Entity Mapping and Persistence with JPA

This document explains the entity mapping and persistence strategy used in a Java application leveraging JPA (Java Persistence API). The application models various entities such as Employee, Doctor, Nurse, Department, Patient, and Ward using inheritance and relationships. The strategy simplifies the schema while ensuring efficient data management.

1. Inheritance Mapping

JPA provides several inheritance strategies to map an object-oriented hierarchy to a relational database schema. In this model, the SINGLE_TABLE strategy is used, which means all classes in the hierarchy are mapped to a single table.

The base class 'Employee' is defined with the @Entity annotation and @Inheritance(strategy = InheritanceType.SINGLE_TABLE). A discriminator column is used to distinguish between different subclasses, which is specified using the @DiscriminatorColumn annotation.

2. Employee Entity

The 'Employee' class is the base entity for all employees in the system. It contains common fields such as 'employeeNumber', 'surname', 'firstName', 'address', and 'telephoneNumber'. It is mapped as a single table with subclasses using:

- @Inheritance(strategy = InheritanceType.SINGLE_TABLE)
- @DiscriminatorColumn(name = "employee_type", discriminatorType = DiscriminatorType.STRING)

This setup ensures all employees are stored in one table with a column to indicate the specific type.

3. Doctor and Nurse Entities

The 'Doctor' and 'Nurse' classes extend the 'Employee' class and inherit its fields. Additional fields specific to doctors or nurses are included in their respective classes:

Doctor Entity:

- 'speciality' (String): Medical specialty of the doctor.

Nurse Entity:

- 'rotation' (String): Work rotation schedule of the nurse.

- 'salary' (BigDecimal): Salary of the nurse.

Both entities are distinguished in the database using the @DiscriminatorValue annotation, which assigns a value to the 'employee_type' column to identify the record type.

4. Department Entity

The 'Department' entity represents the various departments within the hospital. It includes fields such as 'code', 'name', and 'building'. Additionally, the 'Department' has relationships with the 'Doctor' and 'Nurse' entities:

- One-to-One relationship with 'Doctor' to denote the department director.

- One-to-Many relationship with 'Nurse' to denote nurses assigned to the department.

These relationships are mapped using @OneToOne and @OneToMany annotations.

5. Patient Entity

The 'Patient' entity represents patients in the hospital. It contains fields such as 'number', 'name', 'surname', 'address', 'telephoneNumber', 'bedNumber', and 'diagnosis'. The entity also has relationships with 'Ward' and 'Doctor':

- Many-to-One relationship with 'Ward' to denote the ward the patient is assigned to.

- Many-to-One relationship with 'Doctor' to denote the doctor responsible for the patient.

These relationships are mapped using @ManyToOne annotations.

6. Ward Entity

The 'Ward' entity represents the various wards within the hospital. It includes fields such as 'number' and 'numberOfBeds'. Additionally, the 'Ward' has relationships with the 'Nurse' and 'Patient' entities:

- Many-to-One relationship with 'Nurse' to denote the ward supervisor.

- One-to-Many relationship with 'Patient' to denote patients assigned to the ward.

These relationships are mapped using @ManyToOne and @OneToMany annotations.

7. Database Table Structure

With the SINGLE_TABLE inheritance strategy and various entity relationships, the database schema is structured to optimize data retrieval and management while maintaining a normalized schema. The table structure includes:

- 'Employee' table for all employee types with a discriminator column 'employee_type'.

- 'Department' table for hospital departments with references to employees.

- 'Patient' table for patients with references to wards and doctors.

- 'Ward' table for hospital wards with references to supervisors and patients.

This setup ensures efficient management of hospital data with clear relationships between entities.