Note: Student table has following columns

Id, email, first\_name, last\_name

When using the set list student table and images table can relate like this below we just need to write the student entity class no need to write images entity class we need to map images in student entity like this below it will automatically create the images table.

* ***Set***

@ElementCollection

@CollectionTable(name="image", //defaults to student\_images

joinColumns = @JoinColumn(name="student\_id"))

@Column(name="file\_name") //defaults to images

**private** Set<String> images = **new** HashSet<String> ();

* ***SortedSet***

*OrderBy* order by we can use with set to order the element asc or desc

@ElementCollection

@CollectionTable(name="image")

@org.hibernate.annotations.OrderBy(clause = "file\_name desc" ) //default asc

@Column(name="file\_name") //defaults to images

**private** Set<String> images = **new** LinkedHashSet<String>();

* ***Map***

@ElementCollection

@CollectionTable(name="image")

@MapKeyColumn(name="file\_name")

@Column(name="image\_name")

**private** Map<String, String> images = **new** HashMap<String, String>();

@Embeddable we can use this on class whose property is common and use that class in entity class to automatically iject those propetery in one table

// The Address is @Embeddable, no annotation needed here...

**private** Address homeAddress;

//Overriding Embedded attributes

@AttributeOverrides({

@AttributeOverride(name="street", column=@Column(name="BILLING\_STREET")),

@AttributeOverride(name="zipcode", c column=@Column(name="BILLING\_ZIPCODE")),

@AttributeOverride(name="city", column=@Column(name="BILLING\_CITY")),

})

**private** Address billingAddress;

In the above example billing address and home address are ejected in student entity class those columns will get automatically created.

***Hibernate Inheritance mapping:***

* Single Table
* Table Per Class
* Joined table
* Mapped Superclass

***Single Table:***

The Single Table strategy creates one table for each class hierarchy. This is also the default strategy chosen by JPA if we don't specify one explicitly. We can define the strategy we want to use by adding the @Inheritance annotation to the super-class:

@Entity

@Table(name="user")

@Inheritance(strategy = InheritanceType.***SINGLE\_TABLE***)

@DiscriminatorColumn(name="TYPE\_OF\_USER", discriminatorType=DiscriminatorType.***STRING***)

**public** **abstract** **class** User {

@Entity

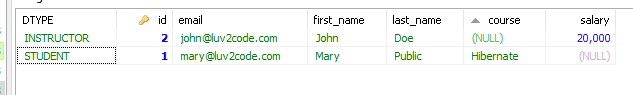
@DiscriminatorValue(value="STUDENT")

**public** **class** Student **extends** User {

@Entity

@DiscriminatorValue(value="INSTRUCTOR")

**public** **class** Instructor **extends** User {

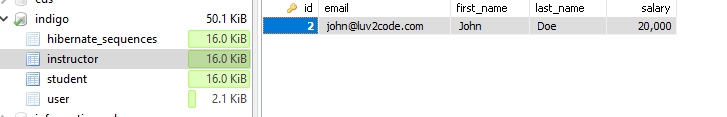


User Class is a super class, Student class is subclass which extend the user class and has only one field course, Instructer class is also extends the User class which has only one field salary the column dtype is auto generated and the name of column we can define in *DiscriminatorColumn* annotation.

So in Single table strategy only one table is created for all the concrete class.

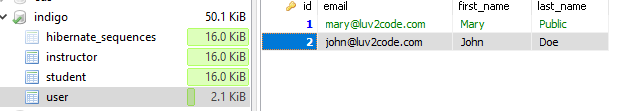
single table is required to map the whole hierarchy, an extra column (known as discriminator column) is added to identify the class

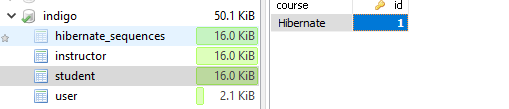
***Table Per Class:*** in table per class we use strategy TABLE\_PER\_CLASS in out super class.

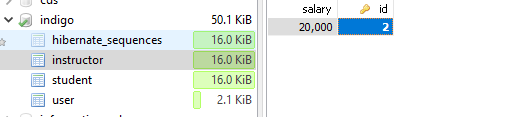


Each table will be created for concrete subclasses and in abstract super class for id field use generation type TABLE and hibernate will create the one more extra table hibernate\_sequences to maintain the auto increment id.

***Joined Table:*** in this stategey table will be created for each class in super class use generation strategy identity for subclasses it will be mapped with primary key in super class and foreign key in subclass which will be automatically created.



****

****

**Mapped Superclass:** in this strategy on subclasses will have table in database with super class columns we should use @Entity annotation only for subclasses not for super class for super class we need to define only inheritance.