

# JPA

Martin Ledvinka

[martin.ledvinka@fel.cvut.cz](mailto:martin.ledvinka@fel.cvut.cz)

Winter Term 2022



# Contents

1 Introduction

2 Tasks

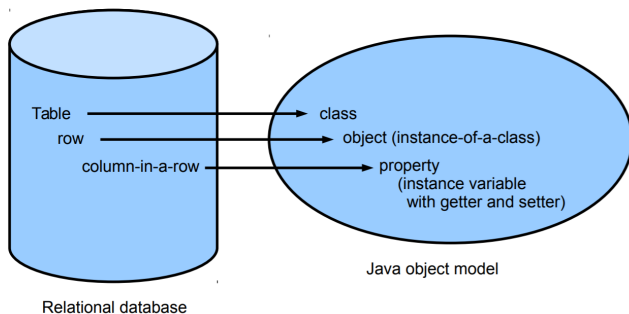


# Introduction



# Object-relational mapping

Mapping between the database (declarative) schema and the data structures in an object oriented language.



# JPA Basics

- The idea: “map Java classes to database records”
- Object-relational mapping in Java

```
@Entity
public Person {
    @Id
    private Long id;
    private String firstName;
    private String lastName
    // setters+getters
}
```

```
CREATE TABLE PERSON (
  ID bigint PRIMARY KEY NOT NULL,
  FIRSTNAME varchar(255),
  LASTNAME varchar(255)
);
```



# JPA Main Concepts

**Entity class** – a Java class representing a set of persistent objects mapped onto a relational table

**Persistence Unit** – the set of all entity classes that are persistently mapped to one database

**Persistence Context** – the set of all entities defined in the persistence unit being used at a given time

**Entity manager** – the interface for interacting with a Persistence Context



# JPA Main Concepts – Visual

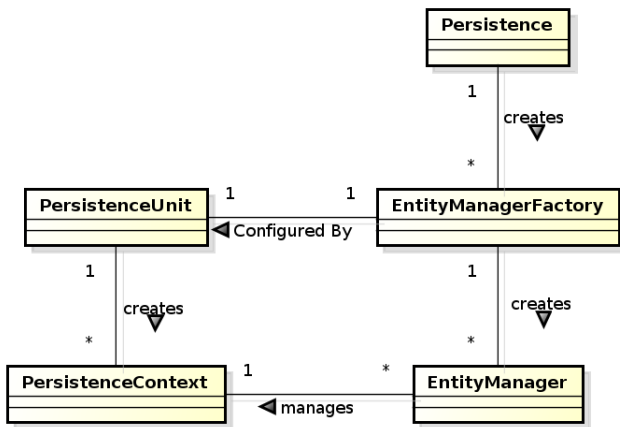


Figure: Main JPA concepts and their relationship.



# JPA – EntityManager

- **EntityManager (EM)** instance is in fact a generic DAO, while entities can be understood as DTO's.
- Selected operations on EM (CRUD):

**Create** : `em.persist`

**Read** : `em.find`, `em.refresh`

**Update** : `em.merge`

**Delete** : `em.remove`

**Native/JPQL queries** : `em.createNativeQuery`,  
`em.createQuery`, `em.createNamedQuery`, etc.

**Resource-local transactions** :

`em.getTransaction.[begin,commit,rollback]`





# Implementations

- We will be using *Eclipselink*
  - Reference JPA implementation
- Another (arguably more popular) solution is *Hibernate*
  - JPA implementation + its own, *Session*-based API
  - *Session* ~ *EntityManager*
  - Provided by the Spring Boot `spring-boot-starter-data-jpa` project by default
- More: *Apache OpenJPA*, *DataNucleus*, etc.
- Each implementation has its quirks, additional options/specific API, and problems



# Tasks



# Syncing Your Fork

- 1 Add upstream remote to the local clone of your fork (if you have not done so already)
  - `git remote add upstream  
git@gitlab.fel.cvut.cz:ear/b221-eshop.git`
- 2 Fetch branches and commits from the upstream repository (EAR/B221-eshop)
  - `git fetch upstream`
- 3 Check out local branch corresponding to the task branch
  - `git checkout -b b221-seminar-03-task`
- 4 Merge changes from the corresponding upstream branch
  - `git merge upstream/b221-seminar-03-task`
- 5 Do your task
- 6 Push the solution to your fork
  - `git push origin b221-seminar-03-task`

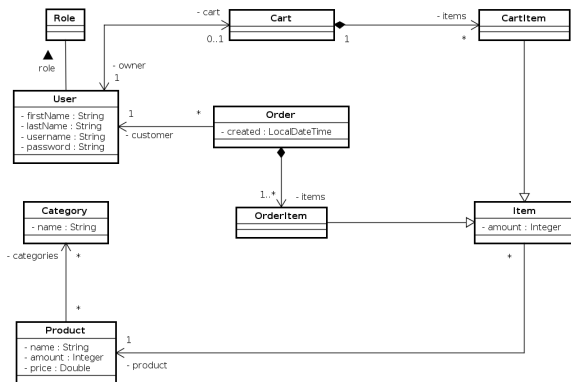


## Task: 1 Point

The application is missing an object model (JPA entity classes). Create an object model corresponding to the class diagram below.

*We will start together, your task is to finish it.*

**Acceptance criteria:** Project is compilable and the object model corresponds to the class diagram below.



# The End



The End

Thank You



# Resources

- [https://cw.fel.cvut.cz/b221/\\_media/courses/b6b36ear/lectures/lecture-03-jpa-s.pdf](https://cw.fel.cvut.cz/b221/_media/courses/b6b36ear/lectures/lecture-03-jpa-s.pdf)
- <http://www.objectdb.com/api/java/jpa/annotations>

