HIBERNATE-MAPPING

Mapping establishes the relationship between two database tables as attributes in your model. that allows you to easily navigate the associations in your model and criteria queries.

you can establish either unidirectional or bidirectional

the relationship that can be established between entities are-

- **one to one** — it represents the one to one relationship between two tables.
- **one to many/many to one** — it represents the one to many relationship between two tables.
- **many to many** — it represents the many to many relationship between two tables.

One to One - Teacher vs Details → Unidirectional

Teacher

```
@Entity
@Table(name = "instructor")
public class Instructor {

    @Id
    @GeneratedValue(strategy = GenerationType.IDENTITY)
    @Column(name = "id")
    private int id;

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

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

    @OneToOne(cascade = CascadeType.ALL)
    @JoinColumn(name = "instructor_detail_id")
    private InstructorDetail instructorDetail;
}
```

Details

```java
@Entity
@Table(name = "instructor_detail")
public class InstructorDetail {

    @Id
    @GeneratedValue(strategy = GenerationType.IDENTITY)
    @Column(name = "id")
    private int id;

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

    @Column(name = "hobby")
    private String hobby;
}
```

One To One - Teacher vs Details →Bidirectional

Teacher

```java
@Entity
@Table(name = "instructor")
public class Instructor {

    @Id
    @GeneratedValue(strategy = GenerationType.IDENTITY)
    @Column(name = "id")
    private int id;

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

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

    @OneToOne(cascade = CascadeType.ALL, mappedBy = "instructor", fetch =
FetchType.LAZY)
    private InstructorDetail instructorDetail;
}
```

Details

```java
@Entity
@Table(name = "instructor_detail")
public class InstructorDetail {

    @Id
    @GeneratedValue(strategy = GenerationType.IDENTITY)
    @Column(name = "id")
    private int id;

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

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

    @OneToOne(fetch = FetchType.LAZY)
    @JoinColumn(name = "instructor_id")
    private Instructor instructor;

}
```

One To Many/Many To One→Unidirectional

Instructor

```java
@Entity
@Table(name = "instructor")
public class Instructor {

    @Id
    @GeneratedValue(strategy = GenerationType.IDENTITY)
    @Column(name = "id")
    private int id;

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

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

    @OneToMany(cascade = CascadeType.ALL)
    private List < Course > courses = new ArrayList < Course > ();
```

}

Course

```java
@Entity
@Table(name = "course")
public class Course {

    @Id
    @GeneratedValue(strategy = GenerationType.IDENTITY)
    @Column(name = "id")
    private int id;

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

}
```

One To Many/Many To One→Bidirectional

Instructor

```java
@Entity
@Table(name = "instructor")
public class Instructor {

    @Id
    @GeneratedValue(strategy = GenerationType.IDENTITY)
    @Column(name = "id")
    private int id;

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

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

    @OneToMany(mappedBy = "instructor", cascade = { CascadeType.PERSIST,
CascadeType.MERGE, CascadeType.DETACH,
      CascadeType.REFRESH })
    private List<Course> courses;
}
```

Course

```java
@Entity
@Table(name = "course")
public class Course {

    @Id
    @GeneratedValue(strategy = GenerationType.IDENTITY)
    @Column(name = "id")
    private int id;

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

    @ManyToOne(cascade = CascadeType.ALL)
    @JoinColumn(name = "instructor_id")
    private Instructor instructor;
}
```

Many To Many - Employee vs Projects

Employee

```java
@Entity
@Table(name = "employees")
public class Employee {

    private static final long serialVersionUID = 1L;

    @Id
    @Column(name = "employee_id")
    @GeneratedValue(strategy = GenerationType.IDENTITY)
    private Long employeeId;

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

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

    @ManyToMany(cascade = {
        CascadeType.ALL
    })
    @JoinTable(
        name = "employees_projects",
        joinColumns = {
            @JoinColumn(name = "employee_id")
        },
        inverseJoinColumns = {
            @JoinColumn(name = "project_id")
        }
    )
    Set < Project > projects = new HashSet < Project > ();
}
```

Projects

```java
@Entity
@Table(name = "projects")
public class Project {

    @Id
    @Column(name = "project_id")
    @GeneratedValue
    private Long projectId;

    @ManyToMany(mappedBy = "projects", cascade = { CascadeType.ALL })
    private Set<Employee> employees = new HashSet<Employee>();
}
```