**Neo4j**

**Creating a Driver Instance**

**All in org.neo4j.driver.\* package**

**•Using GraphDatabase.driver() factory method call to create a driver instance.**

**A screenshot of a computer code

Description automatically generated**

**withConnectionTimeout(30, TimeUnit.SECONDS)**

Đặt thời gian timeout cho kết nối là 30 giây. Điều này có nghĩa là nếu một kết nối không thể được thiết lập trong vòng 30 giây, thì quá trình kết nối sẽ bị hủy bỏ hoặc gây ra một loại lỗi nào đó.

**withMaxConnectionPoolSize(10)**

Đặt kích thước tối đa cho bể kết nối là 10. Điều này định nghĩa số lượng kết nối tối đa được phép trong bể kết nối. Nếu số lượng kết nối vượt quá giới hạn này, thì các yêu cầu kết nối mới sẽ phải đợi cho đến khi có một kết nối khả dụng hoặc bể kết nối đủ sức chứa.

A screenshot of a computer

Description automatically generated

**neo4j**:

* + Đây là loại kết nối không mã hóa đến cơ sở dữ liệu Neo4j.
  + Thường được sử dụng khi:
    - Kết nối đến một cơ sở dữ liệu Neo4j cục bộ trên máy tính của bạn hoặc trong mạng nội bộ mà không có nhu cầu bảo mật cao.
    - Bạn chưa cấu hình mã hóa một cách rõ ràng cho cơ sở dữ liệu Neo4j.

**neo4j+s**:

* + Đây là loại kết nối được mã hóa đến cơ sở dữ liệu Neo4j.
  + Trình điều khiển sẽ kiểm tra tính xác thực của chứng chỉ, đảm bảo rằng bạn đang kết nối với một máy chủ Neo4j đúng đắn.
  + Thường được sử dụng khi:
    - Bảo mật là ưu tiên, và bạn muốn đảm bảo rằng kết nối của mình được mã hóa và an toàn.
    - Bạn có một chứng chỉ SSL (Secure Sockets Layer) hoặc TLS (Transport Layer Security) hợp lệ.

**neo4j+ssc**:

* + Đây cũng là loại kết nối được mã hóa đến cơ sở dữ liệu Neo4j.
  + Tuy nhiên, trình điều khiển không cố gắng xác minh tính xác thực của chứng chỉ.
  + Thường được sử dụng khi:
    - Bạn không cần quan tâm đến tính xác thực của chứng chỉ, ví dụ như khi sử dụng chứng chỉ tự ký hoặc trong một môi trường nơi bạn chấp nhận rủi ro về tính xác thực của máy chủ.
    - Bạn không thể xác minh chứng chỉ một cách hiệu quả, ví dụ như trong một môi trường phát triển hoặc khi làm việc trên mạng nội bộ mà không có cơ chế phức tạp để xác minh tính xác thực.

**bolt:**

* + Tạo một kết nối không mã hóa trực tiếp đến một DBMS Neo4j đơn lẻ.
  + Đây là tùy chọn khi bạn muốn kết nối đến một máy chủ Neo4j mà không yêu cầu mã hóa giao tiếp, thích hợp cho các môi trường nội bộ hoặc khi không có nhu cầu bảo mật cao.

**bolt+s:**

* + Tạo một kết nối được mã hóa trực tiếp đến một DBMS Neo4j đơn lẻ.
  + Trình điều khiển sẽ xác minh tính xác thực của chứng chỉ, đảm bảo rằng bạn đang kết nối với một máy chủ Neo4j chính xác.
  + Đây là tùy chọn an toàn hơn so với "bolt" khi bạn muốn đảm bảo rằng kết nối của mình được mã hóa và đang kết nối đúng đắn.

**bolt+ssc:**

* + Tạo một kết nối được mã hóa trực tiếp đến một DBMS Neo4j đơn lẻ, nhưng không cố gắng xác minh tính xác thực của chứng chỉ.
  + Điều này có thể hữu ích trong trường hợp bạn không quan tâm đến tính xác thực của chứng chỉ hoặc không thể xác minh chúng, như khi sử dụng chứng chỉ tự ký hoặc trong một môi trường phát triển. Tuy nhiên, điều này có thể tạo ra một lỗ hổng bảo mật nếu kẻ tấn công có thể giả mạo chứng chỉ.

**Testing connects to the Neo4j**

**Using maven :** junit-jupiter-api, junit-jupiter-engine :để test

**Sessions and Transactions**

Trong Neo4j, mỗi lần ứng dụng muốn tương tác với cơ sở dữ liệu, nó sẽ tạo ra một Session. Session là một đối tượng giao tiếp với cơ sở dữ liệu, giúp ứng dụng thực hiện các thao tác đọc và ghi dữ liệu.

Mỗi session chứa một chuỗi các giao dịch (transactions). Một giao dịch đại diện cho một loạt các thao tác đọc hoặc ghi dữ liệu được thực hiện trên cơ sở dữ liệu Neo4j.

Sessions được coi là nhẹ và có thể bị hủy khi không cần thiết nữa. Điều này cho phép ứng dụng mở và đóng các session một cách linh hoạt theo nhu cầu.

Sessions:

* Sessions là các đối tượng được tạo ra thông qua trình điều khiển (Driver) để mở kết nối tới cơ sở dữ liệu Neo4j.
* Mỗi session là một container (bộ chứa) cho một chuỗi các giao dịch (transactions).
* Sessions được xem là nhẹ và có thể bị hủy đi khi không cần thiết nữa.
* Khi một session được mở, nó sẽ mượn các kết nối từ một bể kết nối (connection pool) theo nhu cầu.
* Trình điều khiển sẽ mở nhiều kết nối TCP khi kết nối đến cơ sở dữ liệu, và những kết nối này có thể được mượn bởi các session.
* Một truy vấn có thể được gửi qua nhiều kết nối, và kết quả có thể được nhận bởi trình điều khiển qua nhiều kết nối. Điều này giúp tối ưu hóa hiệu suất và tăng khả năng mở rộng của hệ thống.

**Transactions:**

Một transaction là một loạt các thao tác đọc và ghi dữ liệu được thực hiện trên cơ sở dữ liệu Neo4j. Mỗi thao tác có thể là một truy vấn hoặc một thao tác cập nhật dữ liệu.

Transactions đảm bảo tính nhất quán của dữ liệu bằng cách đảm bảo rằng các thao tác trong giao dịch được thực hiện hoàn toàn và không bị gián đoạn bởi các thay đổi từ các giao dịch khác cùng thời điểm.

Khi một giao dịch được bắt đầu, các thao tác trong giao dịch sẽ được thực hiện và xác nhận hoàn thành khi giao dịch kết thúc. Nếu có lỗi xảy ra trong quá trình thực hiện giao dịch, nó có thể được hoàn tác (rollback) để đảm bảo tính nhất quán của dữ liệu.

🡺Sessions và Transactions là hai khái niệm quan trọng trong việc tương tác với cơ sở dữ liệu Neo4j. Sessions là các đối tượng linh hoạt được sử dụng để thực hiện các giao dịch, trong khi Transactions là các chuỗi các thao tác đọc và ghi dữ liệu được thực hiện một cách nhất quán và an toàn.

A screenshot of a computer code

Description automatically generated

1. **Transactions trong Neo4j**:
   * Một giao dịch là một đơn vị công việc được thực hiện đối với cơ sở dữ liệu Neo4j thông qua một session.
   * Mỗi giao dịch được xử lý một cách nhất quán và đáng tin cậy, độc lập với các giao dịch khác.
   * Giao dịch trong Neo4j tuân theo mô hình ACID, là viết tắt của Atomicity, Consistency, Isolation, và Durability.
2. **ACID Transactions**:
   * Mô hình ACID là một bộ các đặc điểm quan trọng của các giao dịch trong cơ sở dữ liệu, bao gồm:
     + **Atomicity (Nguyên tử):** Một giao dịch phải hoặc thực hiện hoàn toàn, hoặc không thực hiện bất kỳ thay đổi nào. Không có trạng thái "trung gian".
     + **Consistency (Nhất quán):** Dữ liệu phải ở trong trạng thái nhất quán trước và sau khi giao dịch được thực hiện. Nếu một giao dịch vi phạm các ràng buộc nhất quán, nó sẽ bị hủy và dữ liệu sẽ không bị ảnh hưởng.
     + **Isolation (Độc lập):** Mỗi giao dịch phải hoạt động độc lập với các giao dịch khác, nghĩa là một giao dịch không được ảnh hưởng bởi hoặc ảnh hưởng đến các giao dịch khác đang chạy cùng lúc.
     + **Durability (Bền vững):** Khi một giao dịch đã được hoàn thành thành công, các thay đổi dữ liệu phải được lưu trữ và không bị mất trong trường hợp xảy ra lỗi hệ thống hoặc sự cố.
3. **Quen thuộc với ACID Transactions**:
   * Nhiều nhà phát triển đã quen thuộc với mô hình ACID từ công việc của họ với cơ sở dữ liệu quan hệ, vì vậy mô hình ACID đã trở thành chuẩn trong một thời gian dài.

Tóm lại, Transactions trong Neo4j là các đơn vị công việc được thực hiện một cách nhất quán và đáng tin cậy, độc lập với các giao dịch khác, và tuân theo mô hình ACID để đảm bảo tính toàn vẹn và nhất quán của dữ liệu.

**Result**

1. Đối tượng Result:
   * Khi bạn thực hiện một truy vấn trong Neo4j thông qua trình điều khiển (Driver), kết quả của truy vấn sẽ được trả về dưới dạng một đối tượng gọi là Result.
   * Đối tượng Result này chứa tất cả các dữ liệu bạn nhận được từ truy vấn, cũng như các thông tin bổ sung về cách thức thực thi truy vấn và kết quả.
2. Bản ghi (Record):
   * Mỗi kết quả từ truy vấn được gọi là một Bản ghi (Record).
   * Một Bản ghi chứa các giá trị tương ứng với các cột mà bạn đã chỉ định trong phần RETURN của câu lệnh truy vấn.
   * Ví dụ, nếu bạn truy vấn các người dùng từ cơ sở dữ liệu, mỗi Bản ghi có thể chứa thông tin về một người dùng như tên, tuổi, địa chỉ, v.v.
3. Cách truy cập vào Bản ghi:
   * Bạn có thể truy cập vào mỗi Bản ghi từ đối tượng Result thông qua các cách sau:
     + Sử dụng một vòng lặp để duyệt qua từng Bản ghi một cách tuần tự.
     + Sử dụng phương thức stream() để lấy một luồng dữ liệu các Bản ghi.
     + Sử dụng phương thức list() để lấy một danh sách các Bản ghi.
     + Sử dụng phương thức single() để lấy một Bản ghi duy nhất, nhưng lưu ý rằng nó chỉ hoạt động khi kết quả truy vấn có một Bản ghi duy nhất.
4. Siêu dữ liệu về kết quả:
   * Ngoài các Bản ghi, đối tượng Result còn chứa các thông tin bổ sung về kết quả và truy vấn.
   * Các thông tin này có thể bao gồm thời gian thực thi của truy vấn, số lượng Bản ghi trả về, hoặc các thông tin khác liên quan đến quá trình thực thi truy vấn.

Tóm lại, đối tượng Result là nơi chứa tất cả kết quả từ một truy vấn trong Neo4j, bao gồm các Bản ghi và các thông tin bổ sung. Bằng cách truy cập vào đối tượng Result, bạn có thể lấy được dữ liệu kết quả và các thông tin khác liên quan.