Hibernate
Many_To_One Relationship

Overview

This chapter covers:

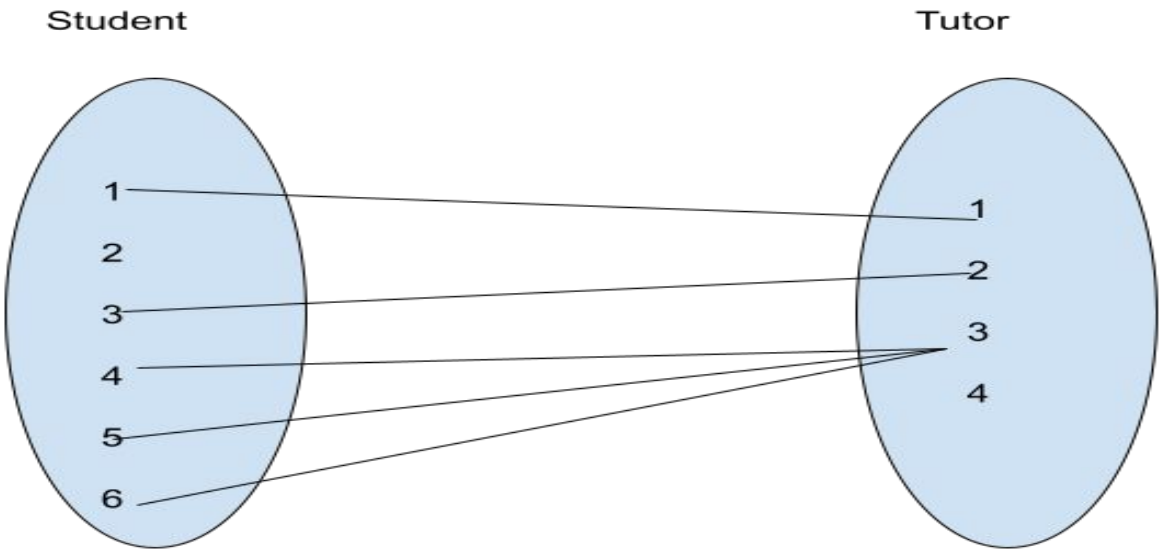
- Understanding Many-to-One relationships in Hibernate.
- Implementing @ManyToOne annotation.
- Handling foreign keys
- The extra methods that we need

Understanding Many-to-One Relationships

In database design, a Many-to-One relationship means:

- Multiple child entities are associated with one parent entity.
- A foreign key in the child table references the parent table.

Many-To-One Relationship



Example Scenario

- A Tutor teaches multiple Students.
- Each Student has only one Tutor.
- This is a Many-to-One relationship from Student → Tutor.

Defining a Many-to-One Relationship in Hibernate

To model this relationship, we annotate the Student class with @ManyToOne.

```
@ManyToOne

private Tutor tutor;
```

@ManyToOne → Defines a Many-to-One relationship with the Tutor class and works here as a foreign key.

What happens in the student table

In the student table, a column tutor_id is created as the foreign key to the tutor table.

Changing the name of the foreign key column

We can change the name of this column in our code:

```
@ManyToOne
@JoinColumn(name="TUTOR_FK")
private Tutor tutor;
```

@JoinColumn(name = "TUTOR_FK") → Specifies the foreign key column in the Student table.

How Hibernate Maps This Relationship

When Hibernate generates the database schema:

- The Student table will have a foreign key column TUTOR_FK referencing the Tutor table.
- The Tutor table remains unchanged (does not contain student information).

Generated Database Schema:

```
CREATE TABLE Student (
    id BIGINT AUTO_INCREMENT PRIMARY KEY,
    name VARCHAR(255),
    TUTOR_FK BIGINT,
    FOREIGN KEY (TUTOR_FK) REFERENCES Tutor(id)
);
```

How does the student table look

```
ij> describe student;
COLUMN_NAME         |TYPE_NAME|DEC&|NUM&|COLUM&|COLUMN_DEF|CHAR_OCTE&|IS_NULL&
------------------------------------------------------------------------------
ID                  |INTEGER  |0   |10  |10    |NULL      |NULL      |NO
ENROLLMENTID        |VARCHAR  |NULL|NULL|255   |NULL      |510       |YES
NAME                |VARCHAR  |NULL|NULL|255   |NULL      |510       |YES
NUM_COURSES         |INTEGER  |0   |10  |10    |NULL      |NULL      |YES
TUTOR_FK            |INTEGER  |0   |10  |10    |NULL      |NULL      |YES

5 rows selected
```

Allocate a tutor to a student

The method *allocateTutor* in the Student class allocates a tutor to a student.

```
public void allocateTutor(Tutor tutor) {
    this.tutor=tutor;
}
```

getTutorName

There is another method in the Student class to get the name of the tutor.

```
public String getTutorName() {
    return this.tutor.getName();
```

Configuration in hibernate.cfg.xml

Tutor class should be add to the hibernate.cfg.xml file:

```
<mapping class="se.yrgo.domain.Tutor"/>
```

Persisting Data in a Many-to-One Relationship

To correctly save a Student and Tutor, follow this order:

```
Transaction tx = session.beginTransaction();
Tutor tutor = new Tutor("ABC123" ,"Edward", 30000);
Student student = new Student("Sara Hedborn");
student.allocateTutor(tutor);

session.save(student);
session.save(tutor);

tx.commit();
```

Comes soon

In the next chapter we will explore:

- One to many relation