Database connectivity (DAO) exercises

For this exercise, you'll be responsible for implementing the data access objects for a CLI application that manages information for employees, departments, and projects. The purpose of this exercise is to practice writing application code that interacts with a database.

Learning objectives

After completing this exercise, you'll understand:

- How to create database connections.
- How to execute SQL statements and use parameters.
- How the DAO pattern encapsulates database access logic.

Evaluation criteria and functional requirements

Your code will be evaluated based on the following criteria:

- The project must not have any build errors.
- The unit tests pass as expected.
- Code is clean, concise, and readable.

You may use the provided CLI application to test your code. However, your goal is to make sure the tests pass.

Getting started

1. In the `/database` folder, there's an `EmployeeProjects.sql` SQL script that drops and recreates the tables and data in the `EmployeeProjects` database. You can run that script to create a copy of the database to reference while working. Be aware, however, that the tests don't use that database. The tests use a temporary database with the same structure. You'll find the SQL for that temporary database in `src/test/resources/test-data.sql`.
    - *Note that the `timesheet` table is not used in today's exercise.*
2. Import the DAO exercises project into IntelliJ.
3. Run all tests to see the results of your tests and which ones passed or failed.

Step One: Explore starting code and database schema

Before you begin, review the provided classes in the `model` and `dao` packages.

You'll write your code to complete the data access methods in the `JdbcDepartmentDao`, `JdbcProjectDao`, and `JdbcEmployeeDao` classes.

You should also familiarize yourself with the database schema either by looking at the database in pgAdmin or the `database/EmployeeProjects.sql` script.

Step Two: Implement the `JdbcDepartmentDao` methods

Complete the data access methods in `JdbcDepartmentDao`. Refer to the documentation comments in the `DepartmentDao` interface for the expected input and result of each method.

You can remove any existing `return` statement in the method when you start working on it.

After you complete this step, the tests in `JdbcDepartmentDaoTests` pass.

Step Three: Implement the `JdbcProjectDao` methods

Complete the data access methods in `JdbcProjectDao`. Refer to the documentation comments in the `ProjectDao` interface for the expected input and result of each method.

You can remove any existing `return` statement in the method when you start working on it.

After you complete this step, the tests in `JdbcProjectDaoTests` pass.

Step Four: Implement the `JdbcEmployeeDao` methods

Complete the data access methods in `JdbcEmployeeDao`. Refer to the documentation comments in the `EmployeeDao` interface for the expected input and result of each method.

You can remove any existing `return` statement in the method when you start working on it.

After you complete this step, the tests in `JdbcEmployeeDaoTests` pass.