detail": "120167.030924.100642.NGUYEN HOAI NAM ung ho"
  },
  {
    "date_time": "03/09/2024_5017.43849",
    "trans_no": "13",
    "credit": "9000",
    "debit": "0",
    "detail": "888828.030924.121121.NGUYEN THI KIEU OANH chuyen
        tien"
  },
```

3 Cấu trúc dữ liệu

Các key biểu thị tên cột là điều kiện tìm kiếm (như `credit`, `detail`) và các giá trị là các giá trị cần tìm kiếm. Kết quả tìm kiếm được lưu trữ trong danh sách dictionary. Mỗi dictionary đại diện cho một hàng từ file CSV phù hợp với điều kiện tìm kiếm.

Flask cung cấp `jsonify` để chuyển đổi các cấu trúc dictionary sang định dạng JSON.

```
results = [
  { "date_time": "03/09/2024_5017.43849", "trans_no": "13", "
     credit": "9000", "debit": "0", "detail": "
     888828.030924.121121.NGUYEN THI KIEU OANH chuyen tien" }
]
```

4 Thuật toán

Lọc tìm kiếm: ứng dụng lọc dữ liệu dựa trên các điều kiện tìm kiếm bằng cách sử dụng phương pháp **linear search**. Đối với mỗi hàng trong CSV, ứng dụng sẽ kiểm tra xem hàng đó có đáp ứng tất cả các điều kiện do người dùng chỉ định hay không. Các bước thực hiện:

- Lặp lại qua từng hàng của tệp CSV.
- Đối với mỗi điều kiện trong `search_conditions`, kiểm tra xem hàng đó có đáp ứng điều kiện không.
  - Đối với các chuỗi khớp chính xác (`detail`, `trans_no`), hãy kiểm tra xem giá trị của cột có chứa thuật ngữ tìm kiếm không.
  - Đối với các phạm vi (`credit_min`, `credit_max`), kiểm tra xem giá trị có nằm trong phạm vi đã chỉ định không.
  - Nếu hàng khớp với tất cả các điều kiện, hàng đó sẽ được thêm vào danh sách kết quả.

```
    with open(file_path, mode="r", newline="") as file:
      reader = csv.DictReader(file)
      results = []

      for row in reader:
        match = True
        for column_name, search_term in search_conditions.items()
            :
          if column_name == "credit_min":
            if float(row["credit"]) < search_term:
              match = False
              break
          elif column_name == "credit_max":
            if float(row["credit"]) > search_term:
              match = False
              break
          else:
            if search_term not in row[column_name].lower():
              match = False
              break
        if match:
          results.append(row)

      return results
```

4 Thuật toán

Time Complexity:

- Đọc file CSV: time complexity là $O(n)$, trong đó n là số hàng trong file CSV.
- Tìm kiếm: quá trình tìm kiếm bao gồm việc lặp lại từng hàng và kiểm tra từng điều kiện tìm kiếm. Time complexity trong trường hợp xấu nhất là $O(n \times m)$, trong đó n là số hàng và m là số điều kiện trong `search_conditions`.

5 Thiết lập môi trường

Trong thư mục chứa dự án, tiến hành mở Terminal:

1. Cài đặt môi trường ảo (venv): `$ sudo apt install python3-venv`
2. Tạo một môi trường ảo mới có tên `flask_env`: `$ python3 -m venv flask_env`
3. Kích hoạt: `$ source flask_env/bin/activate`
4. Cài đặt Flask trong môi trường ảo: `(flask_env)$ pip install Flask`
5. Chạy ứng dụng Python: `(flask_env)$ python3 app.py`
6. Để thoát khỏi môi trường ảo: `(flask_env)$ deactivate`