Agenda

Assoziationen

Cascading

n-1

1-n

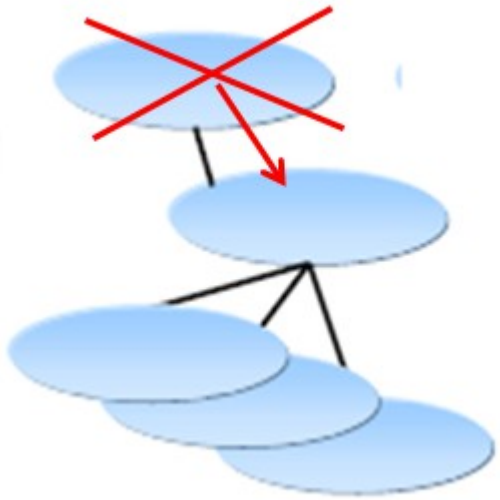
1-1

m-n

Komponenten

Propagieren von CRUDE-Operationen

- Soll ein persist(), delete() etc. auch auf abhängige Objekte angewandt werden?



Kaskadierungsverhalten mit Cascade Style

- Folgende Cascade Style sind vorhanden:
  - PERSIST, MERGE, REMOVE, REFRESH, DETACH
  - ALL
  - Default: Kaskadierung deaktiviert

- Delete-Orphan seit JPA 2.0

```
@OneToOne(cascade={CascadeType.PERSIST,
CascadeType.REMOVE,  CascadeType.REFRESH})
private Address address;

@OneToMany(cascade=CascadeType.ALL,
orphanRemoval=true)
private List<Role> roles;
```

Cascade Styles (Hibernate)

- Folgende Cascade Style sind vorhanden:
  - save-update, delete, lock, evict, replicate, all, delete-orphan

```
<one-to-one name= "adresse" cascade="save-update”/>
<one-to-one name="adresse" cascade="save-update, delete"/>
```

Agenda

Assoziationen

Cascading

n-1

1-n

1-1

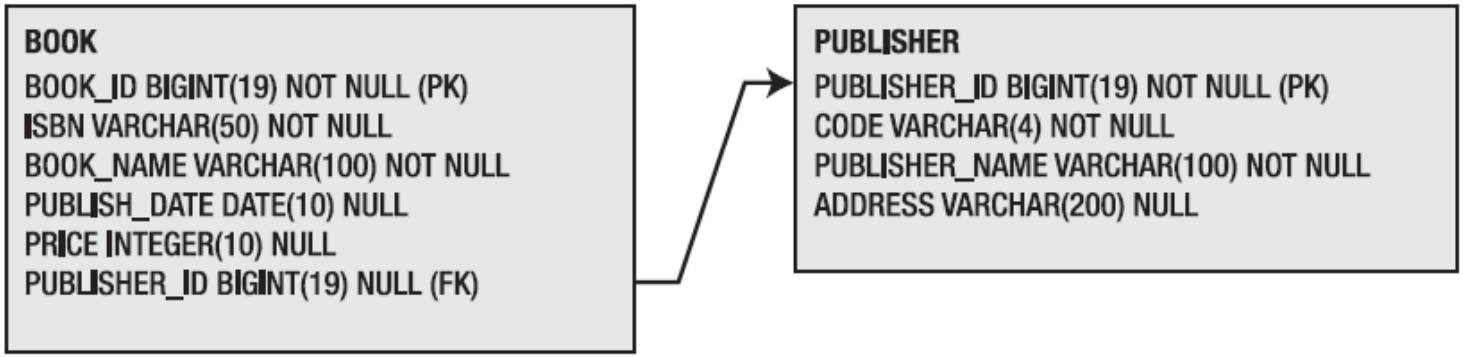
m-n

Komponenten

Relationen vs. Assoziationen

Relationen	Assoziationen
Verweis auf andere Tabellenzeile	uni- und bidirektional
Beziehung über Fremdschlüssel	Referenzen zu anderen Entitäten
gerichtet	Collection zu vielen anderen Entitäten (Set, List, Map)

Beispiel n-1 unidirektional



Beispiel n-1 unidirektional JPA


```
package com.hibernaterecipes.chapter5;
//imports

@Entity
//@Table(name = "PUBLISHER", schema = "BOOK5
public class Publisher implements Serializab
   @Id
   @Column(name = "PUBLISHER_ID")
   @GeneratedValue(strategy = GenerationType.AUTO)
   private Long publisher_id;
   @Column(name = "CODE")
   private String code;
   @Column(name = "PUBLISHER_NAME")
   private String name;
   @Column(name = "ADDRESS")
   private String address;
   // getters and setters
}
```

Beispiel n-1 unidirektional JPA


```
package com.hibernaterecipes.chapter5;
//imports
@Entity
//@Table(name = "BOOK", schema = "BOOK5")
public class Book_5_1 implements Serializable {
   @Id
   @Column(name = "BOOK_ID")
   @GeneratedValue(strategy = GenerationType.AUTO)
   private Long book_id;
   @Column(name = "isbn")
   private String isbn;
   @Column(name = "BOOK_NAME")
   private String name;
   @ManyToOne
   @Cascade(value = CascadeType.SAVE_UPDATE)
   @JoinColumn(name = "PUBLISHER_ID")
   private Publisher publisher;
   @Column(name = "price")
   private Integer price;
   // getters and setters
}
```

Beispiel n-1 unidirektional Hibernate


```
package com.hibernaterecipes.chapter5;

import java.io.Serializable;

public class Publisher implements Serializable {
  private Long publisher_id;
  private String code;
  private String name;
   private String address;
  // getters and setters
}
```

Beispiel n-1 unidirektional hbm.xml


```
<!DOCTYPE hibernate-mapping PUBLIC
"-//Hibernate/Hibernate Mapping DTD 3.0//EN"
"http://hibernate.sourceforge.net/hibernate-mapping-3.0.dtd">
<hibernate-mapping package="com.hibernaterecipes.chapter5">
   <class name="Publisher" table="Publisher" schema="BOOK5">
    <id name="Publisher_id" type="long" column="PUBLISHER_ID">
             <generator class="native">
             </generator>
    </id>
    <property name="code" type="string">
             <column name="CODE" length="4" not-null="true" unique="true" />
    </property>
    <property name="name" type="string">
             <column name="PUBLISHER_NAME" length="100" not-null="true" />
    </property>
    <property name="address" type="string">

             <column name="ADDRESS" length="200" />
    </property>
   </class>
</hibernate-mapping>
```

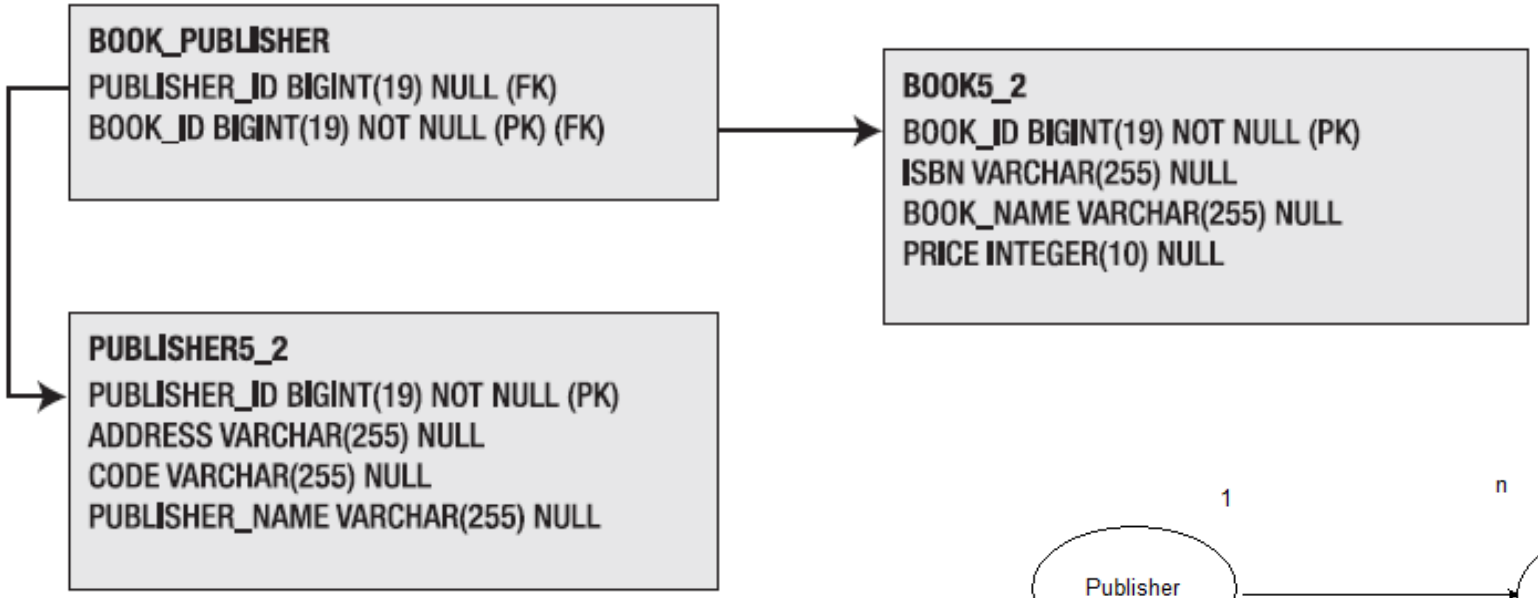
Beispiel n-1 unidirektional Hibernate


```
package com.hibernaterecipes.chapter5;

import java.io.Serializable;
import java.util.Date;

public class Book_5_1 implements Serializable {
   private Long book_id;
   private String isbn;
   private String name;
   private Publisher publisher;
   private Date publishDate;
   private Integer price;
   // getters and setters
}
```

Beispiel n-1 unidirektional hbm.xml-Mapping


```
<!DOCTYPE hibernate-mapping PUBLIC
"-//Hibernate/Hibernate Mapping DTD 3.0//EN"
"http://hibernate.sourceforge.net/hibernate-mapping-3.0.dtd">
<hibernate-mapping package="com.hibernaterecipes.chapter5">
   <class name="Book_5_1" table="Book" schema="BOOK5">
    <id name="book_id" type="long" column="BOOK_ID">
             <generator class="native">
             </generator>
    </id>
    <property name="isbn" type="string">
             <column name="ISBN" length="50" not-null="true" unique="true" />
    </property>
    <property name="name" type="string">
             <column name="BOOK_NAME" length="100" not-null="true" />
    </property>
    <property name="publishDate" type="date" column="PUBLISH_DATE" />
    <property name="price" type="int" column="PRICE" />
    <many-to-one name="publisher" class="Publisher" column="PUBLISHER_ID"
             cascade="save-update" />
   </class>
</hibernate-mapping>
```

Beispiel n-1 unidirektional Join Table



Beispiel n-1 unidirektional Join Table JPA


```
package com.hibernaterecipes.chapter5;
// imports
@Entity
//@Table(name = "BOOK5_2")
public class Book_5_2 implements Serializable {
   // …….
   @ManyToOne
   @Cascade(value = CascadeType.SAVE_UPDATE)
   @JoinTable( name = "BOOK_PUBLISHER", joinColumns =
   @JoinColumn(name = "BOOK_ID"), inverseJoinColumns =
   @JoinColumn(name = "PUBLISHER_ID"))
   private Publisher5_2 publisher
    // getters and setters
}
```

Beispiel n-1 unidirektional Join Table Hibernate


```
<!DOCTYPE hibernate-mapping PUBLIC
"-//Hibernate/Hibernate Mapping DTD 3.0//EN"
"http://hibernate.sourceforge.net/hibernate-mapping-3.0.dtd">
<hibernate-mapping package="com.hibernaterecipes.chapter5">
<class name="Book_5_2" table="Book5_2" >
   <id name="book_id" type="long" column="BOOK_ID"
    <generator class="native">
    </generator>
   </id>
   <property name="isbn" type="string">
    <column name="ISBN" length="50" not-null="true" unique="true" />
   </property>
   <property name="name" type="string">
    <column name="BOOK_NAME" length="100" not-null="true" />
   </property>
   <property name="publishDate" type="date" column="PUBLISH_DATE" />
   <property name="price" type="int" column="PRICE" />
   <join table="BOOK_PUBLISHER" optional="true" >
    <key column="BOOK_ID" unique="true" />
    <many-to-one name="publisher" class="Publisher5_2" column="PUBLISHER_ID"
             not-null="true" cascade="save-update" lazy="false" />
   </join>
</class>
</hibernate-mapping>
```

Agenda

Assoziationen

Cascading

n-1

1-n

1-1

m-n

Komponenten

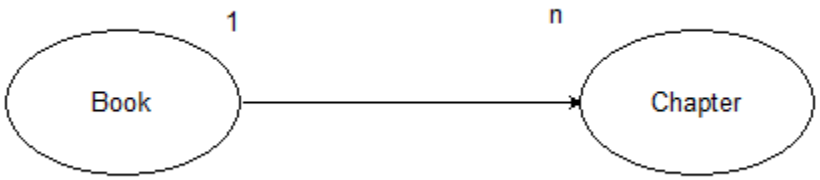
1:n Assoziation

- Beispiel:

- Book - Chapter
- 1 : n

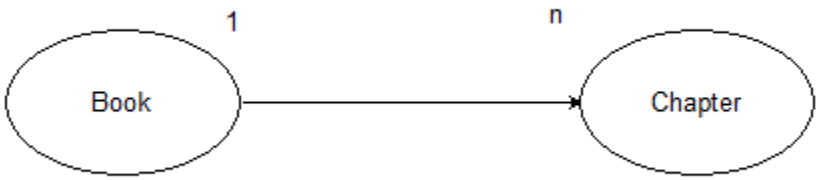
1:n Assoziation unidirektional Hibernate


```
package com.hibernaterecipes.chapter7;

import java.util.Date;
import java.util.Set;

public class Book {
  ..//

  private Set chapters;

  // getters and setters
}
```



1:n Assoziation unidirektional Hibernate


```
package com.hibernaterecipes.chapter7;

public class Chapter {
  private Long id;
  private String title;
  private int noOfPages

// getters and setters
}
```



1:n Assoziation unidirektional Hibernate


```
<!DOCTYPE hibernate-mapping PUBLIC
"-//Hibernate/Hibernate Mapping DTD 3.0//EN"
"http://hibernate.sourceforge.net/hibernate-mapping-3.0.dtd">
<hibernate-mapping package="com.hibernaterecipes.chapter7">
   <class name="Book" table="BOOK" >
    <id name="book_id" column="BOOK_ID" type="long">
            <generator class="native">
            </generator>
    </id>
    <property name="isbn" type="string" column="ISBN" />
    <property name="bookName" type="string" column="BOOK_NAME" />
    <property name="publishDate" type="date" column="PUBLISH_DATE" />
    <property name="price" type="long" column="PRICE" />
    <set name="chapters">
            <key column="BOOK_ID" />
            <one-to-many class="Chapter" />
    </set>
   </class>
</hibernate-mapping>
```

1:n Assoziation u… Hiberna…


```
<!DOCTYPE hibernate-mapping PUBLIC
"-//Hibernate/Hibernate Mapping DTD 3.0//EN"
"http://hibernate.sourceforge.net/hibernate-mapping-3.0.dtd">
<hibernate-mapping package="com.hibernaterecipes.chapter7">
<class name="Chapter" table="CHAPTER" dynamic-insert="true"
  dynamic-update="true“>
     <id name="id" column="id" type="long">
        <generator class="native">
        </generator>
     </id>
     <property name="title" type="string" column="title" />
     <property name="noOfPages" type="int"
    column="NUM_OF_PAGES" />
</class>
</hibernate-mapping>
```

1:n Assoziation unidirektional JPA


```
package com.hibernaterecipes.chapter7;

// imports

@Entity(name = "bkch2")
@Table(name = "BOOK7_1" )
public class Book {
  //..
  @OneToMany(targetEntity = Chapter.class)
  @JoinColumn(name = "book_id")
  @Cascade(value = { CascadeType.SAVE_UPDATE,
  CascadeType.DELETE_ORPHAN })
  Set chapters=new HashSet();
  // setters and gettes
}
```

1:n Assoziation bidirektional Hibernate


```
<!DOCTYPE hibernate-mapping PUBLIC
"-//Hibernate/Hibernate Mapping DTD 3.0//EN"
"http://hibernate.sourceforge.net/hibernate-mapping-3.0.dtd">
<hibernate-mapping package="com.hibernaterecipes.chapter7">
   <class name="Book" table="BOOK">
    <id name="book_id" column="BOOK_ID" type="long">
            <generator class="native">
            </generator>
    </id>
    <property name="isbn" type="string" column="ISBN" />
    <property name="bookName" type="string" column="BOOK_NAME" />
    <property name="publishDate" type="date" column="PUBLISH_DATE" />
    <property name="price" type="long" column="PRICE" />
    <set name="chapters" inverse="true">
            <key column="BOOK_ID" />
            <one-to-many class="Chapter" />
    </set>
   </class>
</hibernate-mapping>
```

1:n Assoziation bidirektional Hibernate


```
<!DOCTYPE hibernate-mapping PUBLIC
"-//Hibernate/Hibernate Mapping DTD 3.0//EN"
"http://hibernate.sourceforge.net/hibernate-mapping-3.0.dtd">
<hibernate-mapping package="com.hibernaterecipes.chapter7">
   <class name="Chapter" table="CHAPTER" dynamic-insert="true"
    dynamic-update="true" >
    <id name="id" column="id" type="long">
            <generator class="native">
            </generator>
    </id>
    <property name="title" type="string" column="title" />
    <property name="noOfPages" type="int" column="NUM_OF_PAGES" />
    <many-to-one name="book" column="book_id" class="Book"></many-to-
   one>
   </class>
</hibernate-mapping>
```

1:n Assoziation bidirektional JPA


```
package com.hibernaterecipes.chapter7;

//..
@Entity(name = "chapter")
@Table(name = "Chapter7_1")
public class Chapter {
  @ManyToOne
  //@JoinColumn(name = "book_id")
  private Book book;

  // getters and setters
}
```

1:n Assoziation bidirektional JPA


```
package com.hibernaterecipes.chapter7;
//imports
@Entity
@Table(name = "BOOK7_1")
public class Book {
  //..

  @OneToMany(targetEntity = Chapter.class)
  @JoinColumn(name = "BOOK_ID")
  @Cascade(value = { CascadeType.SAVE_UPDATE,
  CascadeType.DELETE_ORPHAN })
  Set chapters=new HashSet();
  // getters and setters
}
```

1:n Assoziation unidirektional Hibernate in Tables


```
package com.hibernaterecipes.chapter7;

public class Chapter {
  private Long id;
  private String title;
  private int noOfPages;
  // getters and setters
}
```

1:n Assoziation unidirektional Hibernate in Table


```
package com.hibernaterecipes.chapter7;

import java.util.Date;
import java.util.Set;

public class Book {
  private Long book_id;
  private String isbn;
  private String bookName;
  private Date publishDate;
  private Long price;
  private Set chapters;
  // getters and setters
}
```

1:n Assoziation bidirektional Hibernate in Tables


```
<!DOCTYPE hibernate-mapping PUBLIC
"-//Hibernate/Hibernate Mapping DTD 3.0//EN"
"http://hibernate.sourceforge.net/hibernate-mapping-3.0.dtd">
<hibernate-mapping package="com.hibernaterecipes.chapter7">
  <class name="Chapter" table="CHAPTER" dynamic-insert="true"
   dynamic-update="true">
   <id name="id" column="id" type="long">
          <generator class="native">
          </generator>
   </id>
   <property name="title" type="string" column="title" />
   <property name="noOfPages" type="int"
  column="NUM_OF_PAGES" />
  </class>
</hibernate-mapping>
```

1:n Assoziation bidirektional Hibernate in Tables


```
<!DOCTYPE hibernate-mapping PUBLIC
"-//Hibernate/Hibernate Mapping DTD 3.0//EN"
"http://hibernate.sourceforge.net/hibernate-mapping-3.0.dtd">
<hibernate-mapping package="com.hibernaterecipes.chapter7">
   <class name="Book" table="BOOK">
    <id name="book_id" column="BOOK_ID" type="long">
            <generator class="native">
            </generator>
    </id>
    <property name="isbn" type="string" column="ISBN" />
    <property name="bookName" type="string" column="BOOK_NAME" />
    <property name="publishDate" type="date" column="PUBLISH_DATE" />
    <property name="price" type="long" column="PRICE" />
    <set name="chapters" table="BOOK_CHAPTER" cascade="save-update,
delete-orphan">
            <key column="BOOK_ID" />
            <many-to-many column="CHAPTER_ID" class="Chapter"
                    unique="true" />
    </set>
   </class>
</hibernate-mapping>
```

1:n Assoziation bidirektional Hibernate Join Table


```
<hibernate-mapping package="com.hibernaterecipes.chapter7">
   <class name="Book" table="BOOK">

<set name="chapters" table="BOOK_CHAPTER" cascade="save-update,
delete-orphan">
            <key column="BOOK_ID" />
            <many-to-many column="CHAPTER_ID" class="Chapter"
                    unique="true" />
    </set>
   </class>
</hibernate-mapping>
```

- Anstatt one-to-many nutzt man **<many-to-many>,** da one-to-many nichts über Join Tables weiss.
- Mit **unique attribute** auf true sagt man dass ein **Book** ein **Chapter** nur einmal besitzen kann (durch hashCode() und equals()) und man implementiert dadurch eine one-to-many Association.

1:n Assoziation unidirektional JPA Join Table


```
package com.hibernaterecipes.chapter7;

// imports
@Entity
@Table(name = "BOOK7_3")
public class Book {
   //..
   @OneToMany(targetEntity = Chapter.class)
   @JoinTable(name = "Book_Chapter",  joinColumns =
   { @JoinColumn(name = "book_id") })
   @Cascade(value = { CascadeType.SAVE_UPDATE,
   CascadeType.DELETE_ORPHAN })
   private Set chapters=new HashSet();

   // getters and setters
}
```

1:n Assoziation bidirektional Hibernate in Tables


```
<!DOCTYPE hibernate-mapping PUBLIC
"-//Hibernate/Hibernate Mapping DTD 3.0//EN"
"http://hibernate.sourceforge.net/hibernate-mapping-3.0.dtd">
<hibernate-mapping package="com.hibernaterecipes.chapter7">
        <class name="Book" table="BOOK">
                <id name="book_id" column="BOOK_ID" type="long">
                        <generator class="native">
                        </generator>
                </id>
                <property name="isbn" type="string" column="ISBN" />
                <property name="bookName" type="string" column="BOOK_NAME" />
                <property name="publishDate" type="date"
column="PUBLISH_DATE" />
                <property name="price" type="long" column="PRICE" />
                <set name="chapters" table="BOOK_CHAPTER" cascade="save-update,
                                delete-orphan">
                        <key column="BOOK_ID" />
                        <many-to-many column="CHAPTER_ID" class="Chapter"
                                unique="true" />
                </set>
        </class>
</hibernate-mapping>
```

1:n Assoziation bidirektional Hibernate in Tables


```
<!DOCTYPE hibernate-mapping PUBLIC
"-//Hibernate/Hibernate Mapping DTD 3.0//EN"
"http://hibernate.sourceforge.net/hibernate-mapping-3.0.dtd">
<hibernate-mapping package="com.hibernaterecipes.chapter7">
        <class name="Chapter" table="CHAPTER" dynamic-insert="true"
                dynamic-update="
                        true" schema="BookShop7">
                 id name="id" column="id" type="long">
                        <generator class="native">
                        </generator>
                 /id>
                <property name="title" type="string" column="title" />
                <property name="noOfPages" type="int" column="NUM_OF_PAGES" />
                <join table="BOOK_CHAPTER" optional="true" inverse="true" >
                        <key column="CHAPTER_ID" unique="true" />
                        <many-to-one name="book" class="Book" column="BOOK_ID"
                                not-null="true" />
                </join>
        </class>
</hibernate-mapping>
```

1:n Assoziation bidirektional JPA Join Table


```
package com.hibernaterecipes.chapter7;
// imports

@Entity(name = "bkch73")
@Table(name = "BOOK7_39")
public class Book {
  //..
  @OneToMany(targetEntity = Chapter.class)
  @JoinTable(name = "Book_Chapter", joinColumns =
  { @JoinColumn(name = "book_id") })
  @Cascade(value = { CascadeType.SAVE_UPDATE,
  CascadeType.DELETE_ORPHAN })
  private Set chapters = new HashSet();

// getter & setter
}
```

Aufgabe

- Aufgabe 4:
- Beziehungen (n-1 und 1-n)

1:n Assoziation bidirektional JPA Join Table


```
package com.hibernaterecipes.chapter7;
//imports

@Entity
@Table(name = "Chapter7_30")
public class Chapter {
  //..
  @ManyToOne
  @JoinTable(name = "Book_Chapter5",  joinColumns =
  @JoinColumn(name = "id"))
  @JoinColumn(name = "book_id")
  private Book book;
   // getters and setters
}
```

Agenda

Assoziationen

Cascading

n-1

1-n

1-1

m-n

Komponenten

Beispiel Sharing Primary Key Hibernate


```
package com.hibernaterecipes.chapter5;
// imports
public class Customer5_1 implements Serializable {
  private static final long serialVersionUID =
  -3534434932962734600L;
  private Long id;
  private String countryCode;
  private String idCardNo;
  private String firstName;
  private String lastName;
  private Address5_1 address;
  private String email;
  // getters and setters
}
```

Beispiel Sharing Primary Key Hibernate


```
<!DOCTYPE hibernate-mapping PUBLIC
"-//Hibernate/Hibernate Mapping DTD 3.0//EN"
"http://hibernate.sourceforge.net/hibernate-mapping-3.0.dtd">
<hibernate-mapping package="com.hibernaterecipes.chapter5">
   <class name="Customer5_1" table="CUSTOMER" >
    <id name="id" type="long" column="ID">
            <generator class="native"></generator>
    </id>
    <property name="firstName" type="string" column="FIRST_NAME" />
    <property name="lastName" type="string" column="LAST_NAME" />
    <property name="idCardNo" type="string" column="ID_CARD_NO" />
    <property name="countryCode" type="string"
   column="COUNTRY_CODE" />
    <property name="email" type="string" column="EMAIL" />
    <one-to-one name="address"
   class="com.hibernaterecipes.chapter5.Address5_1"
            cascade="save-update"></one-to-one>
   </class>
</hibernate-mapping>
```

Beispiel Sharing Primary Key Hibernate


```
package com.hibernaterecipes.chapter5;

import java.io.Serializable;

public class Address5_1 implements Serializable {
   private static final long serialVersionUID =
   -6054747662873145 91L;
   private Long id;
   private String city;
   private String street;
    }
   private String doorplate;
   private Customer5_1 customer;
   // getters and setters
}
```

Beispiel Sharing Primary Key Hibernate


```
<!DOCTYPE hibernate-mapping PUBLIC
"-//Hibernate/Hibernate Mapping DTD 3.0//EN"
"http://hibernate.sourceforge.net/hibernate-mapping-3.0.dtd">
<hibernate-mapping package="com.hibernaterecipes.chapter5">
   <class name="Address5_1" table="ADDRESS" >
    <id name="id" type="long" column="ID">
            <generator class="foreign">
                    <param name="property">customer</param>
            </generator>
    </id>
    <property name="city" type="string" column="CITY" />
    <property name="street" type="string" column="STREET" />
    <property name="doorplate" type="string" column="DOOR_PLATE" />
    <one-to-one name="customer" class="Customer5_1"
            constrained="true">
    </one-to-one>
   </class>
</hibernate-mapping>
```

Beispiel Sharing Primary Key JPA


```
package com.hibernaterecipes.chapter5;
// imports
@Entity
@Table(name = "ADDRESS")
public class Address5_1 implements Serializable {
   private static final long serialVersionUID = -6054747662873145 91L;
   @Id
   @Column(name = "ADDRESS_ID")
   private Long id;
   @Column(name = "CITY")
   private String city;
   @Column(name = "STREET")
   private String street;
   @Column(name = "DOOR_PLATE")
   private String doorplate;
   // getters and setters
}
```

Beispiel Sharing Primary Key JPA


```
package com.hibernaterecipes.chapter5;

//imports

@Entity
@Table(name = "CUSTOMER")
public class Customer5_1 implements Serializable {

   /…

   @OneToOne
   @PrimaryKeyJoinColumn(name = "ID")
   private Address5_1 address;
   // getters and setters
}
```

Beispiel Sharing Primary Key JPA

- **Achtung**
- Werden **Address-Objekte** persistiert
  - muss manuell hierzu die Customer-Id gesetzt werden.

Beispiel 1-1 bidirektional Hibernate


```
<!DOCTYPE hibernate-mapping PUBLIC
"-//Hibernate/Hibernate Mapping DTD 3.0//EN"
"http://hibernate.sourceforge.net/hibernate-mapping-3.0.dtd">
<hibernate-mapping package="com.hibernaterecipes.chapter5">
   <class name="Customer5_2" table="CUSTOMER" schema="BOOK5">
    <id name="id" type="long" column="ID">
            <generator class="native"></generator>
    </id>
    <property name="firstName" type="string" column="FIRST_NAME" />
    <property name="lastName" type="string" column="LAST_NAME" />
    <property name="idCardNo" type="string" column="ID_CARD_NO" />
    <property name="countryCode" type="string" column="COUNTRY_CODE" />
    <property name="email" type="string" column="EMAIL" />
    <many-to-one name="address"
            class="com.hibernaterecipes.chapter5.Address5_2"
   column="ADDRESS_ID"
            cascade="save-update" unique="true">
    </many-to-one>
   </class>
</hibernate-mapping>
```

Macht es zu einer 1:1 Beziehung

Beispiel 1-1 bidirektional Hibernate


```
<!DOCTYPE hibernate-mapping PUBLIC
"-//Hibernate/Hibernate Mapping DTD 3.0//EN"
"http://hibernate.sourceforge.net/hibernate-mapping-3.0.dtd">
<hibernate-mapping package="com.hibernaterecipes.chapter5">
<class name="Address5_2" table="ADDRESS" schema="BOOK5">
    <id name="id" type="long" column="ADDRESS_ID" >
                    <generator class="native“>
                    </generator>
    </id>
    <property name="city" type="string" column="CITY" />
    <property name="street" type="string" column="STREET" />
    <property name="doorplate" type="string" column="DOOR_PLATE" />
    <one-to-many name="customer" class="Customer5_2“
   propertyref="address">
    </one-to-many>
</class>
</hibernate-mapping>
```

Beispiel 1-1 bidirektional JPA

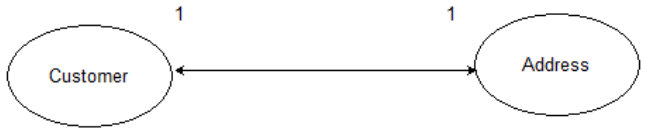

```
package com.hibernaterecipes.chapter5;
// imports
@Entity
@Table(name = "ADDRESS")
public class Address5_2 implements Serializable {
   //..
   @OneToOne(mappedBy = "address")
   private Customer5_2 customer;
   // getters and setters
}
```

Beispiel 1-1 bidirektional JPA


```
package com.hibernaterecipes.chapter5;
// imports

@Entity
@Table(name = "CUSTOMER")
public class Customer5_2 implements Serializable {
  //…

  @OneToOne(cascade = CascadeType.ALL)
  @JoinColumn(name = "ADDRESS_ID")
  private Address5_2 address;
   // getters and setters
}
```

Beispiel 1-1 bidirektional Hibernate Join Table


```
<!DOCTYPE hibernate-mapping PUBLIC
"-//Hibernate/Hibernate Mapping DTD 3.0//EN"
"http://hibernate.sourceforge.net/hibernate-mapping-3.0.dtd">
<hibernate-mapping package="com.hibernaterecipes.chapter5">
   <class name="Address5_3" table="Address" >
    <id name="addressId" type="long" column="ADDRESS_ID">
            <generator class="native">
            </generator>
    </id>
    <property name="city" type="string" column="CITY" />
    <property name="street" type="string" column="STREET" />
    <property name="doorplate" type="string" column="DOOR_PLATE" />
    <join table="CustomerAddress" optional="true" inverse="true">
            <key column="ADDRESS_ID" unique="true" />
            <many-to-one name="customer"
                    class="com.hibernaterecipes.chapter5.Customer5_3"
   column="ID"
                    unique="true" not-null="true">
            </many-to-one>
    </join>
   </class>
</hibernate-mapping>
```



Beispiel 1-1 bidirektional Hibernate in Tabelle


```
<!DOCTYPE hibernate-mapping PUBLIC
"-//Hibernate/Hibernate Mapping DTD 3.0//EN"
"http://hibernate.sourceforge.net/hibernate-mapping-3.0.dtd">
<hibernate-mapping package="com.hibernaterecipes.chapter5">
   <class name="Customer5_3" table="Customer" >
    <id name="id" type="long" column="ID">
             <generator class="native"></generato
    </id>
    <property name="firstName" type="string" column="FIRST_NAME" />
    <property name="lastName" type="string" column="LAST_NAME" />
    <property name="idCardNo" type="string" column="ID_CARD_NO" />
    <property name="countryCode" type="string" column="COUNTRY_CODE" />
    <property name="email" type="string" column="EMAIL" />
    <join table="CustomerAddress" optional="true" >
             <key column="ID" unique="true">
             </key>
             <many-to-one name="address" column="ADDRESS_ID" not-null="true"
                      cascade="save-update" unique="true">
             </many-to-one>
    </join>
   </class>
</hibernate-mapping>
```

Beispiel 1-1 bidirektional JPA Join Table


```
package com.hibernaterecipes.chapter5:
//imports
@Entity
@Table(name = "ADDRESS")
public class Address5_3 implements Serializable {

  //..
  @OneToOne(mappedBy = "address")
  private Customer5_3 customer;

// getters and setters
}
```

Beispiel 1-1 bidirektional JPA Join Table


```
package com.hibernaterecipes.chapter5;
// imports

@Entity
@Table(name = "CUSTOMER")
public class Customer5_3 implements Serializable {
   //..

   @OneToOne(cascade = CascadeType.ALL)
   @JoinTable(name = "CustomerAddress",  joinColumns =
   @JoinColumn(name = "ID"), inverseJoinColumns =
   @JoinColumn(name = "ADDRESS_ID"))
   private Address5_3 address;

   // getters and setters
}
```

Aufgabe

- Aufgabe 5: Beziehungen (1-1)

Agenda

Assoziationen

Cascading

n-1

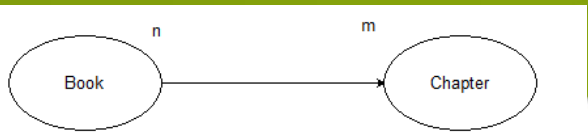
1-n

1-1

m-n

Komponenten

n:m Assoziation unidirektional Hibernate




```
<!DOCTYPE hibernate-mapping PUBLIC
"-//Hibernate/Hibernate Mapping DTD 3.0//EN"
"http://hibernate.sourceforge.net/hibernate-mapping-3.0.dtd">
<hibernate-mapping package="com.hibernaterecipes.chapter7">
   <class name="Book" table="BOOK" >
    <id name="book_id" column="BOOK_ID" type="long">
             <generator class="native">
             </generator>
    </id>
    <property name="isbn" type="string" column="ISBN" />
    <property name="bookName" type="string" column="BOOK_NAME" />
    <property name="publishDate" type="date" column="PUBLISH_DATE" />
    <property name="price" type="long" column="PRICE" />
    <set name="chapters" table="BOOK_CHAPTER"
             cascade="save-update,delete-orphan">
             <key column="BOOK_ID" />
             <many-to-many column="ID" class="Chapter" unique="true" />
    </set>
   </class>
</hibernate-mapping>
```

n:m Assoziation unidirektional Hibernate


```
<!DOCTYPE hibernate-mapping PUBLIC
"-//Hibernate/Hibernate Mapping DTD 3.0//EN"
"http://hibernate.sourceforge.net/hibernate-mapping-3.0.dtd">
<hibernate-mapping package="com.hibernaterecipes.chapter7">
  <class name="Chapter" table="CHAPTER7_4" dynamic-
  insert="true"
   dynamic-update="true" >
   <id name="id" column="id" type="long">
          <generator class="native">
          </generator>
   </id>
   <property name="title" type="string" column="title" />
   <property name="noOfPages" type="int"
  column="NUM_OF_PAGES" />
  </class>
</hibernate-mapping>
```

n:m Assoziation unidirektional JPA

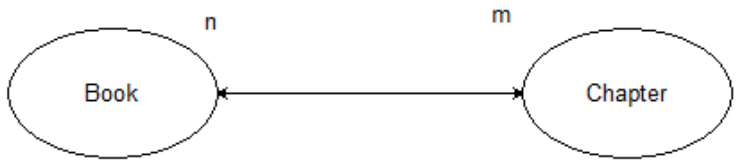

```
package com.hibernaterecipes.chapter7;

// imports
@Entity(name = "bkch74")
@Table(name = "BOOK77")
public class Book {
  //..
  @ManyToMany(targetEntity = Chapter.class)
  @JoinTable(name = "Book_Chapter22",  joinColumns = {
  @JoinColumn(name = "book_id") }, inverseJoinColumns
  = { @JoinColumn(name = "chapter_id") })
  @Cascade(value = { CascadeType.SAVE_UPDATE,
  CascadeType.DELETE_ORPHAN })
  private Set chapters=new HashSet();
  // setter & getter
}
```

n:m Assoziation bidirektional Hibernate in Tables


```
<!DOCTYPE hibernate-mapping PUBLIC
"-//Hibernate/Hibernate Mapping DTD 3.0//EN"
"http://hibernate.sourceforge.net/hibernate-mapping-3.0.dtd">
<hibernate-mapping package="com.hibernaterecipes.chapter7">
   <class name="Book" table="BOOK7_41" >
    <id name="book_id" column="BOOK_ID" type="long">
             <generator class="native">
             </generator>
    </id>
    <property name="isbn" type="string" column="ISBN" />
    <property name="bookName" type="string" column="BOOK_NAME" />
    <property name="publishDate" type="date" column="PUBLISH_DATE" />
    <property name="price" type="long" column="PRICE" />
    <set name="chapters" table="BOOK_CHAPTER"
             cascade="save-update,delete-orphan">
             <key column="BOOK_ID" />
             <many-to-many column="ID" class="Chapter" unique="true" />
    </set>
   </class>
</hibernate-mapping>
```

n:m Assoziation bidirektional Hibernate in Tables


```
<!DOCTYPE hibernate-mapping PUBLIC
"-//Hibernate/Hibernate Mapping DTD 3.0//EN"
"http://hibernate.sourceforge.net/hibernate-mapping-3.0.dtd">
<hibernate-mapping package="com.hibernaterecipes.chapter7">
   <class name="Chapter" table="CHAPTER" dynamic-insert="true"
    dynamic-update="true" >
    <id name="id" column="id" type="long">
            <generator class="native">
            </generator>
    </id>
    <property name="title" type="string" column="title" />
    <property name="noOfPages" type="int" column="NUM_OF_PAGES" />
    <set name="book" table="BOOK_CHAPTER" inverse="true"
   cascade="save-update">
            <key column="id"></key>
            <many-to-many class="Book" column="BOOK_ID" />
    </set>
   </class>
</hibernate-mapping>
```

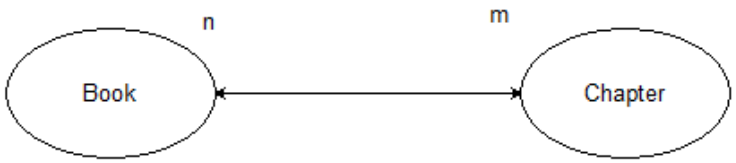


n:m Assoziation bidirektional JPA Join Table


```
package com.hibernaterecipes.chapter7;

@Entity(name = "bkch741")
@Table(name = "BOOK7_41666")
public class Book {

  @ManyToMany
  @JoinTable(name = "Book_Chapter",  joinColumns =
  { @JoinColumn(name = "book_id") },
  inverseJoinColumns = { @JoinColumn(name = "id") })
  @Cascade(value = { CascadeType.SAVE_UPDATE })
  private Set<Chapter> chapters = new
  HashSet<Chapter>();
  // getters and setters
  }
```

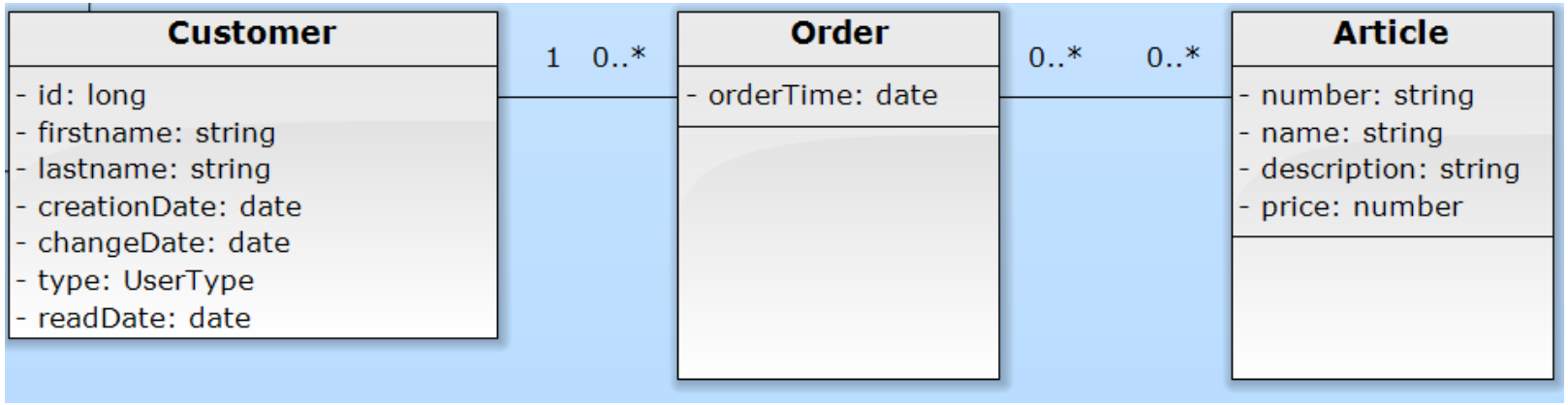
n:m Assoziation bidirektional JPA Join Table


```
package com.hibernaterecipes.chapter7;

//imports

@Entity(name = "chapter741")
@Table(name = "Chapter7_417777")
public class Chapter {
  //..
  @ManyToMany(mappedBy = "chapters")
  private Set<Book> book = new HashSet<Book>();

  // getters and setters
}
```



Aufgabe

- Aufgabe 6 (optional):
- Beziehungen (m-n)



Agenda

Assoziationen

Cascading

n-1

1-n

1-1

m-n

Komponenten

Component Mapping

- Entity-Komponente
  - haben einen Primärschlüssel
  - haben einen Lebenszyklus
- Value-Komponente
  - haben keine Datenbankidentität
  - haben keinen Primärschlüssel
  - gehören zu einer Entity und ihr Zustand wird innerhalb der Tabelle der dazugehörigen Entity gesichert
  - Ausnahme sind Collections von Value-Typen
  - typisch sind einfache Objekte vom Typ String
  - Lebensdauer eines Value-Typ ist ist immer an den Lebenszyklus der entsprechenden Entity gebunden
- Komponenten ermöglichen die Abbildung mehrerer Klassen auf eine Tabelle

Component Mapping

- Annotation @Embeddable definiert Komponente auf Klassenebene
- Kennzeichnung @Embedded beim entsprechenden Attribut bzw. bei der Getter-Methode

```
@Entity
public class Person {
   @Embedded
   private Address address;
   …
   // getter/setter
}
@Embeddable
public class Address implements Serializable{
```

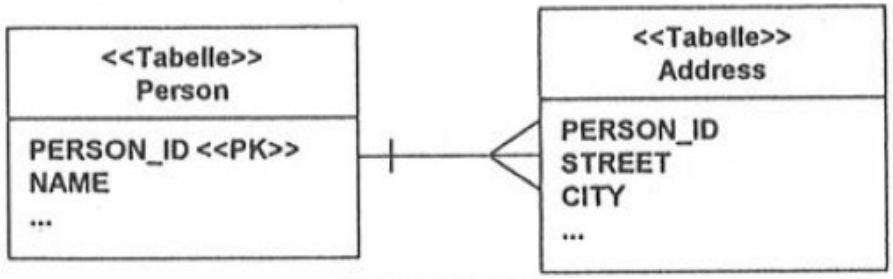
Component Mapping (2)

- Spalten können umbenannt werden

```
@Entity
public class Person {
   @Embedded
   @AttributeOverrides({
            @AttributeOverride(name="street",
                    column=@Column (name="strasse"))
            @AttributeOverride(name="city",
                    column=@Column(name="stadt"))
            @AttributeOverride(name="country",
                    column=@Column(name="land"))
})
private Address address;
…
}
```

Collection von Components

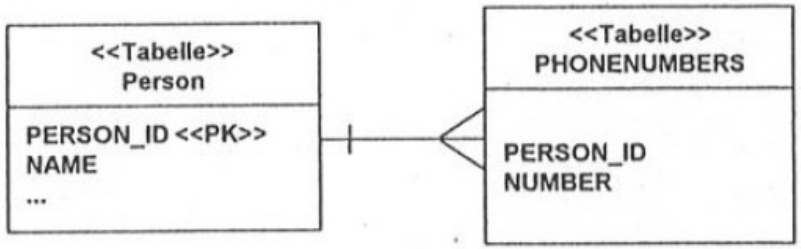
- eigene Tabelle ohne PK (JPA2)




```
@Entity
public class Person {
   @ElementCollection
   private List<Address> addresses;
   …
   // getter/setter
}
@Embeddable
public class Address implements Serializable{
   …
}
```

Collection von Basistypen

- eigene Tabelle ohne PK




```
@Entity
public class Person {
  @ElementCollection
  private List<String> phoneNumbers;
  …
// getter/setter
```

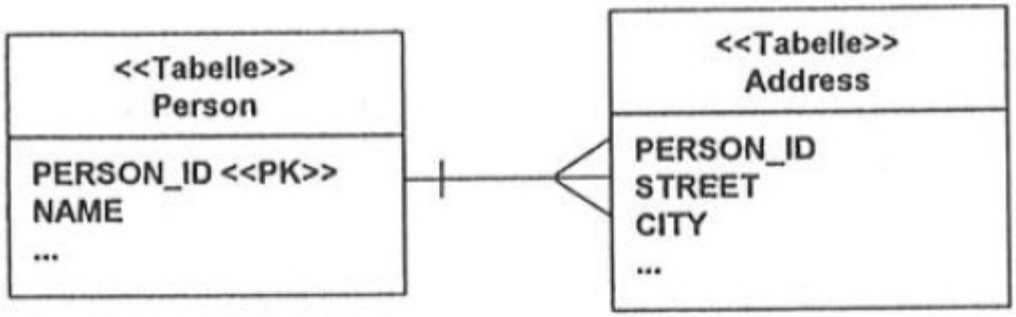
Component Mapping (Hibernate)


```
<class name=" Person ">
   <id name="id">
       <generator .. />
   </id>
   <property name="name" />
   <component name="address"
   class="Address">
       <property name="street" />
       <property name="city" />
       <parent name="person"/>
   </component>
</class>
```

- Subkomponenten und Assoziationen zu anderen Komponenten
- keine geteilten Referenzen!
- Mapping erlaubt Rückreferenz mit <parent />

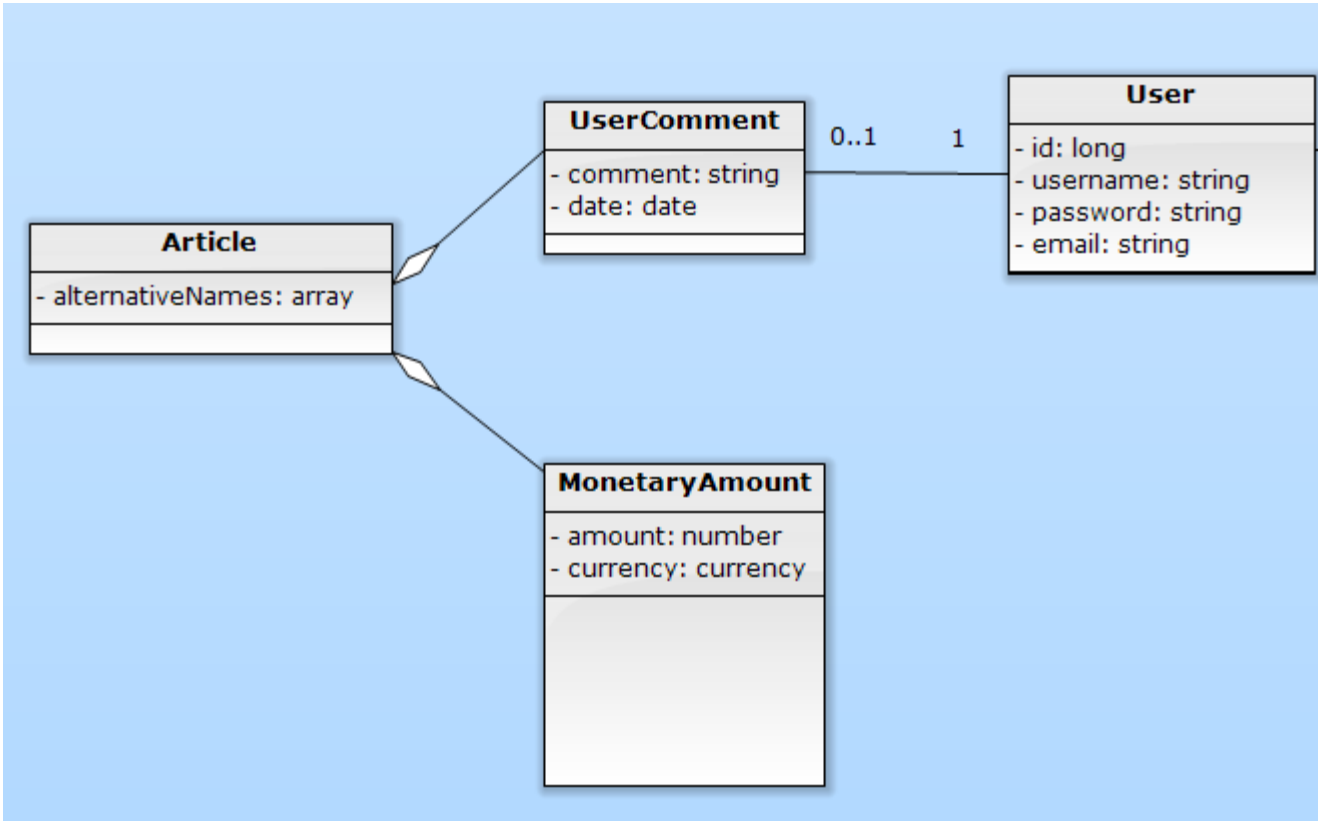
Collections von Komponenten (Hibernate)


```
<class name="Person">
   <id name="person_id">
       <generator .. />
   </id>
   <property name="name" />
   <set name="addresses" table="addresses" lazy="true">
       <key column="person_id"/>
       <composite-element class="Address">
             <property name="street"/>
             <property name="city"/>
       </composite-element>
   </set>
</class>
```



Aufgabe

- Aufgabe 7 (optional):
- Komponenten



@SecondaryTable (1)

- Partitionierung der Attribute einer Klasse in zwei (oder mehr) Tabellen

```
@Entity
@Table(name = "Person")
@SecondaryTable(name =
  "EMBEDDED_ADDRESS",      …)
public class Person implements
  Serializable {
…
```

@SecondaryTable (2)

- Ausgewählte Spalten können nun in die zweite Tabelle gelegt werden

```
@Embeddable
public class Address implements Serializable{
…
@Column(name = "STREET", table = "EMBEDDED_ADDRESS")
private String street
}
```