**BỘ CÔNG THƯƠNG**

**TRƯỜNG ĐẠI HỌC CÔNG THƯƠNG TP. HỒ CHÍ MINH**

**KHOA CÔNG NGHỆ THÔNG TIN**

🙜🙢🙠🙞



**ĐỒ ÁN**

**CƠ SỞ DỮ LIỆU NOSQL**

**THIẾT KẾ CSDL ĐỒ THỊ QUẢN LÝ THÔNG TIN GIA PHẢ VÀ XÂY DỰNG ỨNG DỤNG MINH HỌA**

Giảng viên hướng dẫn: ThS. Trần Thị Vân Anh

Sinh viên thực hiện:

1. 2001215707 – Nguyễn Từ Thành Đạt
2. 2001216070 – Dương Thuận Quang
3. 2001216112 – Trương Ngọc Sơn
4. 2001216181 – Ngô Trường Thịnh
5. 2001215830 – Trương Quốc Huy

TP. HỒ CHÍ MINH – 2024

MỤC LỤC

[MỤC LỤC 2](#_Toc179457384)

[CHƯƠNG 1: THIẾT KẾ MÔ HÌNH DỮ LIỆU 3](#_Toc179457385)

[1.1 Khảo sát hệ thống quản lý 3](#_Toc179457386)

[1.2 Thiết kê mô hình dữ liệu 4](#_Toc179457387)

[1.3 Xây dựng cơ sở dữ liệu dựa vào mô hình đã thiết kế 5](#_Toc179457388)

[CHƯƠNG 2: KHAI THÁC DỮ LIỆU 8](#_Toc179457389)

[2.1 Các yêu cầu khai thác dữ liệu và lệnh truy vấn tương ứng 8](#_Toc179457390)

[2.2 Sao lưu và phục hồi cơ sở dữ liệu 8](#_Toc179457391)

1. THIẾT KẾ MÔ HÌNH DỮ LIỆU
   * 1. Khảo sát hệ thống quản lý
        1. Mục tiêu của hệ thống

Hệ thống quản lý thông tin gia phả nhằm tạo ra một nền tảng cho phép người dùng dễ dàng lưu trữ, quản lý và khai thác thông tin về các thành viên trong gia đình. Mục tiêu chính bao gồm:

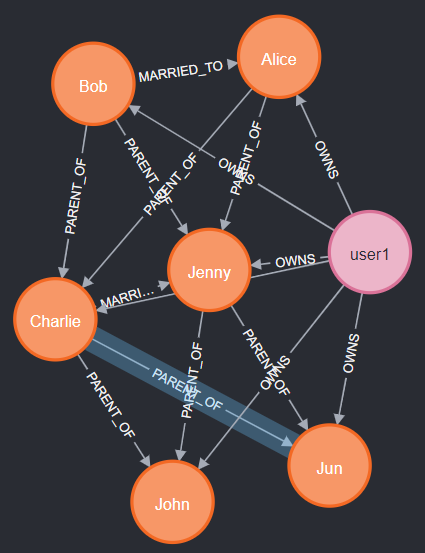
* Lưu trữ thông tin cá nhân: Hệ thống cần ghi nhận thông tin cá nhân của từng thành viên, bao gồm tên, ngày sinh, giới tính, địa chỉ và các thông tin bổ sung khác.
* Quản lý mối quan hệ: Cung cấp khả năng thiết lập và duy trì các mối quan hệ trong gia đình, từ cha mẹ, con cái cho đến ông bà và các quan hệ phức tạp khác.
* Khả năng truy vấn linh hoạt: Hệ thống cần hỗ trợ tìm kiếm và truy vấn thông tin theo nhiều tiêu chí khác nhau, giúp người dùng dễ dàng tìm kiếm thông tin mà họ cần.
  + - 1. Phân tích yêu cầu
* Yêu cầu chức năng:
  + Thêm, sửa, xóa thông tin cá nhân.
  + Thiết lập và quản lý các mối quan hệ giữa các thành viên.
  + Hiển thị thông tin cá nhân và các mối quan hệ.
  + Tìm kiếm và truy vấn thông tin theo tên, mối quan hệ và các thuộc tính khác.
  + Sao lưu và phục hồi cơ sở dữ liệu để đảm bảo an toàn dữ liệu.
* Yêu cầu phi chức năng:
  + Giao diện thân thiện, dễ sử dụng cho người dùng không chuyên.
  + Tính ổn định và hiệu suất cao trong việc xử lý truy vấn.
  + Bảo mật thông tin cá nhân của người dùng.
    - 1. Các đối tượng người dùng

Hệ thống sẽ hỗ trợ cho các đối tượng người dùng khác nhau, bao gồm:

* Người dùng cá nhân: Những người muốn lưu trữ và quản lý thông tin gia phả cho gia đình mình.
* Nhà nghiên cứu: Những người cần tìm hiểu về lịch sử gia đình hoặc các mối quan hệ trong một dòng họ cụ thể.
  + 1. Thiết kê mô hình dữ liệu
       1. Mô hình dữ liệu đồ thị

Để phù hợp với yêu cầu quản lý thông tin gia phả, mô hình dữ liệu được xây dựng dựa trên cấu trúc đồ thị, với các node và các mối quan hệ cụ thể. Mô hình này cho phép việc thể hiện các mối quan hệ một cách trực quan và dễ dàng hơn so với các mô hình quan hệ truyền thống.

* Nodes:
* Person: Mỗi node Person đại diện cho một thành viên trong gia đình.
* Attributes:
  + name: Tên người.
  + date\_of\_birth: Ngày sinh.
  + gender: Giới tính (Male/Female).x
  + address: Địa chỉ nơi cư trú.
  + phone\_number: Số điện thoại liên lạc.
  + occupation: Nghề nghiệp.
  + date\_of\_death: Ngày mất (nếu có).
  + biography: Tiểu sử ngắn gọn.
* Relationships:
  + PARENT\_OF (Cha mẹ của): Liên kết giữa cha mẹ và con cái.
  + MARRIED\_TO (Kết hôn với): Liên kết giữa hai người đã kết hôn.
    - 1. Sơ đồ mô hình dữ liệu



* + 1. Xây dựng cơ sở dữ liệu dựa vào mô hình đã thiết kế

Sử dụng Neo4j để tạo cơ sở dữ liệu theo mô hình đã thiết kế. Dưới đây là ví dụ về lệnh tạo node và mối quan hệ trong Neo4j:

CREATE (user1:User {

username: 'user1',

password: '123'

}),

(alice:Person {

name: 'Alice',

date\_of\_birth: '1950-01-01',

gender: 'Female',

address: '123 Main St',

phone\_number: '123456789',

occupation: 'Teacher',

date\_of\_death: '2020-12-20'

}),

(bob:Person {

name: 'Bob',

date\_of\_birth: '1948-05-05',

gender: 'Male',

address: '123 Main St',

phone\_number: '987654321',

occupation: 'Engineer',

date\_of\_death: '2018-03-15'

}),

(charlie:Person {

name: 'Charlie',

date\_of\_birth: '1978-10-10',

gender: 'Male',

address: '789 Oak St',

phone\_number: '555555555',

occupation: 'Software Engineer',

date\_of\_death: null

}),

(jenny:Person {

name: 'Jenny',

date\_of\_birth: '1980-08-12',

gender: 'Female',

address: '123 Main St',

phone\_number: '666666666',

occupation: 'Nurse',

date\_of\_death: null

}),

(john:Person {

name: 'John',

date\_of\_birth: '2003-08-10',

gender: 'Male',

address: '123 Main St',

phone\_number: '666666666',

occupation: 'Student',

date\_of\_death: null

}),

(jun:Person {

name: 'Jun',

date\_of\_birth: '2005-08-10',

gender: 'Male',

address: '123 Main St',

phone\_number: '666666666',

occupation: 'Student',

date\_of\_death: null

}),

(alice)-[:PARENT\_OF]->(charlie),

(alice)-[:PARENT\_OF]->(jenny),

(bob)-[:PARENT\_OF]->(charlie),

(bob)-[:PARENT\_OF]->(jenny),

(bob)-[:MARRIED\_TO]->(alice),

(jenny)-[:PARENT\_OF]->(john),

(jenny)-[:PARENT\_OF]->(jun),

(charlie)-[:PARENT\_OF]->(john),

(charlie)-[:PARENT\_OF]->(jun),

(charlie)-[:MARRIED\_TO]->(jenny),

(user1)-[:OWNS]->(alice),

(user1)-[:OWNS]->(bob),

(user1)-[:OWNS]->(charlie),

(user1)-[:OWNS]->(jenny),

(user1)-[:OWNS]->(john),

(user1)-[:OWNS]->(jun)

CREATE (user2:User {

username: 'user2',

password: '456'

}),

(david:Person {

name: 'David',

date\_of\_birth: '1985-03-03',

gender: 'Male',

address: '456 Elm St',

phone\_number: '444444444',

occupation: 'Doctor',

date\_of\_death: null

}),

(emily:Person {

name: 'Emily',

date\_of\_birth: '1992-12-12',

gender: 'Female',

address: '456 Elm St',

phone\_number: '333333333',

occupation: 'Artist',

date\_of\_death: null

}),

(sam:Person {

name: 'Sam',

date\_of\_birth: '1995-05-05',

gender: 'Male',

address: '456 Elm St',

phone\_number: '222222222',

occupation: 'Engineer',

date\_of\_death: null

}),

(david)-[:PARENT\_OF]->(sam),

(emily)-[:MARRIED\_TO]->(david),

(user2)-[:OWNS]->(david),

(user2)-[:OWNS]->(emily),

(user2)-[:OWNS]->(sam);

1. KHAI THÁC DỮ LIỆU
   * 1. Các yêu cầu khai thác dữ liệu và lệnh truy vấn tương ứng

Hệ thống sẽ cung cấp một loạt các yêu cầu khai thác dữ liệu. Dưới đây là một số yêu cầu tiêu biểu cùng với lệnh truy vấn tương ứng trong Neo4j:

**Yêu cầu 1: Lấy thông tin của một người.**

MATCH (p:Person {name: 'Charlie'})

RETURN p

**Yêu cầu 2: Lấy danh sách cha mẹ của một người.**

MATCH (p:Person {name: 'Charlie'})<-[:PARENT\_OF]-(parent)

RETURN parent

**Yêu cầu 3: Lấy danh sách anh chị em của một người.**

MATCH (p:Person {name: 'Charlie'})<-[:PARENT\_OF]-(parent)-[:SIBLING\_OF]->(sibling)

RETURN sibling

**Yêu cầu 4: Tìm cây gia phả của một người**

MATCH (u:User {username: ‘user1’ })-[:OWNS]->(p:Person {name: ‘Bob’ })

OPTIONAL MATCH (p)-[:PARENT\_OF\*1..]->(descendants)

OPTIONAL MATCH (p)-[:MARRIED\_TO]-(spouse)

RETURN p, collect(distinct descendants) AS descendants, spouse

**Yêu cầu 5: Tìm kiếm gia đình có mối quan hệ kết hôn.**

MATCH (p:Person)-[:MARRIED\_TO]-(spouse)

RETURN p, spouse

**Yêu cầu 6: Tìm kiếm những người đã mất.**

MATCH (p:Person)

WHERE p.date\_of\_death IS NOT NULL

RETURN p

* + 1. Sao lưu và phục hồi cơ sở dữ liệu

Sao lưu và phục hồi cơ sở dữ liệu là hai hoạt động quan trọng nhằm đảm bảo an toàn cho dữ liệu. Quá trình thực hiện như sau:

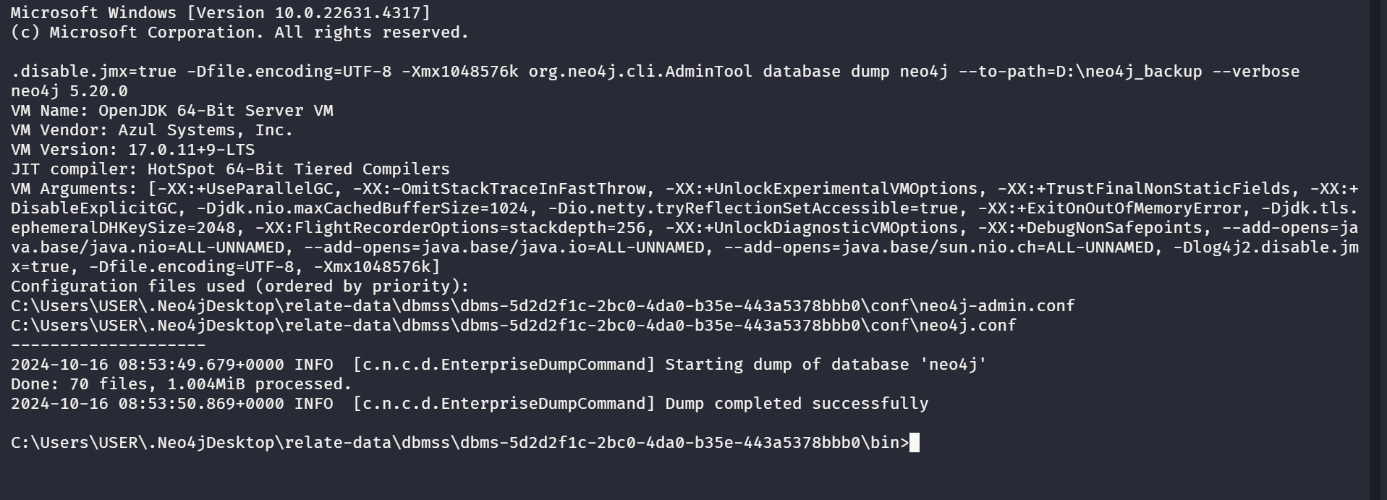
* **Sao lưu cơ sở dữ liệu:**

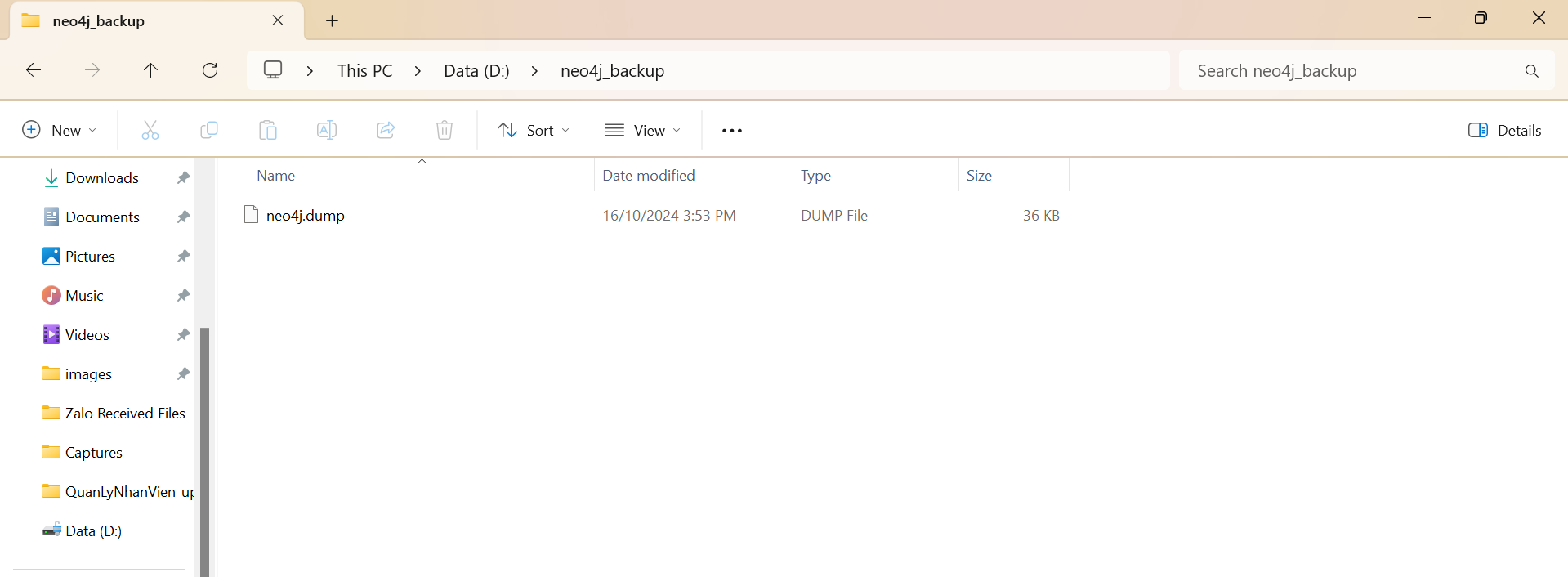
Để sao lưu cơ sở dữ liệu Neo4j, bạn có thể sử dụng lệnh sau trong terminal hoặc command prompt:

**neo4j-admin database dump neo4j --to-path="D:\neo4j\_backup" –verbose**

Giải thích các tham số:

* neo4j-admin database dump: Đây là lệnh để tạo bản sao lưu (dump) cơ sở dữ liệu.
* neo4j: Đây là tên cơ sở dữ liệu mà bạn muốn sao lưu (thông thường là neo4j).
* --to-path="D:\neo4j\_backup": Chỉ định đường dẫn nơi bạn muốn lưu bản sao lưu (file.dump).
* --verbose: Tùy chọn này sẽ cung cấp thông tin chi tiết về quá trình thực hiện.

Ví dụ về kết quả sau khi sao lưu thành công:

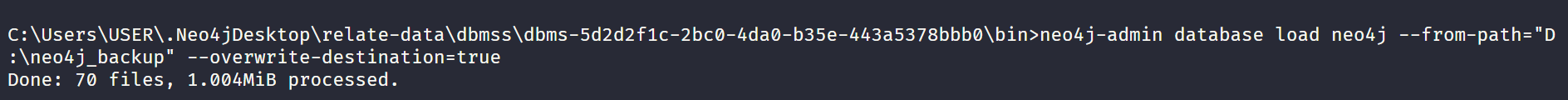
File sẽ được lưu vào:

* **Phục hồi cơ sở dữ liệu từ bản sao lưu:**

Để phục hồi cơ sở dữ liệu từ file dump, bạn có thể sử dụng lệnh sau:

neo4j-admin database load neo4j --from-path="D:\neo4j\_backup" --overwrite-destination=true

Giải thích các tham số:

* neo4j-admin database load: Đây là lệnh để nạp lại dữ liệu từ bản sao lưu vào cơ sở dữ liệu.
* neo4j: Đây là tên cơ sở dữ liệu mà bạn muốn phục hồi.
* --from-path="D:\neo4j\_backup": Chỉ định đường dẫn tới thư mục chứa file.dump để phục hồi.
* --overwrite-destination=true: Tham số này cho phép ghi đè lên cơ sở dữ liệu hiện tại (nếu có). Đây là tham số quan trọng nếu bạn muốn phục hồi cơ sở dữ liệu với cùng tên.
* Sau khi thực hiện lệnh, cơ sở dữ liệu sẽ được khôi phục từ file dump và sẵn sàng để sử dụng. Ví dụ về kết quả khi quá trình phục hồi thành công: