## 1. Kết nối đến Cơ sở dữ liệu

Để làm việc với cơ sở dữ liệu, đầu tiên cần cấu hình kết nối trong hai file testHelper.py và Interface.py. Cụ thể, trong hàm getOpenConnection, phải thay đổi thông tin người dùng và mật khẩu mặc định bằng thông tin đăng nhập PostgreSQL trên máy cá nhân.

Ví dụ:

```
def getopenconnection(user='postgres', password='1234', dbname='postgres'):
    return psycopg2.connect("dbname='" + dbname + "' user='" + user + "localhost" + password + "'")
```

## 2. Cài đặt hàm LoadRatings

Hàm loadratings có nhiệm vụ đọc dữ liệu từ một tệp văn bản chứa thông tin đánh giá phim và lưu trữ vào một bảng trong cơ sở dữ liệu PostgreSQL. Dữ liệu được lưu dưới định dạng userid:movieid:rating:timestamp – với dấu : là dấu phân cách giữa các trường.

Cú pháp hàm:

```
def loadratings(ratingstablename, ratingsfilepath, openconnection):
```

- ratingstablename: Tên bảng cần tạo để lưu dữ liệu đánh giá.
- ratingsfilepath: Đường dẫn tới tệp dữ liệu đánh giá (dạng text).
- openconnection: Kết nối mở tới cơ sở dữ liệu PostgreSQL.

Các bước thực hiện:

**B1. Tạo bảng tạm để lưu dữ liệu thô:**

```
cur.execute("create table " + ratingstablename + "(userid integer, extra1 char, movieid integer, extra2 char, rating float, extra3 char, timestamp bigint);")
```

Vì dữ liệu trong file sử dụng dấu : làm dấu phân cách, PostgreSQL cần định nghĩa đủ số cột tạm để lưu tất cả các thành phần, bao gồm các dấu : không cần thiết (extra1, extra2, extra3).

**B2. Chèn dữ liệu từ file vào bảng:**

```
cur.copy_from(open(ratingsfilepath), ratingstablename, sep=':')
```

**B3. Xoá các cột thừa (extra1, extra2, extra3, timestamp):**

```
cur.execute("alter table " + ratingstablename + " drop column extra1, drop column extra2, drop column extra3, drop column timestamp;")
```

**B4. Lưu thay đổi và đóng kết nối:**

```
cur.close()
con.commit()
```

Kết quả:
Sau khi thực thi hàm, bảng ratingstablename sẽ chứa dữ liệu dạng:

| | userid integer | movieid integer | rating double precision |
|---|---|---|---|
| 1 | 1 | 122 | 5 |
| 2 | 1 | 185 | 5 |
| 3 | 1 | 231 | 5 |
| 4 | 1 | 292 | 5 |
| 5 | 1 | 316 | 5 |
| 6 | 1 | 329 | 5 |

3. **Cài đặt hàm Range_Partition**

Tính khoảng chia delta đại diện cho độ rộng của rating trên mỗi phân vùng.

Tạo vòng lặp từ 0 đến số lượng phân vùng – 1, với mỗi vòng lặp:

- Tính khoảng rating bắt đầu bằng cách lấy delta nhân với biến đếm vòng lặp.
- Tính khoảng rating kết thúc bằng cách lấy khoảng bắt đầu cộng với delta.
- Tạo bảng mới đại diện cho phân vùng.
- Chèn dữ liệu từ bảng gốc vào phân vùng với điều kiện rating lớn hơn khoảng bắt đầu và nhỏ hơn hoặc bằng khoảng kết thúc (Với phân vùng đầu tiên sẽ dùng điều kiện rating lớn hơn hoặc bằng khoảng bắt đầu để không bỏ sót rating = 0).

4. **Cài đặt hàm RoundRobin_Partition**

Tạo vòng lặp từ 0 đến numberofpartitions - 1, với mỗi vòng lặp:

- Tạo bảng mới đại diện cho phân vùng.
- Chèn dữ liệu từ bảng gốc vào phân vùng theo nguyên tắc:
  - Đánh số thứ tự tất cả các dòng của bảng gốc bằng cách sử dụng hàm ROW_NUMBER().
  - Với mỗi dòng, tính chỉ số phân vùng bằng công thức:
  - (row_number − 1) mod numberofpartitions
  - Chỉ những dòng có kết quả bằng i mới được chèn vào phân vùng thứ i.

5. **Cài đặt hàm Range_Insert**

Chèn dòng dữ liệu mới vào bảng chính.

Lấy số lượng phân vùng hiện tại bằng cách gọi hàm count_partitions(...) với tiền tố range_part.

Tính độ rộng của mỗi phân vùng (delta): 5/numberofpartitions.

Xác định chỉ số phân vùng (index) dựa trên giá trị rating:

**Index = int(rating/delta)**

Xử lý biên trái: Nếu rating nằm đúng ở ranh giới chia và index != 0, thì giảm index đi 1 để đảm bảo tính đúng phạm vi.

Tạo tên bảng phân vùng theo dạng: range_part{index}.

Chèn dòng dữ liệu mới vào bảng phân vùng tương ứng.

6. **Cài đặt hàm RoundRobin_Insert**

Chèn dòng mới vào bảng chính.

Đếm tổng số dòng hiện có trong bảng chính (sau khi thêm dòng mới).

Lấy số lượng phân vùng hiện tại bằng cách gọi hàm count_partitions(...) với tiền tố rrobin_part.

Xác định chỉ số phân vùng (index) cần thêm dòng vào bằng công thức:

**Index = (total_rows − 1) mod numberofpartitions**

Tạo tên bảng phân vùng theo dạng rrobin_part{index}.

Chèn dòng dữ liệu mới vào bảng phân vùng tương ứng.

7. **Bổ sung**

Chỉnh sửa code cho phép nhập số lượng phân mảnh mong muốn cho các hàm tạo phân mảnh và cho phép nhập vào dữ liệu trước khi thực hiện các hàm insert.