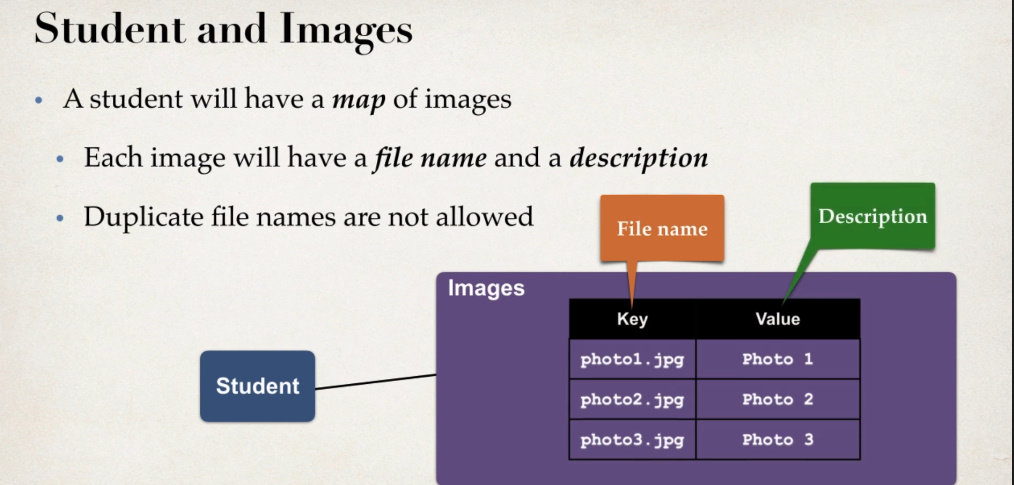
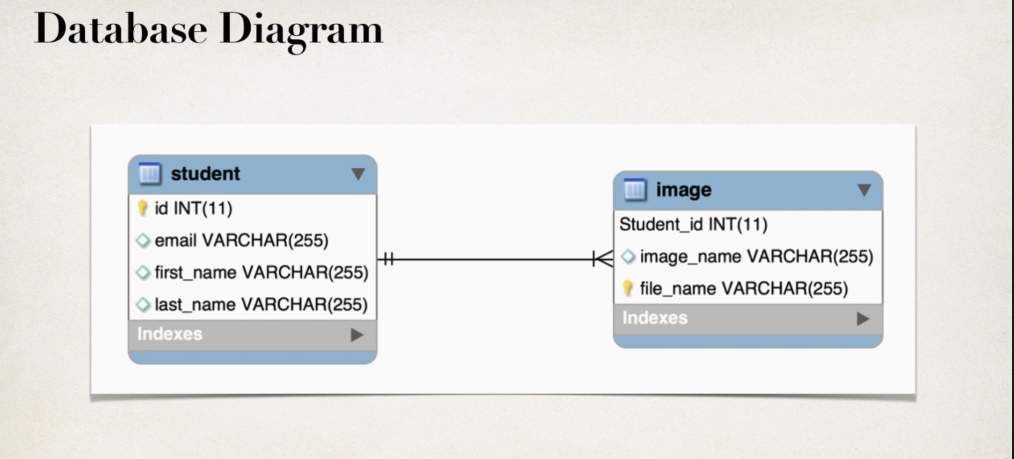


**Project**

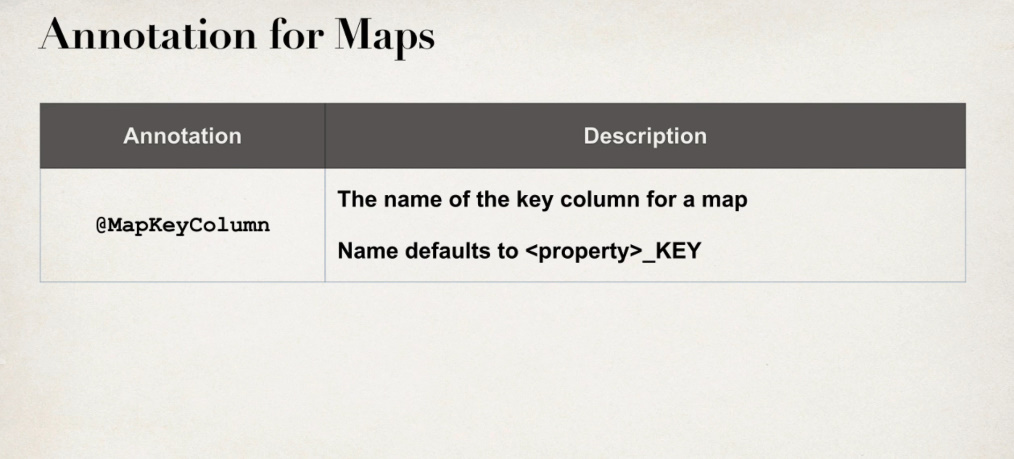


Ca exemplu, key va fi file\_name si value va fi image\_name. file\_name va fi primary jey



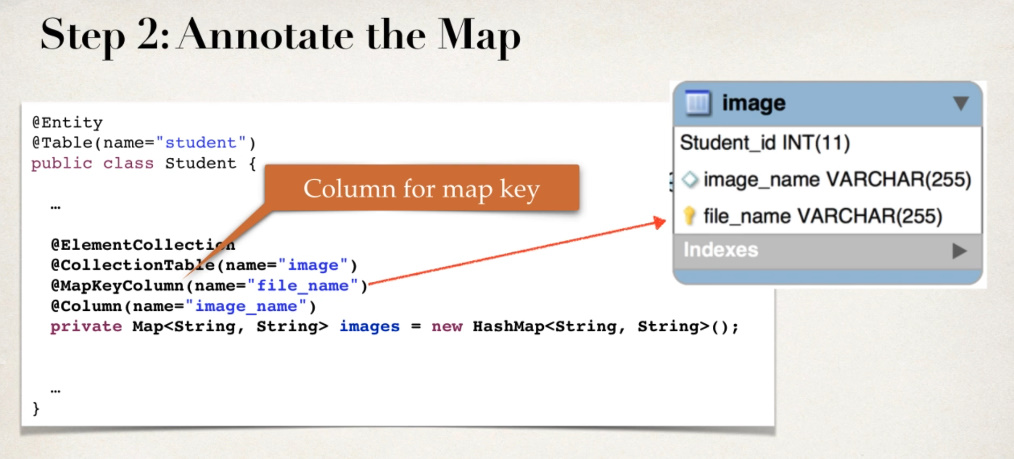
Deci, pentru un student se va returna o lista de Map<> ce vor avea ca Key file\_name si value image\_name, deci imaginile corespunzatoare acelui user cu numele la file si la imagine

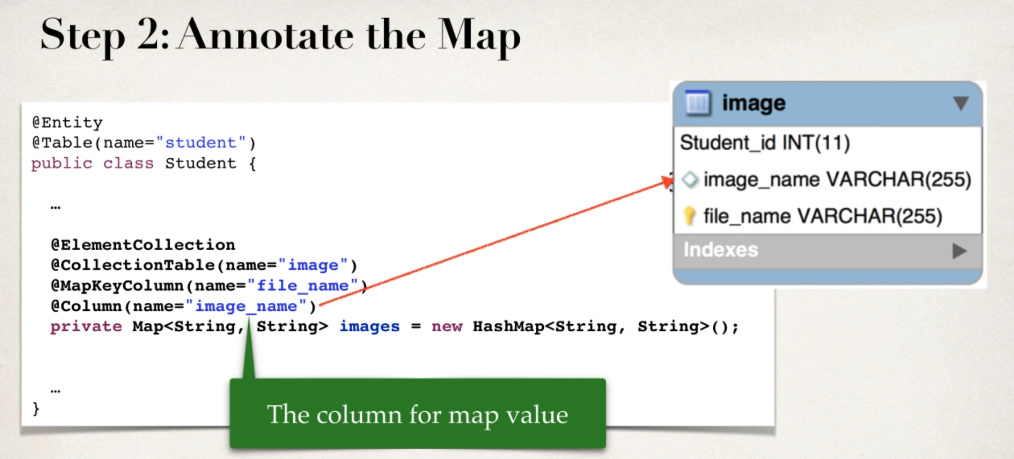
**@MapKeyColumn(name = “”)**



Aceasta anotatie indica coloana care va fi key. Ea trebuie sa accepte doar valori unice si sa fie primary key din alt tabel.

Numele default va fi <property>\_KEY



si @Column deja va fi numele coloanei pentru value  


@ElementCollection  
@CollectionTable(name = "image")  
@MapKeyColumn(name = "file\_name")  
@Column(name = "image\_name")  
private Map<String, String> images = new HashMap<>();

Deci, pur si simplu se vor cauta elementele din Image table unde Student\_id este egal cu id din tabelul Student, si tuples gasite vor fi puse in HashMap. @Column e value si @MapKeyColumn e coloana key

- file\_name – numele la imagine

- imagine\_name – descrierea la imagine

**Daca nu folosim joinColumns, automat se va lua numele la entity curent + \_id, gen student\_id, dar daca asa column nu exista, avem problema, de aceea mai bine scriiem.**

Acum, daca vom da asa payload:

{

    "firstName" : "Mititiuc",

    "lastName" : "Eduard",

    "email" : "test@mail.ru",

    "images" : {

        "image1.png" : "My image 1",

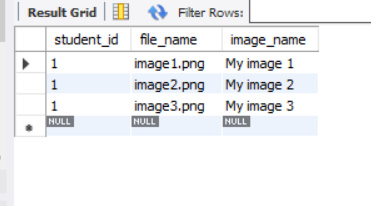
        "image2.png": "My image 2",

        "image3.png": "My image 3"

    }

}

In baza de date vom vedea asta:



* Iarasi, primary key din image nu e foreign key