Exam-preperation jpa 1

# General part

## • Describe how you have handled persistence in the last three semesters.

In the first semester we started out initially by not handling persistence at all. Nothing was saved and programs ran from clean slate every time we started them up. Then we were introduced to text files and input and output writers. These allowed us to open and write to files on our hard disks locally as well as read from them. This allowed the persistence of data on the most primitive level. On the second semester we were introduced to MySQL databases. These allowed us to transform entity classes in Java domain into data that could be stored in tables in database domain. We could now persist data in tables in databases using SQL queries. Finally, on the third semester we expand on the method from the 2nd semester working with ORM and JPA. These allow us to stay in the Java domain and have the SQL queries to and from a SQL database happen in the background for us not having to write any SQL ourselves.   
When working with HTTP and browsers we were able to “fake” state in an otherwise stateless protocol with the use of cookies and headers. Headers contained data in its body such as login authentication for instance. Cookies are similar to text files and contain data about the user again such as login information.

## Explain the Pros & Cons in using an Object Relational Mapping Framework

### Pros

We already touched a little on the pros of using ORM and JPA in the previous section. One pro is that we as programmers can work only in the Object-oriented domain (in Java) and not worry or care about the SQL at all. We reap the benefits of only working OO in Java.   
Furthermore, we heavily reduce the amount of coded that we now need to write because frameworks like JPA give us several prebuild tools to use.   
Working with ORM and JPA also means that we have the option to use concurrency.   
Thanks to our entity manager factories and managers, all access to and from the database is handled in transactions that obey the principles of ACID. We do not need to enforce this manually.

### Cons

The structure of all the things persisted to the database can be messy. We can end up with duplicate entries, null entries (based on our strategy with inheritance for instance) and normalization is difficult.

Every time you add a new layer of abstraction that tries to provide a generalized implementation of something that you used to do by hand there is going to be some loss of performance/efficiency.

When we are working exclusively object-oriented which I an advantage for the developer, it means that it is automatically a disadvantage on the end of the database, which doesn’t operate in the same domain. As a result, we lose performance efficiency which is proportional with how much we scale up. Think for instance 100.000 users and the queries that we have to do.

### Elaborate on some of the problems a ORM tries to solve

At a very high level: ORMs help to reduce the Object-Relational impedance mismatch. They allow you to store and retrieve full live objects from a relational database without doing a lot of parsing/serialization yourself.

When working with databases the developer has to write a lot of boilerplate-code. ORM attempts to reduce this amount, which makes it easier for the developer, or rather helps him by allowing him to write less code and faster.

Lastly the goal of using ORM can be summarized as the attempt at mapping objects from the OO world into tables in a relational database and provide a runtime mechanism for keeping an in-memory graph of objects and a set of database tables in sync

## Explain the basic “components” involved when using Java’s JPA

Annotations is the component that allow us to specify the behavior of the auto generated SQL. It enables us to customize how our entities are interpreted and it is a component that basically gives us control.

We have entity classes which represent a row in an end-result table. Each entity represents each entry in a database’s table. These exist as objects in Java domain and as rows in database domain.

Entities are handled by entity managers. Entity managers provide us with transactions that involve entities. All the manipulation of entities is done by these.

Entity managers are generated by entity manager factories.

The persistence unit is a xml document. It specifies what type of persistence provider we are using and it allows us to “Persist” something from Java domain to database domain.