
Q1 :

**** Modify the 011-Jdbc assignment as below**

Create the Address table with addressLine, city, pin, country

Establish a one-to-one relation between employee and address (Alter/modify the employee table or address table as required for setting primary/foriegn key)

Modify addEmployee method of dao to use transaction management (The signature of method will now have employee as well as address object)
The method shud insert employee as well as address both or neither.

Rewrite the tests to fit new requirements

Also modify the EmployeeService and the implementation accordingly.

Think in terms of creating AddressDAO interface and implementation for Address.java that you will create to map to Address table.
This DAO shud handle CRUD operations for Address.
This dao will be injected in EmployeeDAOImpl

Additionally Also create service layer to independantly query Addresses

---

Q2 :

**** Create the following maven application.**

Make Use of Transaction management. Make use of HokariConnectionPools
Externalize the required configutaion properties.

Create a Savingsaccount table with the columns as acct_num, name, balance.

Create a SacvingsAccount class with name, AccNo, balance.

Create AccountsDAO interface with methods withdraw(int fromAccNo,double amount) and deposit(int toAccNo,double amount) to
simulate transfer amount from one account to other account. (If deemed necessary add any other methods required to the interface)

Create class AccountsDAOImpl that implements AccountsDAO
Make Use of custom exceptions to handle minimum amount withdrawn must be 500 or above. After withdrawal a balance of 1000 must be maintained
An Amount upto 25000 only can be transferred. Also note that both the accounts must be present in the table.
Only in case of No exceptions the table rows should be updated appropriately for

accounts being withdrawn from and account being deposited to,
Else No rows in DB table needs to be updated.

Create an interface TransferService with method transfer(int fromAcc, int toAcc, double amount);

Create the class TransferServiceImpl that implements the TransferService. The transfer method invokes the withdraw and deposit of Dao
Provide checks to see that fromAcc and toAcc and amount must be non-zero positive values.

Create a class TestTransferService with main method.
Try out the TransferService transfer method.