# Description for One to Many Bi-Directional Association Task:

1. CONFIGURATION FILE
   1. Hibernate.cfg.xml

<?xml version=*'1.0'* encoding=*'utf-8'*?>

<!DOCTYPE hibernate-configuration PUBLIC

"-//Hibernate/Hibernate Configuration DTD 3.0//EN"

"http://www.hibernate.org/dtd/hibernate-configuration-3.0.dtd">

<hibernate-configuration>

<session-factory>

<property name=*"connection.driver\_class"*>com.mysql.jdbc.Driver</property>

**Explanation:**

Define the driver class

<property name=*"connection.url"*>jdbc:mysql://localhost:3306/restful\_crud </property>

**Explanation:**

Define url to the database instance

<property name=*"connection.username"*>sapphire</property>

**Explanation:**

Username of the database

<property name=*"connection.password"*>sapphire123</property>

**Explanation:**

Password of the database

<property name=*"dialect"*>org.hibernate.dialect.MySQLDialect</property>

**Explanation:**

Dialect property is responsible to create appropriate SQL for the database

<property name=*"show\_sql"*>true</property>

**Explanation:**

If the value is true, the query will be displayed on console

<mapping resource=*"Products.hbm.xml"*/>

<mapping resource=*"Seller.hbm.xml"*/>

**Explanation:**

List of mapping files in case of xml mapping, if mapping is performed by using annotation class will be used instead of resource

</session-factory>

</hibernate-configuration>

1. MAPPING FILES
   1. User.hbm.xml

<?xml version=*"1.0"* encoding=*"UTF-8"*?>

<!DOCTYPE hibernate-mapping PUBLIC

"-//Hibernate/Hibernate Mapping DTD 3.0//EN"

"http://www.hibernate.org/dtd/hibernate-mapping-3.0.dtd">

<?xml version=*"1.0"* encoding=*"UTF-8"*?>

<!DOCTYPE hibernate-mapping PUBLIC

"-//Hibernate/Hibernate Mapping DTD 3.0//EN"

"http://hibernate.sourceforge.net/hibernate-mapping-3.0.dtd">

<hibernate-mapping package=*"com.scs.webservices.model"*>

**Explanation:**

Root element containing class elements. Package defines the class package

<class name=*"Products"* table=*"product"*>

**Explanation:**

In class tag, name defines model class containing getter/setter and table defines the database table.

<id name=*"prodId"* column=*"product\_id"*>

**Explanation:**

In id tag, name defines model class attribute and column defines the database column.

<generator class=*"native"*/>

**Explanation:**

Generator class is responsible for automatically primary key generation. Using native class we make hibernate to choose algorithm based on database capabilities

</id>

**Explanation:**

End of id tag

<property name=*"prodName"* type=*"string"* column=*"product\_name"*/>

**Explanation:**

In property tag, name defines model class attribute and column defines the database column.

<property name=*"prodActive"* type=*"int"* column=*"active"*/>

**Explanation:**

In property tag, name defines model class attribute and column defines the database column.

<property name=*"manufactureYear"* type=*"int"* column=*"manufacturing\_year"*/>

**Explanation:**

In property tag, name defines model class attribute and column defines the database column.

<property name=*"productExpiry"* type=*"timestamp"* column=*"expiry\_date"*/>

**Explanation:**

In property tag, name defines model class attribute and column defines the database column.

<many-to-one name=*"seller"* column=*"seller\_id"* class=*"com.scs.webservices.model.Seller"* cascade=*"all"* />

**Explanation:**

We define object “seller” that is used in our POJO class to create a relationship between two classes, and table name. Cascade option is used to select what operations can be performed on the table using parent class

</class>

**Explanation:**

End of class

</hibernate-mapping>

**Explanation:**

End of hibernate-mapping

* 1. Groups.hbm.xml

<hibernate-mapping package=*"com.scs.webservices.model"*>

<class name=*"Seller"* table=*"seller"*>

**Explanation:**

In class tag, name defines model class containing getter/setter and table defines the database table

<id name=*"sellerId"* column=*"seller\_id"*>

**Explanation:**

In id tag, name defines model class attribute and column defines the database column.

<generator class=*"native"*/>

**Explanation:**

Generator class is responsible for automatically primary key generation. Using native class we make hibernate to choose algorithm based on database capabilities

</id>

**Explanation:**

End of Id tag

<property name=*"sellerName"* type=*"string"* column=*"sellername"* />

**Explanation:**

In property tag, name defines model class attribute and column defines the database column.

<property name=*"sellerActive"* type=*"int"* column=*"active"* />

**Explanation:**

In property tag, name defines model class attribute and column defines the database column.

<set name=*"products"* table=*"product"* lazy=*"true"* fetch=*"select"*>

**Explanation:**

We define set that is used in our POJO class to create a relationship between two classes, and table name. Lazy=true means a collection is fetched when the application invokes an operation upon that collection. A second SELECT is used to retrieve the associated entity or collection.

<key column=*"seller\_id"* not-null=*"true"* />

**Explanation:**

Primary Key of the actual table

<one-to-many class=*"com.scs.webservices.model.Products"* />

**Explanation:**

Relationship and column on which the relation is created along with the class name

</set>

**Explanation:**

End of Set

</class>

**Explanation:**

End of class

</hibernate-mapping>

**Explanation:**

End of hibernate mapping