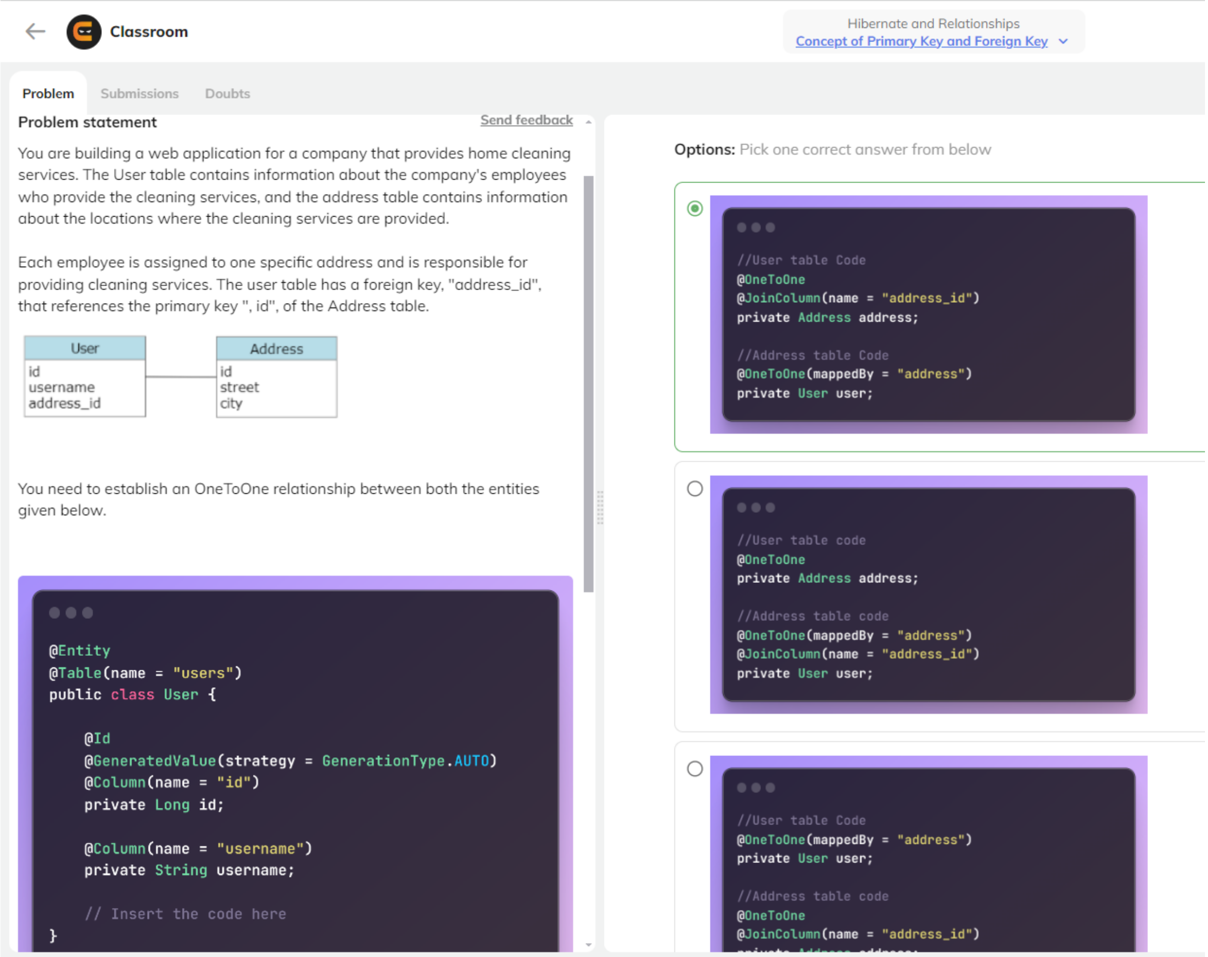
Quiz Questions

(1) Concept of Primary Key and Foreign Key


Classroom
Hibernate and Relationships
Concept of Primary Key and Foreign Key
Problem
Submissions
Doubts
Send feedback

Problem statement

You are building a web application for a company that provides home cleaning services. The User table contains information about the company's employees who provide the cleaning services, and the address table contains information about the locations where the cleaning services are provided.

Each employee is assigned to one specific address and is responsible for providing cleaning services. The user table has a foreign key, "address_id", that references the primary key ", id", of the Address table.

User	Address
id username address_id	id street city

You need to establish an OneToOne relationship between both the entities given below.

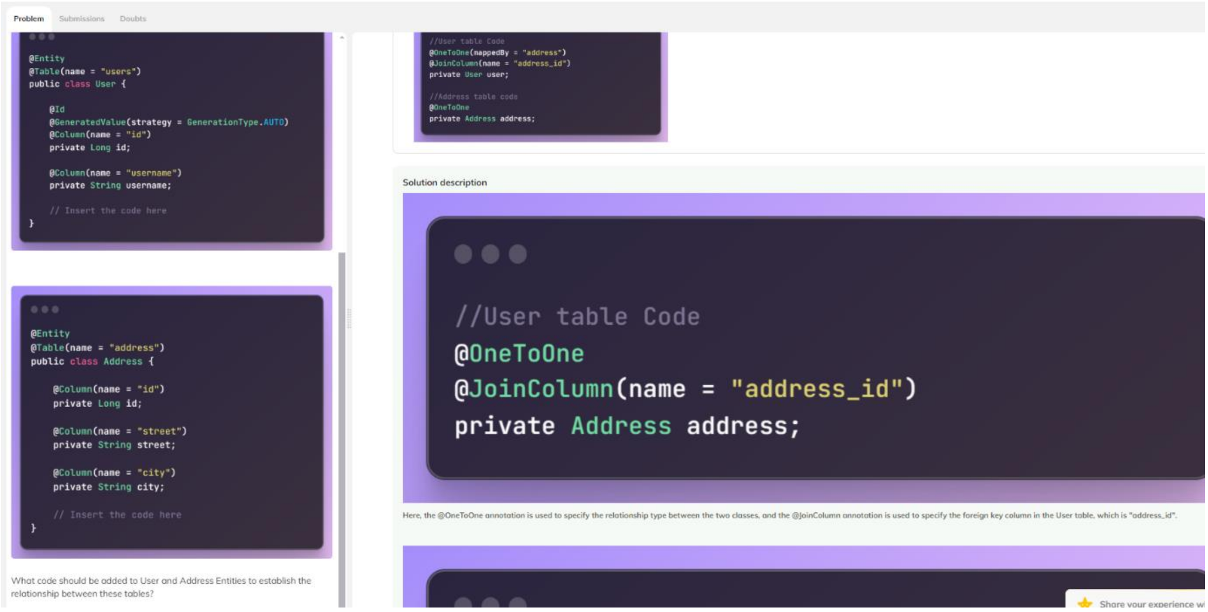
```
@Entity
@Table(name = "users")
public class User {

    @Id
    @GeneratedValue(strategy = GenerationType.AUTO)
    @Column(name = "id")
    private Long id;

    @Column(name = "username")
    private String username;

    // Insert the code here
}
```

Options: Pick one correct answer from below

●
```
//User table Code
@OneToOne
@JoinColumn(name = "address_id")
private Address address;

//Address table Code
@OneToOne(mappedBy = "address")
private User user;
```

○
```
//User table code
@OneToOne
private Address address;

//Address table code
@OneToOne(mappedBy = "address")
@JoinColumn(name = "address_id")
private User user;
```

○
```
//User table Code
@OneToOne(mappedBy = "address")
private User user;

//Address table code
@OneToOne
@JoinColumn(name = "address_id")
private Address address;
```



```
@Entity
@Table(name = "users")
public class User {

    @Id
    @GeneratedValue(strategy = GenerationType.AUTO)
    @Column(name = "id")
    private Long id;

    @Column(name = "username")
    private String username;

    // Insert the code here
}
```

```
@Entity
@Table(name = "address")
public class Address {

    @Column(name = "id")
    private Long id;

    @Column(name = "street")
    private String street;

    @Column(name = "city")
    private String city;

    // Insert the code here
}
```

What code should be added to User and Address Entities to establish the relationship between these tables?

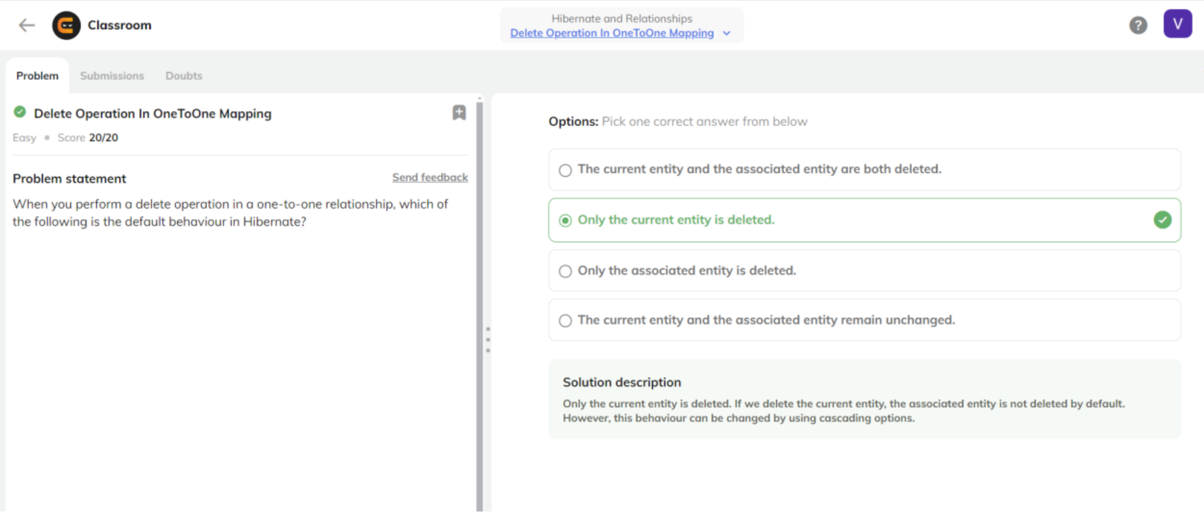
```
//User table Code
@OneToOne(mappedBy = "address")
@JoinColumn(name = "address_id")
private User user;

//Address table code
@OneToOne
private Address address;
```

Solution description

```
//User table Code
@OneToOne
@JoinColumn(name = "address_id")
private Address address;
```

Here, the @OneToOne annotation is used to specify the relationship type between the two classes, and the @JoinColumn annotation is used to specify the foreign key column in the User table, which is "address_id".

(2) Delete Operations in OneToOne Mapping


Classroom
Hibernate and Relationships
Delete Operation In OneToOne Mapping
V
Problem
Submissions
Doubts
Send feedback

✓ Delete Operation In OneToOne Mapping

Easy • Score 20/20

Problem statement

When you perform a delete operation in a one-to-one relationship, which of the following is the default behaviour in Hibernate?

Options: Pick one correct answer from below

○ The current entity and the associated entity are both deleted.

● Only the current entity is deleted. ✓

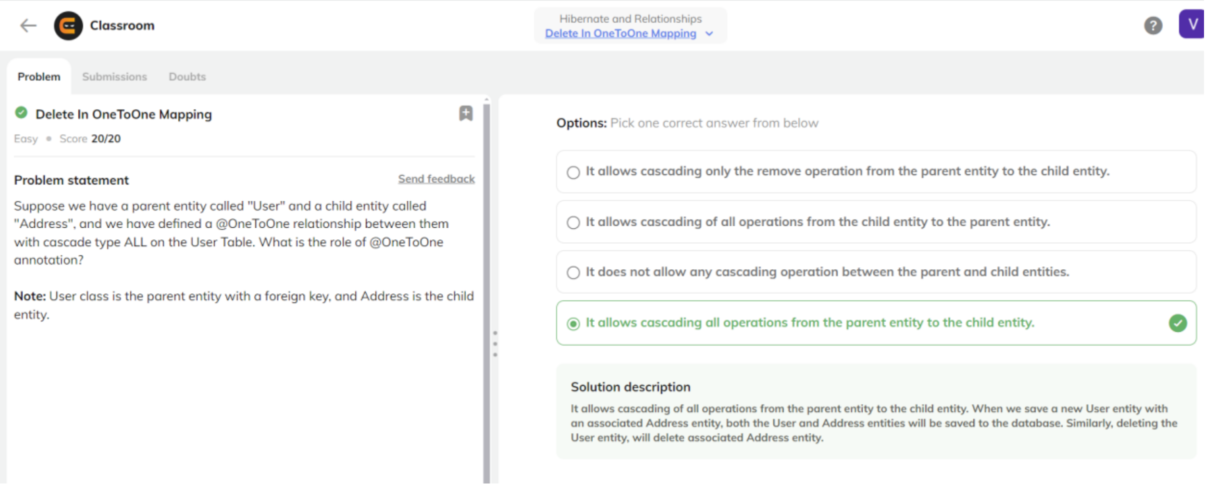
○ Only the associated entity is deleted.

○ The current entity and the associated entity remain unchanged.

Solution description

Only the current entity is deleted. If we delete the current entity, the associated entity is not deleted by default. However, this behaviour can be changed by using cascading options.

(3) Delete in One To One Mapping



✓ Delete In OneToOne Mapping

Easy • Score 20/20

Problem statement

Suppose we have a parent entity called "User" and a child entity called "Address", and we have defined a @OneToOne relationship between them with cascade type ALL on the User Table. What is the role of @OneToOne annotation?

Note: User class is the parent entity with a foreign key, and Address is the child entity.

Options: Pick one correct answer from below

○ It allows cascading only the remove operation from the parent entity to the child entity.

○ It allows cascading of all operations from the child entity to the parent entity.

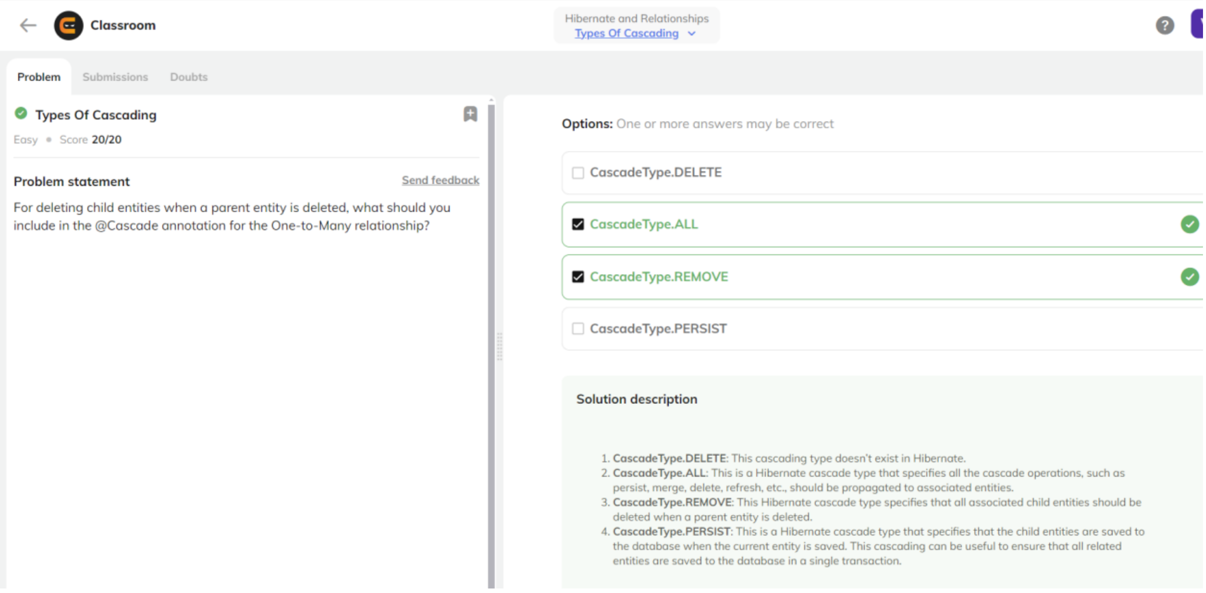
○ It does not allow any cascading operation between the parent and child entities.

● It allows cascading all operations from the parent entity to the child entity. ✓

Solution description

It allows cascading of all operations from the parent entity to the child entity. When we save a new User entity with an associated Address entity, both the User and Address entities will be saved to the database. Similarly, deleting the User entity, will delete associated Address entity.

(4) Types of Cascading



✓ Types Of Cascading

Easy • Score 20/20

Problem statement

For deleting child entities when a parent entity is deleted, what should you include in the @Cascade annotation for the One-to-Many relationship?

Options: One or more answers may be correct

□ CascadeType.DELETE

☑ CascadeType.ALL ✓

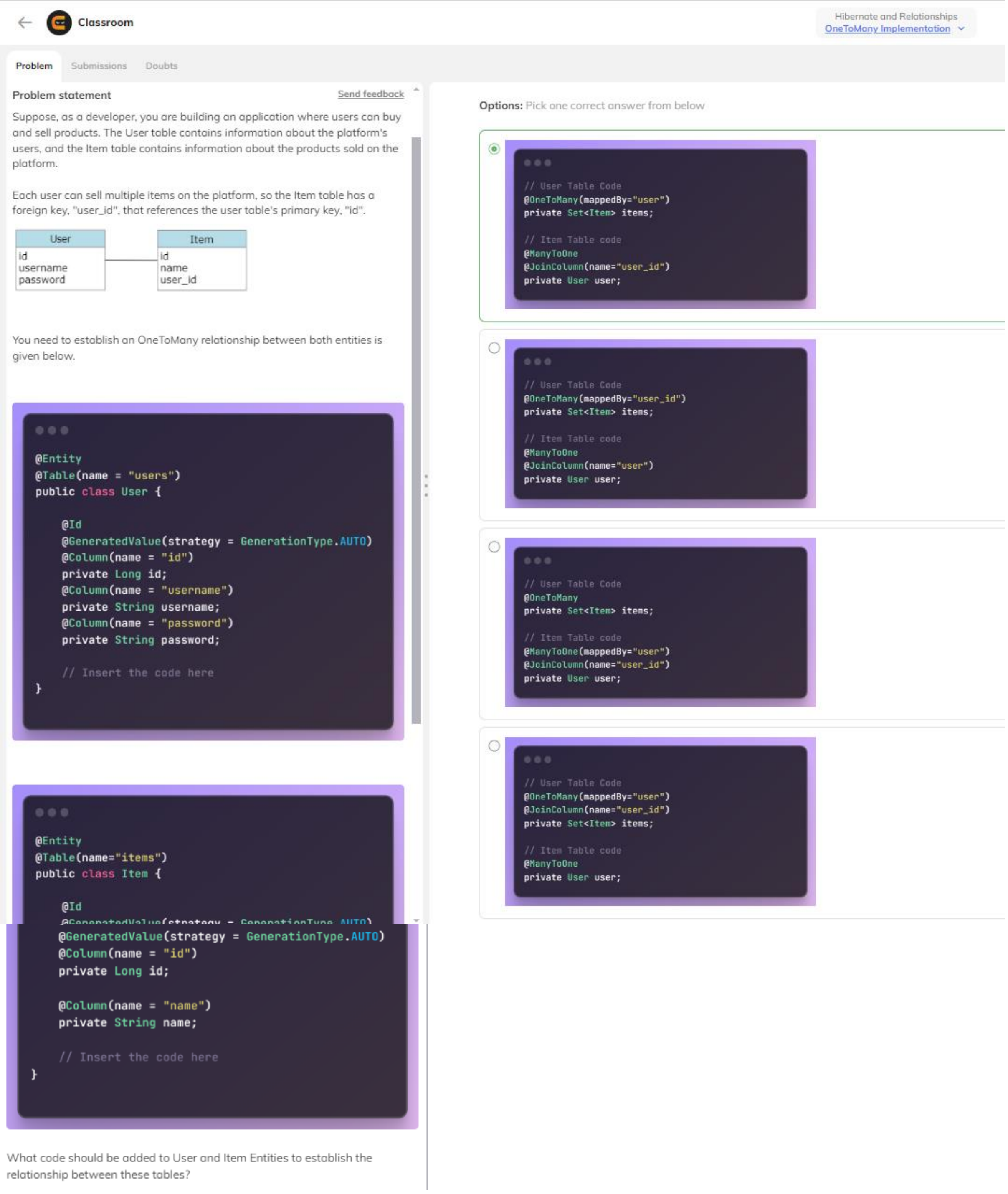
☑ CascadeType.REMOVE ✓

□ CascadeType.PERSIST

Solution description

1. **CascadeType.DELETE**: This cascading type doesn't exist in Hibernate.
2. **CascadeType.ALL**: This is a Hibernate cascade type that specifies all the cascade operations, such as persist, merge, delete, refresh, etc., should be propagated to associated entities.
3. **CascadeType.REMOVE**: This Hibernate cascade type specifies that all associated child entities should be deleted when a parent entity is deleted.
4. **CascadeType.PERSIST**: This is a Hibernate cascade type that specifies that the child entities are saved to the database when the current entity is saved. This cascading can be useful to ensure that all related entities are saved to the database in a single transaction.

(5) One To Many Implementation



Problem statement

Suppose, as a developer, you are building an application where users can buy and sell products. The User table contains information about the platform's users, and the Item table contains information about the products sold on the platform.

Each user can sell multiple items on the platform, so the Item table has a foreign key, "user_id", that references the user table's primary key, "id".

User
id username password

Item
id name user_id

You need to establish an OneToMany relationship between both entities is given below.

```
@Entity
@Table(name = "users")
public class User {

    @Id
    @GeneratedValue(strategy = GenerationType.AUTO)
    @Column(name = "id")
    private Long id;
    @Column(name = "username")
    private String username;
    @Column(name = "password")
    private String password;

    // Insert the code here
}
```

```
@Entity
@Table(name="items")
public class Item {

    @Id
    @GeneratedValue(strategy = GenerationType.AUTO)
    @Column(name = "id")
    private Long id;

    @Column(name = "name")
    private String name;

    // Insert the code here
}
```

What code should be added to User and Item Entities to establish the relationship between these tables?

Options: Pick one correct answer from below

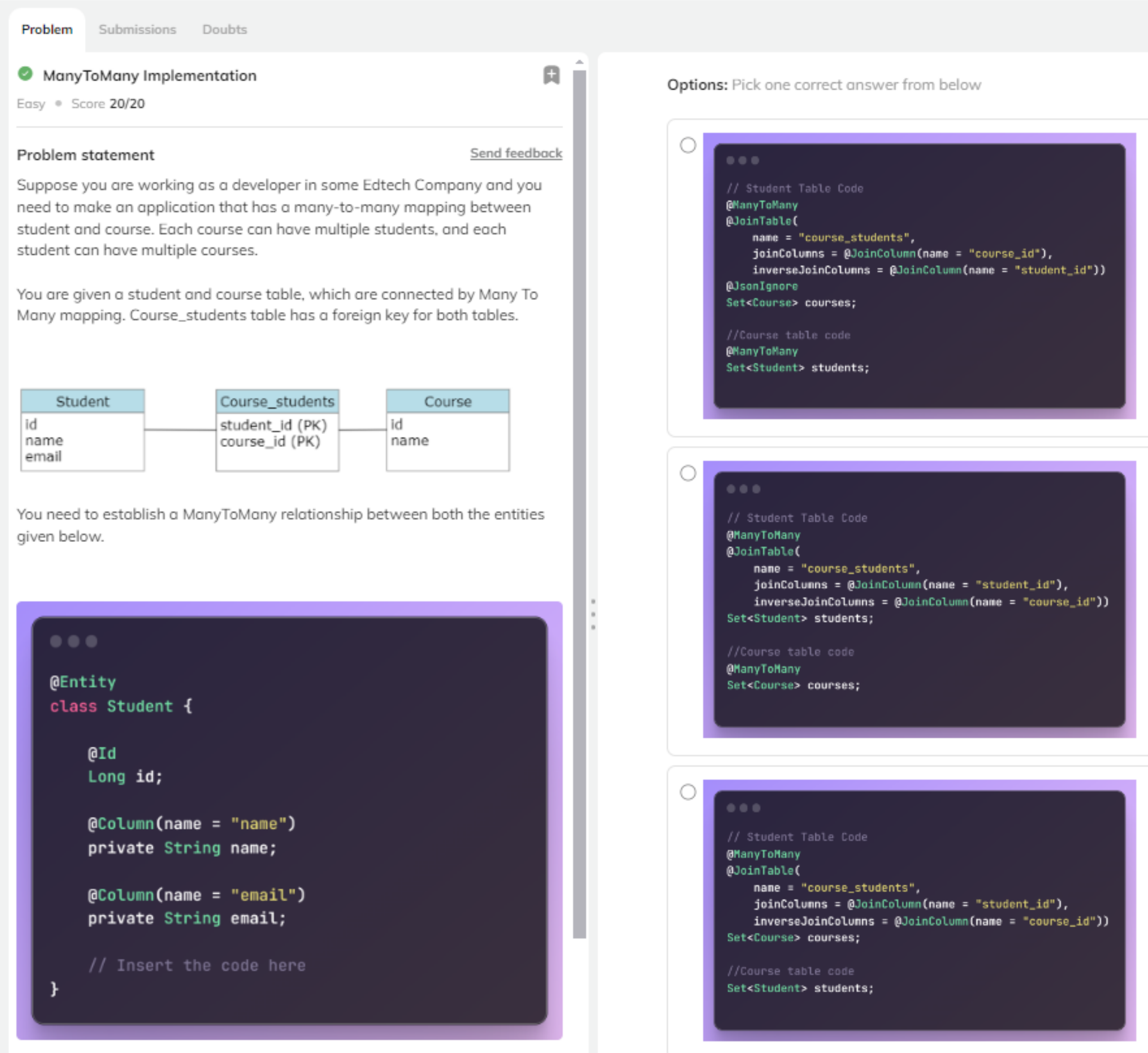
●
```
// User Table Code
@OneToMany(mappedBy="user")
private Set<Item> items;

// Item Table code
@ManyToOne
@JoinColumn(name="user_id")
private User user;
```

○
```
// User Table Code
@OneToMany(mappedBy="user_id")
private Set<Item> items;

// Item Table code
@ManyToOne
@JoinColumn(name="user")
private User user;
```

○
```
// User Table Code
@OneToMany
private Set<Item> items;

// Item Table code
@ManyToOne(mappedBy="user")
@JoinColumn(name="user_id")
private User user;
```

○
```
// User Table Code
@OneToMany(mappedBy="user")
@JoinColumn(name="user_id")
private Set<Item> items;

// Item Table code
@ManyToOne
private User user;
```

(6) Many to Many Implementation



ManyToMany Implementation

Easy • Score 20/20

Problem statement

Send feedback

Suppose you are working as a developer in some Edtech Company and you need to make an application that has a many-to-many mapping between student and course. Each course can have multiple students, and each student can have multiple courses.

You are given a student and course table, which are connected by Many To Many mapping. Course_students table has a foreign key for both tables.

Student
id name email

Course_students
student_id (PK) course_id (PK)

Course
id name

You need to establish a ManyToMany relationship between both the entities given below.

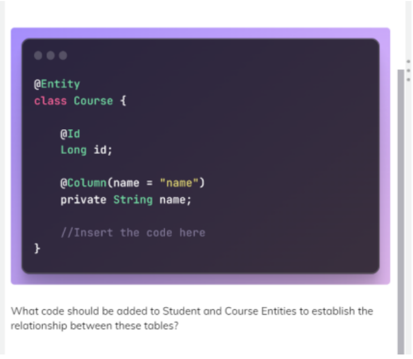
```
@Entity
class Student {

    @Id
    Long id;

    @Column(name = "name")
    private String name;

    @Column(name = "email")
    private String email;

    // Insert the code here
}
```

Options: Pick one correct answer from below

○
```
// Student Table Code
@ManyToMany
@JoinTable(
    name = "course_students",
    joinColumns = @JoinColumn(name = "course_id"),
    inverseJoinColumns = @JoinColumn(name = "student_id"))
@JsonIgnore
Set<Course> courses;

//Course table code
@ManyToMany
Set<Student> students;
```

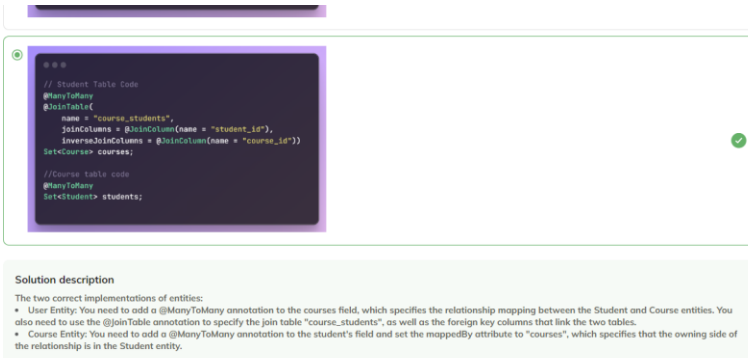
○
```
// Student Table Code
@ManyToMany
@JoinTable(
    name = "course_students",
    joinColumns = @JoinColumn(name = "student_id"),
    inverseJoinColumns = @JoinColumn(name = "course_id"))
Set<Student> students;

//Course table code
@ManyToMany
Set<Course> courses;
```

○
```
// Student Table Code
@ManyToMany
@JoinTable(
    name = "course_students",
    joinColumns = @JoinColumn(name = "student_id"),
    inverseJoinColumns = @JoinColumn(name = "course_id"))
Set<Course> courses;

//Course table code
Set<Student> students;
```



```
@Entity
class Course {

    @Id
    Long id;

    @Column(name = "name")
    private String name;

    //Insert the code here
}
```

What code should be added to Student and Course Entities to establish the relationship between these tables?



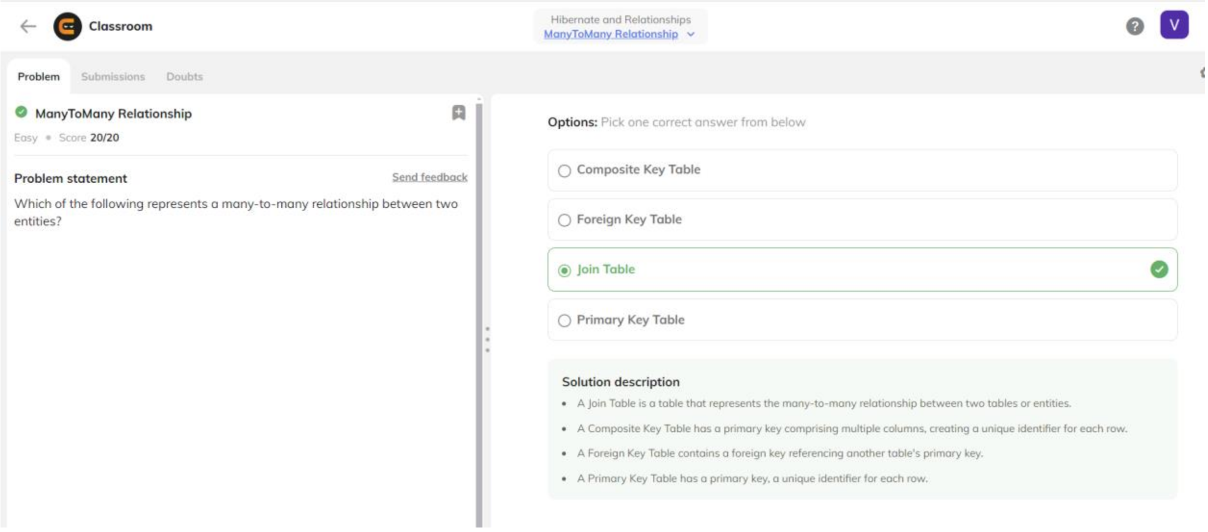
●
```
// Student Table Code
@ManyToMany
@JoinTable(
    name = "course_students",
    joinColumns = @JoinColumn(name = "student_id"),
    inverseJoinColumns = @JoinColumn(name = "course_id"))
Set<Course> courses;

//Course table code
@ManyToMany
Set<Student> students;
```
✓

Solution description

The two correct implementations of entities:

- User Entity: You need to add a @ManyToMany annotation to the courses field, which specifies the relationship mapping between the Student and Course entities. You also need to use the @JoinTable annotation to specify the join table "course_students", as well as the foreign key columns that link the two tables.
- Course Entity: You need to add a @ManyToMany annotation to the student's field and set the mappedBy attribute to "courses", which specifies that the owning side of the relationship is in the Student entity.

(7) Many to Many Relationship



ManyToMany Relationship

Easy • Score 20/20

Problem statement Send feedback

Which of the following represents a many-to-many relationship between two entities?

Options: Pick one correct answer from below

○ Composite Key Table

○ Foreign Key Table

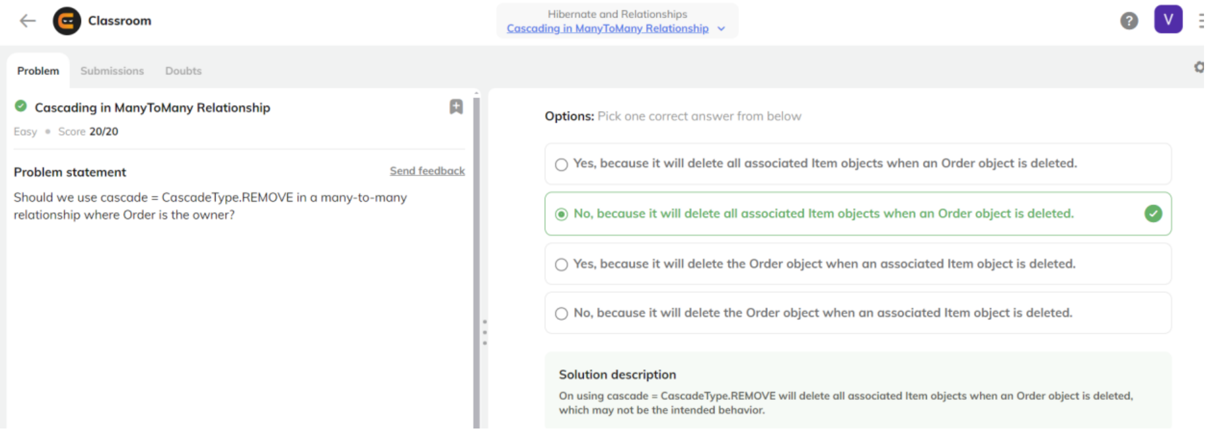
● Join Table ✓

○ Primary Key Table

Solution description

- A Join Table is a table that represents the many-to-many relationship between two tables or entities.
- A Composite Key Table has a primary key comprising multiple columns, creating a unique identifier for each row.
- A Foreign Key Table contains a foreign key referencing another table's primary key.
- A Primary Key Table has a primary key, a unique identifier for each row.

(8) Cascading in Many to Many Relationship


Classroom
Hibernate and Relationships
Cascading in ManyToMany Relationship
V
Problem
Submissions
Doubts

Cascading in ManyToMany Relationship

Easy • Score 20/20

Problem statement Send feedback

Should we use cascade = CascadeType.REMOVE in a many-to-many relationship where Order is the owner?

Options: Pick one correct answer from below

○ Yes, because it will delete all associated Item objects when an Order object is deleted.

● No, because it will delete all associated Item objects when an Order object is deleted. ✓

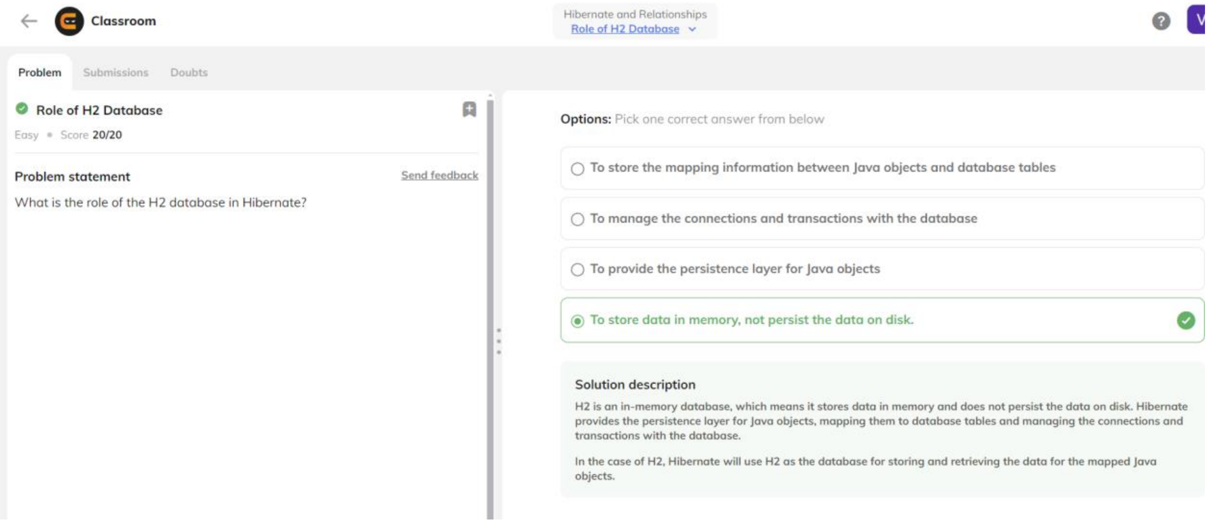
○ Yes, because it will delete the Order object when an associated Item object is deleted.

○ No, because it will delete the Order object when an associated Item object is deleted.

Solution description

On using cascade = CascadeType.REMOVE will delete all associated Item objects when an Order object is deleted, which may not be the intended behavior.

(9) Roles of H2 Database



Role of H2 Database

Easy • Score 20/20

Problem statement Send feedback

What is the role of the H2 database in Hibernate?

Options: Pick one correct answer from below

○ To store the mapping information between Java objects and database tables

○ To manage the connections and transactions with the database

○ To provide the persistence layer for Java objects

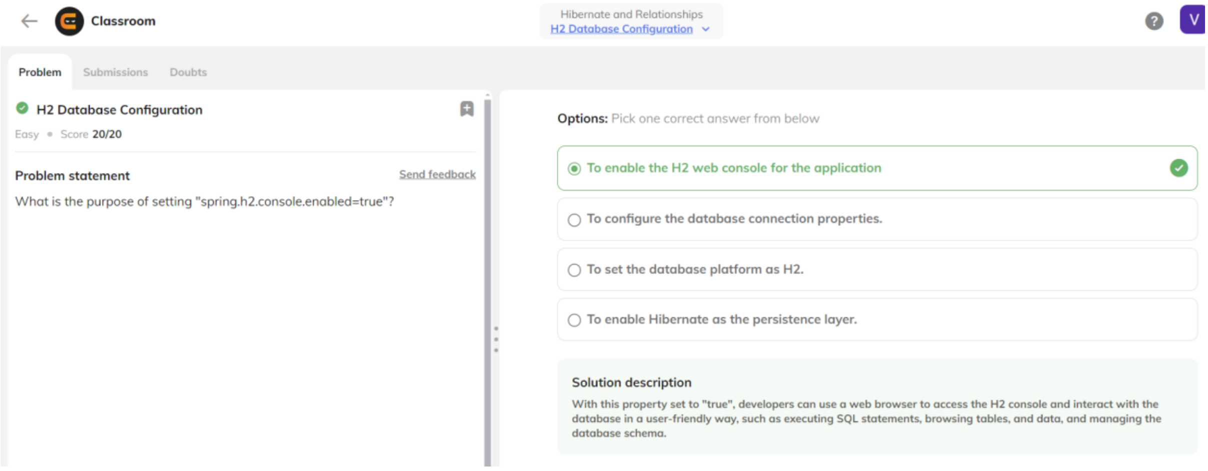
● To store data in memory, not persist the data on disk. ✓

Solution description

H2 is an in-memory database, which means it stores data in memory and does not persist the data on disk. Hibernate provides the persistence layer for Java objects, mapping them to database tables and managing the connections and transactions with the database.

In the case of H2, Hibernate will use H2 as the database for storing and retrieving the data for the mapped Java objects.

(10) H2 Database Configuration



H2 Database Configuration

Easy • Score 20/20

Problem statement

What is the purpose of setting "spring.h2.console.enabled=true"?

Options: Pick one correct answer from below

● To enable the H2 web console for the application ✓

○ To configure the database connection properties.

○ To set the database platform as H2.

○ To enable Hibernate as the persistence layer.

Solution description

With this property set to "true", developers can use a web browser to access the H2 console and interact with the database in a user-friendly way, such as executing SQL statements, browsing tables, and data, and managing the database schema.