AG Heterogene Informationssysteme
Prof. Dr.-Ing. Stefan Deßloch
Fachbereich Informatik
Technische Universität Kaiserslautern

TECHNISCHE UNIVERSITÄT
KAISERSLAUTERN

Middleware for Heterogeneous and Distributed Information Systems – Exercise Sheet 4

Wednesday, November 19, 2008 – 10:00 to 11:30 – Room 48-379

EJB Entity Beans

Enterprise Java Beans (EJBs) are server-side components written in the Java programming language that encapsulate the business logic of an application. EJBs run in an EJB container within a J2EE application server. The EJB container provides runtime services such as transactional coordination, resource management, security, and transparent access to EJBs in remote containers. This allows the developer to focus on the business logic while infrastructural aspects can largely be ignored.

In this exercise we discuss EJB *entity beans*. Entity beans represent business objects in a persistent storage mechanism, typically a relational database. Two persistence mechanisms for entity beans exist, namely *bean-managed persistence* (BMP) and *container-managed persistence* (CMP).

1. As an example, consider an EJB entity bean to store information about university lectures (lecture id, title, description, day of week, and starting time) that uses bean-managed persistence. Name the software artifacts (Java interfaces, Java classes, other files) the entity bean consists of and explain their purpose!

2. Say, we frequently need to retrieve lecture bean instances based on their title. What mechanism is provided by entity beans to address this requirement? How does the implementation look like?

3. Consider another version of the lecture entity bean that uses the container-managed persistence mechanism. What are the differences between both versions? What are the advantages of the CMP mechanism? Do you see any drawbacks?

4. Comment on the following statement: "EJB entity beans do not provide orthogonal persistence; neither does the concept of transitive persistence apply."

EJB Query Language

The EJB Query Language (EJB QL) is used to define queries for CMP entity beans. EJB QL is a subset of SQL92. It provides extensions that allow navigation over the relationships defined in an entity bean's abstract persistence schema. EJB QL queries are specified in the deployment descriptor of a CMP entity bean to define its `finder` methods. EJB QL queries are translated into the language of the underlying data store at deployment time.
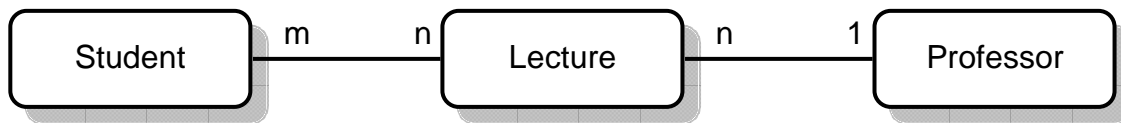
Figure 1: Sample abstract persistence schema

Consider the abstract persistence schema depicted in Figure 1. Give EJB QL statements for the following queries.

1. Find all lectures.

2. Given a lecture title, find all lectures with that title.

3. Given a keyword, find all lectures that contain the keyword in their description.

4. Given a lecture (by its primary key), find all lectures that are scheduled on the same weekday.

5. Given a lecture name, find all students that attend lectures of that name.

6. Given a student id, find all professors that teach this student.

7. Find all professors that do not teach any lecture.

8. Find all students that do not attend lectures starting before 10:00 am.

Java Persistence API

In Java EE 5, entity beans have been replaced by Java persistence API entities. An entity is a lightweight persistence domain object. Object/relational mapping annotations are used to map the entity state and entity relationships to the underlying data store[1]. In contrast to EJB entity beans, the Java persistence API takes an orthogonal, transitive persistence approach.

Consider that the entities denoted in Figure 1 are implemented as both, EJB entity beans and Java persistence API entities. For both versions give an application program that completes the following tasks.

1. Create a new lecture entity.

2. Set the name and description of the newly created lecture.

3. Find all students that study computer science and add them to the lecture.

4. Finally, remove the lecture.

---

[1] See „The Java EE 5Tutorial" available at http://java.sun.com/javaee/5/docs/tutorial/doc/