Связи между сущностями

Связи между сущностями

- Кардинальность
  - Один к одному (@OneToOne)
  - Один ко многим (@OneToMany + @ManyToOne)
  - Многие ко многим (@ManyToMany)
- Полиморфность
  - Значением связи может быть объект не только указанного типа, но и любого из его подтипов
- Направленность
  - Однонаправленная связь
  - Двунаправленная связь
    - *Владеющая сторона* – определяет изменения связи в базе данных
    - Инверсная сторона

Двунаправленные связи

- Инверсная сторона ссылается на владеющую сторону с помощью элемента mappedBy
- В связях «один ко многим» владеющей должна быть сторона «многие»
- В связях «один к одному» владеющей считается сторона, которая содержит внешний ключ
- В связях «многие ко многим» владеющей может быть любая сторона

Обработка связей

- Провайдер персистентности выполняет объектно-реляционное преобразование связей – загрузку, сохранение, обеспечение ссылочной целостности (например, с помощью внешних ключей)
- ~~Целостность связей во время выполнения~~, в том числе взаимное соответствие сторон «один» и «многие» двунаправленной связи, должно обеспечивать само приложение
- Значением связи со стороны «многие» всегда является коллекция (возможно, пустая)

Пример двунаправленной связи

```
@Entity public class Employee {
  @Id private int id;
  private String firstName;
  private String lastName;
  @ManyToOne(fetch=LAZY)
  private Department dept;
  ...
}
@Entity public class Department {
  @Id private int id;
  private String name;
  @OneToMany(mappedBy = "dept", fetch=LAZY)
  private Collection<Employee> emps = new ...;
  ...
}
```

Пример управления двунаправленной связью

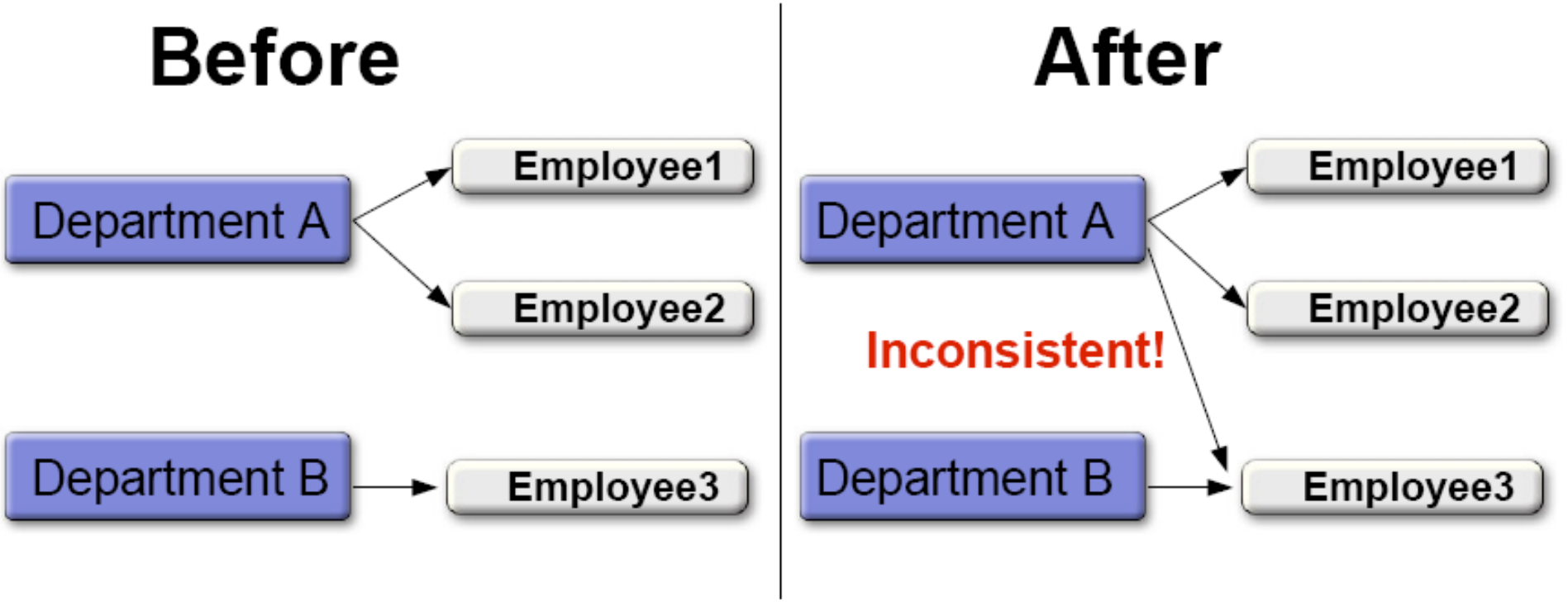
```
public int addNewEmployee(...) {
   Employee e = new Employee(...);
   Department d = new Department(1, ...);

   e.setDepartment(d);
   d.getEmployees().add(e);
   em.persist(e);
   em.persist(d);

   return d.getEmployees().size();
}
```

Пример управления связью

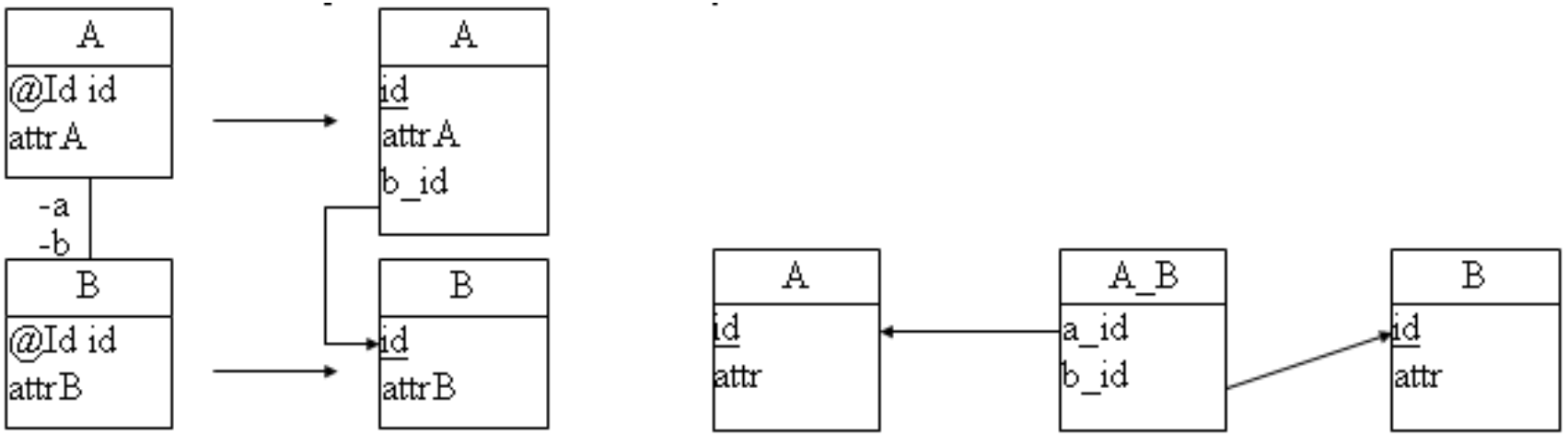
```
deptA.getEmployees().add(e3);
```



Отображение связей по умолчанию

- Сущность A является владельцем связи, сущности A и B отображаются на таблицы A и B

Ситуация	Внешний ключ в таблице A	Промежуточная таблица A_B	Уникальность внешнего ключа на таблицу B
A (1) – (1) B **A (1) –> (1) B**	**+**	**–**	**+**
A (N) – (1) B **A (N) –> (1) B**	**+**	**–**	**–**
A (N) – (N) B **A (N) –> (N) B**	**–**	**+**	**–**
A (1) –> (N) B	**–**	**+**	**+**



Стратегии загрузки данных (fetching)

- EAGER – немедленно
- LAZY – по мере необходимости