Assignment: JPA/Hibernate, one table

Goal:

In this assignment, you learn to carry out basic tasks with JPA and Hibernate in a one-table case.

Instructions:

Before doing this assignment, you should have the following software installed:

1. Eclipse IDE. The newest version can be downloaded at: https://www.eclipse.org/downloads/

2. JDK: https://www.oracle.com/java/technologies/downloads/. Versions 11 ja 17 are long-time support (LTS) versions. Any version (11 or later) will probably work.

For deliverables of this assignment, submit a single pdf file containing the requested items (screenshots, source codes etc.) specified with each task below.

Tasks:

1. First, build the infrastructure for the assignment. Unzip the **Devices.zip** into your local file system. Open the **Devices** project in Eclipse (**File / Open projects from file system**). Locate the **persistence.xml** file, and check its contents for the database and login information.

   In HeidiSQL, create the database (**CREATE DATABASE**) as well as the user account (**CREATE USER**). Edit **persistence.xml** to make the credentials match those specified at the database level. Use **GRANT** statement to provide the newly-created user account with privileges to modify the both the table structure and the data of the database Hint: for simplicity, you can use **GRANT ALL**.

2. Run **DeviceApp.java** (open the file in the IDE, right-click and select **Run as / Java application**). Verify that the code runs without errors and the results are visible at the database level.

   Deliverable: a screenshot of the table contents in the HeidiSQL or MariaDB console.

3. Expand the **Dao** class to contain a method that loads a device based on the identifier value. Make the application print the description of the fetched device on the Java console.

Deliverable: method source code.

4. Expand the **Dao** class by writing a method **boolean updateDeviceDescription(int id, String newDescription)** that changes the device's description. The method should return a boolean value that indicates whether the update was a success or not. Verify the correct functioning at the database level.

    Deliverable: method source code.

5. Modify the annotations in the **Device** class to force auto-generation of the primary key values. Verify the correct functioning at the database level.

    *Deliverable: The annotated source code of the **Device** class.*

6. Expand the **Dao** class by writing a method for removing a device (**find()** followed by **remove()**).

    Deliverable: the source code of the method.