Object Relational Mapping with JPA

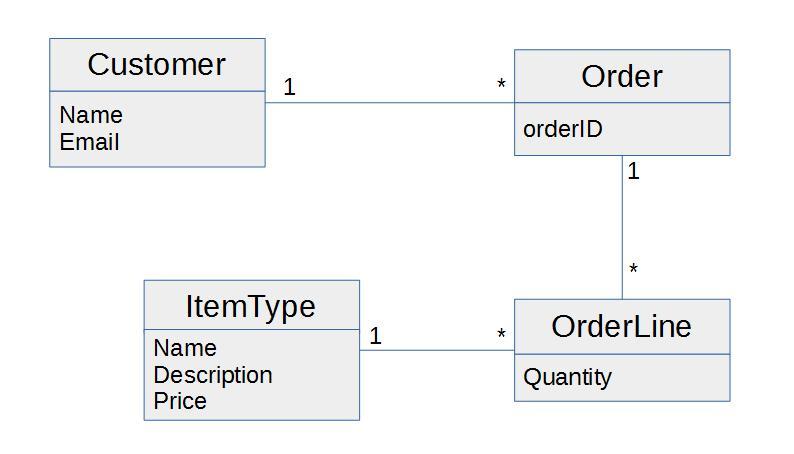
About this exercise

*This exercise was originally designed as a preparation for a two-hour exam, so it does not represent the complexity of the new 24 hours exam. Expect to spend 2-4 hours since right now, you are learning, so complete it Friday. Also, and important, you are not all expected to complete it from A-Z. It was meant as an example of a real exam exercise and they have to cover a span from the grades 2-12.*

|  |  |
| --- | --- |
|  | [*Watch this before you start*](https://cphbusiness.cloud.panopto.eu/Panopto/Pages/Viewer.aspx?id=51d20fb5-7520-40a7-a219-aacc013a4b69) |

General part

* *Explain the rationale behind the topic Object Relational Mapping and the Pros and Cons in using ORM*Pros  
  ORM er smart da det tillader dig at ret nemt tilgå og manipulere objekter I databasen.  
  Det beskytter mod SQL injections.  
  Nemt at vedligeholde.  
  Gør det nemt at skrive SQL  
  Cons:  
  Komplekse sql statements kan give problemer med performance.  
  Databasen kan hurtig blive meget abstrakt, som kan gøre det svært at troubleshoote.
* *Explain the JPA strategy for handling Object Relational Mapping and important classes/annotations involved.*Der er meget der fungerer gennem tags, som fx. @OneToOne og @Column. Vigtige klasser er fx EntityManager, der sørger for al kommunikation gennem ens Persistence Unit.
* *Outline some of the fundamental differences in Database handling using plain JDBC versus JPA*JPA opretter selv tabeller for dig og skriver noget af SQL koden for dig. Hvor JDBC kører de kommandoer du beder den om. Så JDBC alene kræver en større viden om SQL end JPA – Hvilket for nogen kan være en udfordring.

****Practical part

This model represents an initial model for a system that can handle orders. Order refers to a customer and a number of order lines. Each order line has a quantity and it refers to an item type. The item type has a name, a description and a price. The price for each order line is the quantity times the price. The total price for an order is the sum of all its order lines.

1. Examine and understand the diagram.
2. Create a Maven Java Application with NetBeans, and use Object Relational Mapping (JPA) to implement the OO classes and the corresponding Database Tables.
3. Create a façade and implement as many of the methods below as you have time for, not necessarily in the given order:

* Create a Customer
* Find a Customer
* Get all Customers
* Create an ItemType
* Find an ItemType
* Create an Order and Add it to a Customer
* Create an OrderLine for a specific ItemType, and add it to an Order
* Find all Orders, for a specific Customer
* Find the total price of an order ….