This repository Search Explore Features Enterprise Blog Sign up Sign in

neo4j / **neo4j** Watch 238 Star 1,885 Fork 730

Something wrong to create nodes and relations

New issue

Closed **tdppro** opened this issue Apr 4, 2015 · 4 comments

tdppro commented Apr 4, 2015

I have five classes as below:

```
@NodeEntity
public class Person {

    @GraphId
    Long id;

    public String name;

    public String sex;

    public Person() {
    }

    public Person(String name, String sex) {
        this.name = name;
        this.sex = sex;
    }

    @RelatedTo(type = "RATED", direction = Direction.OUTGOING)
    @Fetch
    public Set<Movie> ratedMovies;

    @RelatedToVia(type = "RATED", direction = Direction.OUTGOING)
    @Fetch
    public Set<Rating> ratings;

    public Rating rateMovie(Movie movie, int rate) {
        Rating rating = new Rating(rate, this, movie);
        ratings.add(rating);
        return rating;
    }

    public Long getId() {
        return id;
    }

    public void setId(Long id) {
        this.id = id;
    }

    public String getName() {
        return name;
    }

    public void setName(String name) {
        this.name = name;
    }

    public String getSex() {
        return sex;
    }

    public void setSex(String sex) {
        this.sex = sex;
    }

    public Set<Movie> getRatedMovies() {
        return ratedMovies;
    }

    public void setRatedMovies(Set<Movie> ratedMovies) {
        this.ratedMovies = ratedMovies;
    }

    @Override
```

Labels

None yet

Milestone

No milestone

Assignee

No one assigned

2 participants

```
    @Override
    public int hashCode() {
        final int prime = 31;
        int result = 1;
        result = prime * result + ((id == null) ? 0 : id.hashCode());
        result = prime * result + ((name == null) ? 0 : name.hashCode());
        result = prime * result + ((sex == null) ? 0 : sex.hashCode());
        return result;
    }

    @Override
    public boolean equals(Object obj) {
        if (this == obj)
            return true;
        if (obj == null)
            return false;
        if (getClass() != obj.getClass())
            return false;
        Person other = (Person) obj;
        if (id == null) {
            if (other.id != null)
                return false;
        } else if (!id.equals(other.id))
            return false;
        if (name == null) {
            if (other.name != null)
                return false;
        } else if (!name.equals(other.name))
            return false;
        if (sex == null) {
            if (other.sex != null)
                return false;
        } else if (!sex.equals(other.sex))
            return false;
        return true;
    }
}
```

```
public interface PersonRepository extends CrudRepository<Person, String> {
    Person findByName(String name);
}
```

```
@NodeEntity
public class Movie {

    @GraphId
    private Long id;

    private String title;

    private String releaseDate;

    @RelatedTo(type = "RATED", direction = Direction.INCOMING)
    @Fetch
    public Set<Person> ratedPerson;

    @RelatedToVia(type = "RATED", direction = Direction.INCOMING)
    @Fetch
    public Set<Rating> ratings;

    public Movie() {
    }

    public Movie(String title, String releaseDate) {
        this.title = title;
        this.releaseDate = releaseDate;
    }

    public Rating addRate(Person person, int rate) {
        Rating rating = new Rating(rate, person, this);
        ratings.add(rating);
        return rating;
    }

    public Long getId() {
        return id;
    }

    public void setId(Long id) {
        this.id = id;
    }

    public String getTitle() {
```

```
    public String getTitle() {
        return title;
    }

    public void setTitle(String title) {
        this.title = title;
    }

    public String getReleaseDate() {
        return releaseDate;
    }

    public void setReleaseDate(String releaseDate) {
        this.releaseDate = releaseDate;
    }

    public Set<Person> getRatedPerson() {
        return ratedPerson;
    }

    public void setRatedPerson(Set<Person> ratedPerson) {
        this.ratedPerson = ratedPerson;
    }

    @Override
    public int hashCode() {
        final int prime = 31;
        int result = 1;
        result = prime * result + ((id == null) ? 0 : id.hashCode());
        result = prime * result
                + ((ratedPerson == null) ? 0 : ratedPerson.hashCode());
        result = prime * result
                + ((releaseDate == null) ? 0 : releaseDate.hashCode());
        result = prime * result + ((title == null) ? 0 : title.hashCode());
        return result;
    }

    @Override
    public boolean equals(Object obj) {
        if (this == obj)
            return true;
        if (obj == null)
            return false;
        if (getClass() != obj.getClass())
            return false;
        Movie other = (Movie) obj;
        if (id == null) {
            if (other.id != null)
                return false;
        } else if (!id.equals(other.id))
            return false;
        if (ratedPerson == null) {
            if (other.ratedPerson != null)
                return false;
        } else if (!ratedPerson.equals(other.ratedPerson))
            return false;
        if (releaseDate == null) {
            if (other.releaseDate != null)
                return false;
        } else if (!releaseDate.equals(other.releaseDate))
            return false;
        if (title == null) {
            if (other.title != null)
                return false;
        } else if (!title.equals(other.title))
            return false;
        return true;
    }
}
```

```
public interface MovieRepository extends CrudRepository<Movie, String> {
}
```

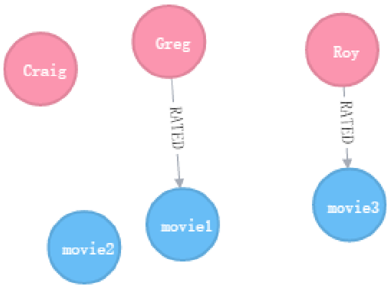
```
@RelationshipEntity(type="RATED")
public class Rating {

    @GraphId
    private Long id;

    private int rate;

    @StartNode
    private Person person;
    @EndNode
    private Movie movie;

    public Rating() {
    }


    public Rating(int rate, Person person, Movie movie) {
        super();
        this.rate = rate;
        this.person = person;
        this.movie = movie;
    }


    public Long getId() {
        return id;
    }

    public void setId(Long id) {
        this.id = id;
    }

    public int getRate() {
        return rate;
    }

    public void setRate(int rate) {
        this.rate = rate;
    }

    public Person getPerson() {
        return person;
    }

    public void setPerson(Person person) {
        this.person = person;
    }

    public Movie getMovie() {
        return movie;
    }

    public void setMovie(Movie movie) {
        this.movie = movie;
    }

    @Override
    public int hashCode() {
        final int prime = 31;
        int result = 1;
        result = prime * result + ((id == null) ? 0 : id.hashCode());
        result = prime * result + rate;
        return result;
    }

    @Override
    public boolean equals(Object obj) {
        if (this == obj)
            return true;
        if (obj == null)
            return false;
        if (getClass() != obj.getClass())
            return false;
        Rating other = (Rating) obj;
        if (id == null) {
            if (other.id != null)
                return false;
        } else if (!id.equals(other.id))
            return false;
        if (rate != other.rate)
            return false;
        return true;
    }
}
```

```
@SpringBootApplication
public class Application implements CommandLineRunner {

    @Configuration
    @EnableNeo4jRepositories(basePackages = "com.tdp.apps.spring_neo4j_demo")
    static class ApplicationConfig extends Neo4jConfiguration {

        public ApplicationConfig() {
            setBasePackage("com.tdp.apps.spring_neo4j_demo");
        }

        @Bean
        GraphDatabaseService graphDatabaseService() {
            return new GraphDatabaseFactory()
                    .newEmbeddedDatabase("accessingdataneo4j.db");
        }
    }

    @Autowired
    PersonRepository personRepository;
    @Autowired
    MovieRepository movieRepository;

    @Autowired
    GraphDatabase graphDatabase;

    public void run(String... args) throws Exception {

        Person greg = new Person("Greg", "male");
        Person roy = new Person("Roy", "female");
        Person craig = new Person("Craig", "male");

        Movie movie1 = new Movie("movie1", "1997");
        Movie movie2 = new Movie("movie2", "1998");
        Movie movie3 = new Movie("movie3", "1999");

        Transaction tx = graphDatabase.beginTx();
        try {
            greg = personRepository.save(greg);
            roy = personRepository.save(roy);
            craig = personRepository.save(craig);

            movie1 = movieRepository.save(movie1);
            movie2 = movieRepository.save(movie2);
            movie3 = movieRepository.save(movie3);

            greg.rateMovie(movie1, 3);
            greg.rateMovie(movie2, 4);
            greg = personRepository.save(greg);

            roy.rateMovie(movie2, 5);
            roy.rateMovie(movie3, 2);
            roy = personRepository.save(roy);

            movie2.addRate(craig, 1);
            movie2 = movieRepository.save(movie2);
            tx.success();
        } finally {
            tx.close();
        }
    }

    public static void main(String[] args) throws Exception {
        FileUtils.deleteRecursively(new File("accessingdataneo4j.db"));

        SpringApplication.run(Application.class, args);
    }
}
```

I want to build the relations greg->movie1 grep->movie2 roy->movie2 roy->movie3 and build the relation from moive2 to craig. So I use the code below:

```
try {
            greg = personRepository.save(greg);
            roy = personRepository.save(roy);
            craig = personRepository.save(craig);

            movie1 = movieRepository.save(movie1);
            movie2 = movieRepository.save(movie2);
            movie3 = movieRepository.save(movie3);

            greg.rateMovie(movie1, 3);
            greg.rateMovie(movie2, 4);
            greg = personRepository.save(greg);

            roy.rateMovie(movie2, 5);
            roy.rateMovie(movie3, 2);
            roy = personRepository.save(roy);

            movie2.addRate(craig, 1);
            movie2 = movieRepository.save(movie2);
            tx.success();
        } finally {
            tx.close();
        }
```

But the result shows as below



The people have no relations with movie2 . I don't know why . Can you help me to handle it ?

jexp commented Apr 4, 2015 Collaborator

This is not a support forum but github issues, for reporting issues with Neo4j.

Please post your question to Stackoverflow.
That said, your equals method for Rating is wrong if you have two new Ratings (id=null) they will eliminate each other in a Set.

jexp closed this Apr 4, 2015

tdppro commented Apr 5, 2015

I am not asking support, I rewrite the hashCode and equals method of the Person, Movie, Rating entity.
But the problem stills exist.
So I think its some bugs in sdn.

The new Person, Movie, Rating are as belows.

```
@NodeEntity
public class Person {

    @GraphId
    Long id;

    private String name;

    private String sex;

    @RelatedTo(type = "RATED", direction = Direction.OUTGOING)
    @Fetch
    private Set<Movie> ratedMovies;

    @RelatedToVia(type = "RATED", direction = Direction.OUTGOING)
    @Fetch
    private Set<Rating> ratings;
    public Person() {
    }

    public Person(String name, String sex) {
        this.name = name;
        this.sex = sex;
    }

    public Rating rateMovie(Movie movie, int rate) {
        Rating rating = new Rating(rate, this, movie);
        ratings.add(rating);
        return rating;
    }

    public Long getId() {
        return id;
    }

    public void setId(Long id) {
        this.id = id;
    }

    public String getName() {
        return name;
    }

    public void setName(String name) {
        this.name = name;
    }

    public String getSex() {
        return sex;
    }

    public void setSex(String sex) {
        this.sex = sex;
    }

    public Set<Movie> getRatedMovies() {
        return ratedMovies;
    }

    public void setRatedMovies(Set<Movie> ratedMovies) {
        this.ratedMovies = ratedMovies;
    }

    public Set<Rating> getRatings() {
        return ratings;
    }

    public void setRatings(Set<Rating> ratings) {
        this.ratings = ratings;
    }

    @Override
    public int hashCode() {
        final int prime = 31;
        int result = 1;
        result = prime * result + ((name == null) ? 0 : name.hashCode());
        result = prime * result + ((sex == null) ? 0 : sex.hashCode());
        return result;
    }

    @Override
    public boolean equals(Object obj) {
        if (this == obj)
            return true;
        if (obj == null)
            return false;
        if (getClass() != obj.getClass())
            return false;
        Person other = (Person) obj;
        if (name == null) {
            if (other.name != null)
                return false;
        } else if (!name.equals(other.name))
            return false;
        if (sex == null) {
            if (other.sex != null)
                return false;
        } else if (!sex.equals(other.sex))
            return false;
        return true;
    }
```

```java
}

@NodeEntity
public class Movie {

    @GraphId
    private Long id;

    private String title;

    private String releaseDate;

    @RelatedTo(type = "RATED", direction = Direction.INCOMING)
    @Fetch
    private Set<Person> ratedPerson;

    @RelatedToVia(type = "RATED", direction = Direction.INCOMING)
    @Fetch
    private Set<Rating> ratings;

    public Movie() {
    }

    public Movie(String title, String releaseDate) {
        this.title = title;
        this.releaseDate = releaseDate;
    }

    public Rating addRate(Person person, int rate) {
        Rating rating = new Rating(rate, person, this);
        ratings.add(rating);
        return rating;
    }

    public Long getId() {
        return id;
    }

    public void setId(Long id) {
        this.id = id;
    }

    public String getTitle() {
        return title;
    }

    public void setTitle(String title) {
        this.title = title;
    }

    public String getReleaseDate() {
        return releaseDate;
    }

    public void setReleaseDate(String releaseDate) {
        this.releaseDate = releaseDate;
    }

    public Set<Person> getRatedPerson() {
        return ratedPerson;
    }

    public void setRatedPerson(Set<Person> ratedPerson) {
        this.ratedPerson = ratedPerson;
    }

    public Set<Rating> getRatings() {
        return ratings;
    }

    public void setRatings(Set<Rating> ratings) {
        this.ratings = ratings;
    }

    @Override
    public int hashCode() {
        final int prime = 31;
        int result = 1;
        result = prime * result
                + ((releaseDate == null) ? 0 : releaseDate.hashCode());
        result = prime * result + ((title == null) ? 0 : title.hashCode());
        return result;
    }

    @Override
    public boolean equals(Object obj) {
```

```
    public boolean equals(Object obj) {
        if (this == obj)
            return true;
        if (obj == null)
            return false;
        if (getClass() != obj.getClass())
            return false;
        Movie other = (Movie) obj;
        if (releaseDate == null) {
            if (other.releaseDate != null)
                return false;
        } else if (!releaseDate.equals(other.releaseDate))
            return false;
        if (title == null) {
            if (other.title != null)
                return false;
        } else if (!title.equals(other.title))
            return false;
        return true;
    }



}

@RelationshipEntity(type="RATED")
public class Rating {

    @GraphId
    private Long id;

    private int rate;

    @StartNode
    private Person person;

    @EndNode
    private Movie movie;

    public Rating() {
    }


    public Rating(int rate, Person person, Movie movie) {
        super();
        this.rate = rate;
        this.person = person;
        this.movie = movie;
    }


    public Long getId() {
        return id;
    }

    public void setId(Long id) {
        this.id = id;
    }

    public int getRate() {
        return rate;
    }

    public void setRate(int rate) {
        this.rate = rate;
    }

    public Person getPerson() {
        return person;
    }

    public void setPerson(Person person) {
        this.person = person;
    }

    public Movie getMovie() {
        return movie;
    }

    public void setMovie(Movie movie) {
        this.movie = movie;
    }


    @Override
    public int hashCode() {
        final int prime = 31;
        final int prime = 31;
        int result = 1;
        result = prime * result + ((movie == null) ? 0 : movie.hashCode());
        result = prime * result + ((person == null) ? 0 : person.hashCode());
        result = prime * result + rate;
        return result;
    }


    @Override
    public boolean equals(Object obj) {
        if (this == obj)
            return true;
        if (obj == null)
            return false;
        if (getClass() != obj.getClass())
            return false;
        Rating other = (Rating) obj;
        if (movie == null) {
            if (other.movie != null)
                return false;
        } else if (!movie.equals(other.movie))
            return false;
        if (person == null) {
            if (other.person != null)
                return false;
        } else if (!person.equals(other.person))
            return false;
        if (rate != other.rate)
            return false;
        return true;
    }

}
```

jexp commented Apr 5, 2015 Collaborator

Please write a test that asserts that your objects behave correctly in a Set.

And this is the Neo4j Repository, not SDN, if you want to raise he issue against SDN please go to: https://jira.spring.io/browse/DATAGRAPH

tdppro commented Apr 5, 2015

Sorry for my mistake. I mixed the neo4j and spring-data-neo4j project!