

A large, light gray play button icon is positioned on the left side of the slide. It consists of a white right-pointing triangle centered within a series of concentric circles in varying shades of gray.

Querying and Modifying Entities

1. Querying entities
2. Modifying entities

1. Querying Entities

- Defining a repository component
- Finding an entity by primary key
- Working with queries
- Performing a simple query
- Getting a list of entities

Defining a Repository Component

- In this chapter we'll use JPA to query and modify entities
 - Via methods in the JPA `EntityManager` class
- We'll put all our data-access code in a repository component
 - We'll inject an `EntityManager` bean as follows:

```
import jakarta.persistence.*;
...

@Repository
public class EmployeeRepository {

    @PersistenceContext
    private EntityManager entityManager;

    // Methods to create, read, update, delete entities.
    ...
}
```

`EmployeeRepository.java`

Finding an Entity by Primary Key

- To find an entity by primary key:
 - Call `find()` on the `EntityManager`
 - Returns `null` if entity not found

```
public Employee getEmployee(long employeeId) {  
    return entityManager.find(Employee.class, employeeId);  
}
```

EmployeeRepository.java

Working with Queries

- Define a query string
 - Using JPQL (or SQL)
- Pass it into `EntityManager's createQuery()` method
 - The method returns a `TypedQuery<T>` object
- Call one of these methods on the `TypedQuery<T>` object:
 - `getSingleResult()`
 - `getResultList()`

Performing a Simple Query

- Here's a query that returns a single result:

```
public long getEmployeeCount() {  
    String jpql = "select count(e) from Employee e";  
    TypedQuery<Long> query = entityManager.createQuery(jpql, Long.class);  
    return query.getSingleResult();  
}
```

EmployeeRepository.java

Getting a List of Entities

- Here's a query that returns a list of entities:

```
public List<Employee> getEmployees() {  
    String jpql = "select e from Employee e";  
    TypedQuery<Employee> query = entityManager.createQuery(jpql, Employee.class);  
    return query.getResultList();  
}  
EmployeeRepository.java
```

2. Modifying Entities

- Overview
- Inserting an entity
- Updating an entity
- Deleting an entity

Overview

- JPA lets you insert, update, and delete entities
- You must put these operations in a transactional method in a component class
 - Annotate method with `@Transactional`

```
@Transactional  
public void someMethodToModifyEntities() {  
    ...  
}
```

Inserting an Entity

- This is how you insert an entity in the database:

```
@Transactional  
public void insertEmployee(Employee e) {  
    entityManager.persist(e);  
}
```

EmployeeRepository.java

Updating an Entity

- This is how you update an entity in the database:

```
@Transactional
public void employeePayRise(long id, double payRise) {
    Employee emp = entityManager.find(Employee.class, id);
    emp.setDosh(emp.getDosh() + payRise);
}
```

EmployeeRepository.java

Deleting an Entity

- This is how you delete an entity in the database:

```
@Transactional
public void deleteEmployee(long employeeId) {
    Employee e = entityManager.find(Employee.class, employeeId);
    entityManager.remove(e);
}
```

EmployeeRepository.java

A large, light gray play button icon is positioned on the left side of the slide. It consists of a white right-pointing triangle centered within a series of concentric circles in varying shades of gray.

Summary

- Querying entities
- Modifying entities

