Hibernate One-To-Many Övningar

Sammanfattning

I den tidigare övningen använde vi @ManyToOne-annoteringen för att visa relationen mellan Student och Tutor. En student har endast en tutor, men en tutor kan undervisa flera studenter.

I denna övning kommer vi att:

- Vända på relationen och skapa en One-To-Many-relation i Tutor-klassen.
- Använda Set för att hantera studenter som en grupp.
- Lägga till, hämta och uppdatera studenter i gruppen.
- Använda @JoinColumn för att skapa en foreign key.
- Byta ut Set mot Map och testa olika metoder för att hantera data.

Steg 1: Uppdatera Student-klassen

- Ta bort Tutor-fältet och anteckningen many-to-one och allt som har med Tutor att göra i Student klass (getters,setters, osv)
- Säkerställ att studenten inte längre har en referens till en enskild tutor.

Steg 2: Skapa One-To-Many-relationen

- I Tutor-klassen, skapa en samling av studenter:
    - `Set<Student>teachingGroup`

- Använd @OneToMany-annoteringen ovanför samlingen.

- Lägg till denna egenskap (teachingGroup) i din konstruktor och initiera den som ett HashSet<Student>()

- Implementera en metod för att lägga till studenter till tutorns grupp (addStudentToTeachingGroup)
- Skapa en metod för att hämta alla studenter i tutorns grupp (getTeachingGroup)

Steg 3: Skapa och testa relationen

- I huvudmetoden, skapa en Tutor och flera Student-objekt.
- Tilldela studenter till tutorns teachingGroup.
- Spara studenter och tutor i databasen.
    - I ij>
        - Droppa tabellerna student och tutor.
    - Kör om koden.
    - Verifiera att tre tabeller nu finns: student, tutor och tutor_student (en länktabell).
        - show tables
        - select * from student;
        - describe student;
    - Observera att det inte finns någon foreign key i student-tabellen, men tutor_student innehåller primärnycklar från både student och tutor.
- Hämta studenter från en tutors grupp:
    - Kommentera bort all kod från att skapa en tutor till slutet av loopen i huvudmetoden.
    - Hämta en tutor från databasen genom att använda session.get() och ange tutor-ID.
    - Hämta alla studenter i tutorns teachingGroup genom att anropa en metod i Tutor-klassen.
    - Loopa igenom studenterna och skriv ut deras information.

Steg 4: Lägg till en ny student till gruppen

- Skapa en ny student och spara den i databasen.
- Koppla den nya studenten till en befintlig tutor.
- Kör koden och kontrollera i databasen att studenten nu är en del av tutorns grupp.

Steg 5: Använda @JoinColumn för en Foreign Key

- I Tutor-klassen, lägg till @JoinColumn(name="TUTOR_FK") ovanför studentlistan.
- Droppa befintliga tabeller.
- Lägg till en ny tutor och några studenter Koppla dem genom att lägga till studenter till tutorns teachingGroup och kör om koden.
- Kontrollera i databasen att student-tabellen nu har en foreign key till tutor.

Steg 6: Använda Map istället för Set

OBS. Ha en kopia av din Set-kod någonstans, eftersom vi skulle gå tillbaka till Set efter att ha gjort map och list.

- I Tutor-klassen, ändra typen av studentlistan från Set till Map<String, Student>.
- I Student-klassen, lägg till enrollmentID i konstruktorn.
- Använd @MapKey(name="enrollmentID") ovanför Map-attributet, under @OneToMany
- Uppdatera metoderna för att hantera Map-strukturen.
- Spara data och hämta en specifik student genom att använda enrollmentID som nyckel.

Steg 7: Testa att hämta data från databasen

- Ändra studenter i mainmetoden.
    - Använd denna konstruktor: *public Student(String name, String enrollmentID)*
- Droppa befintliga tabeller.
- Kör koden och verifiera att du har data i tabeller.
- Hämta studenter till en specifik tutor:
    - Kommentera bort all kod om att skapa en tutor och studenter i huvudmetoden.
    - Hämta en tutor från databasen genom att använda session.get() och ange ett giltigt tutor-ID.
    - Hämta alla studenter i tutorns teachingGroup genom att anropa getTeachingGroup().
    - Loopa igenom studenterna och skriv ut deras information.