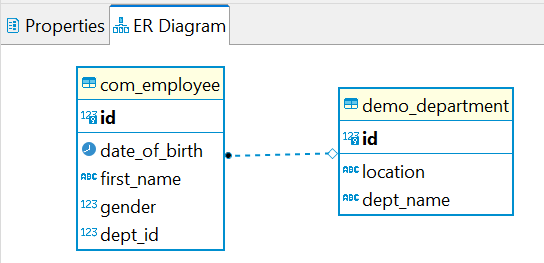
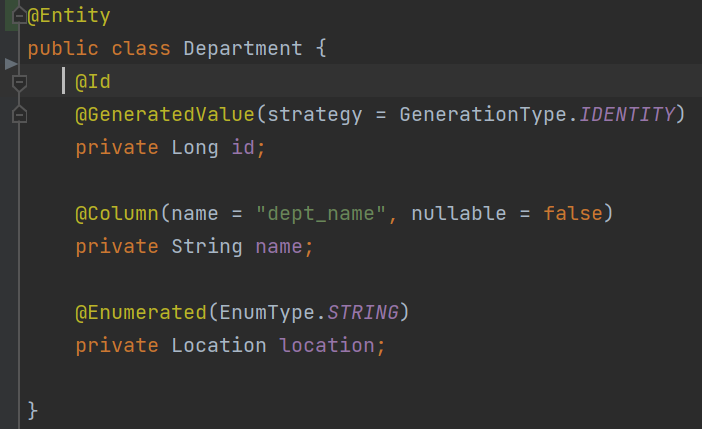
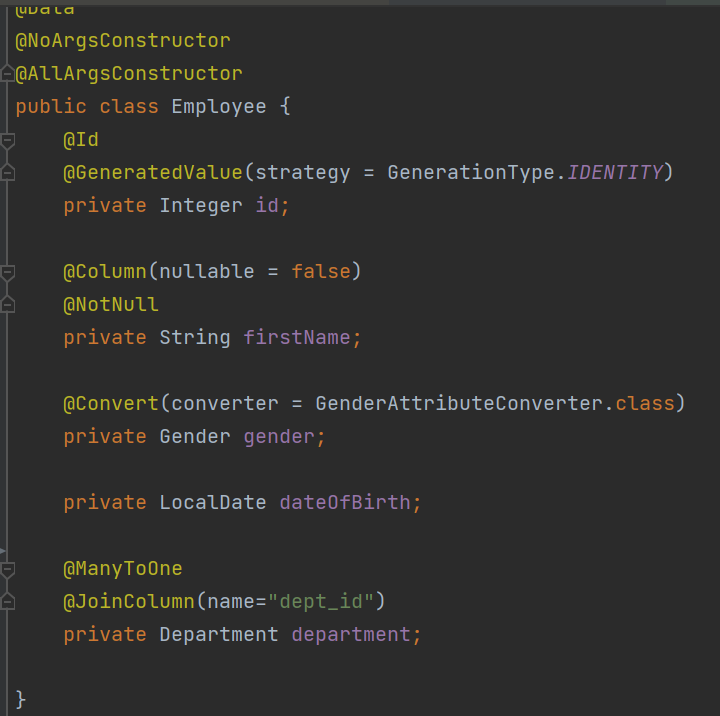
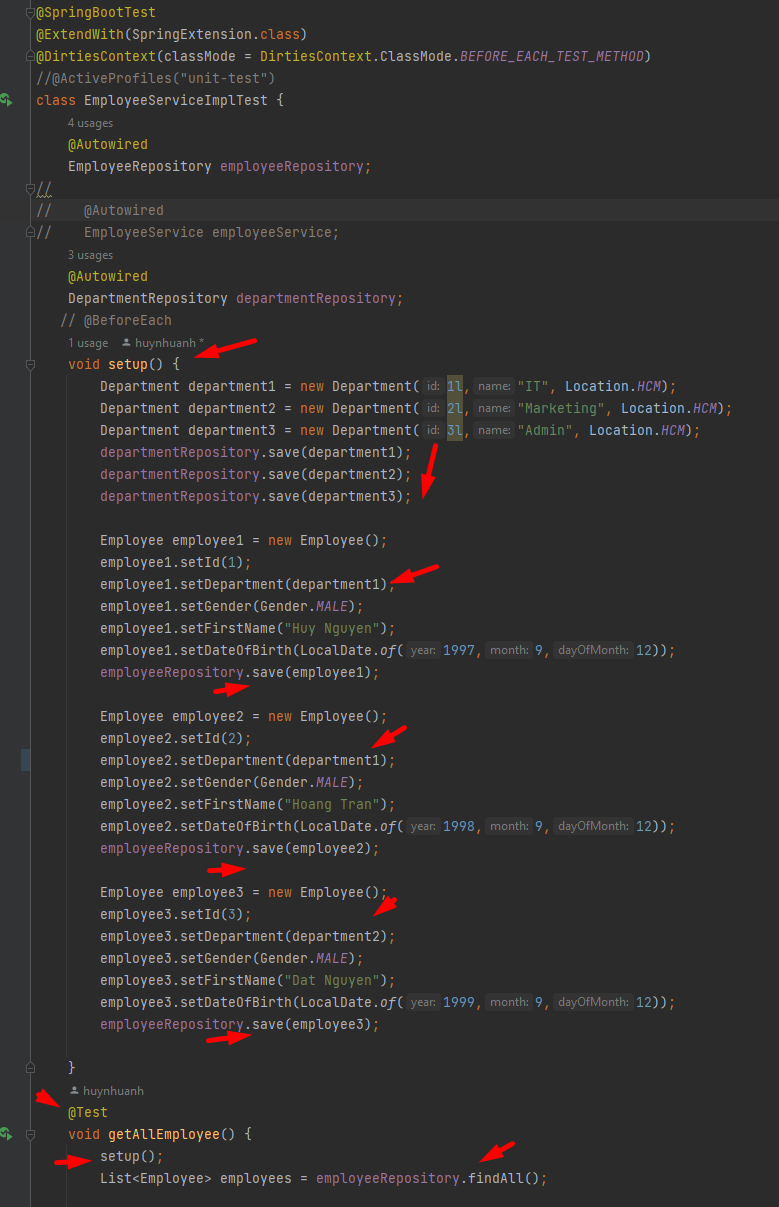
N+1 Problem



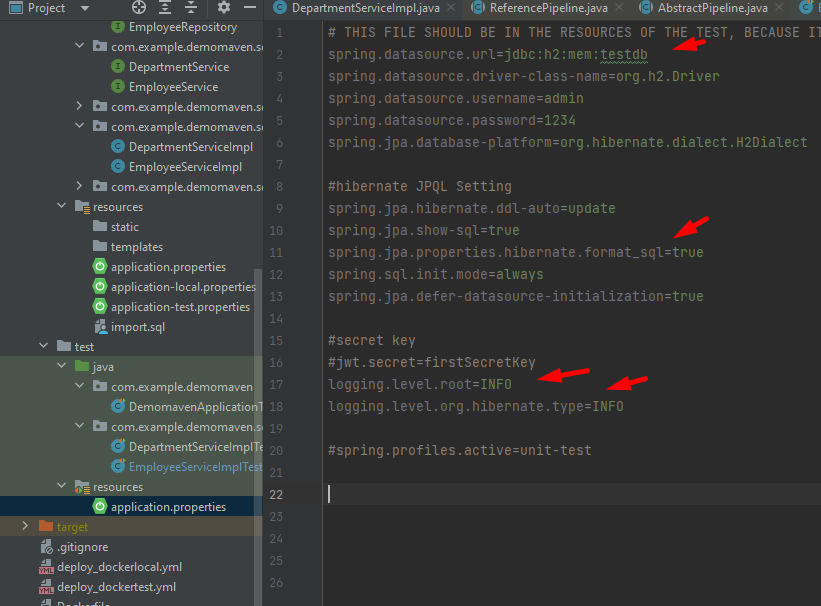
Write Unit test for EmployeeService. Connect to H2 Database and create a setup method to:

* Insert 3 Departments: IT, Marketing & Admin to Department table
* Insert 3 Employees: 2 first Employees in IT Department and 3rd belongs to Marketing

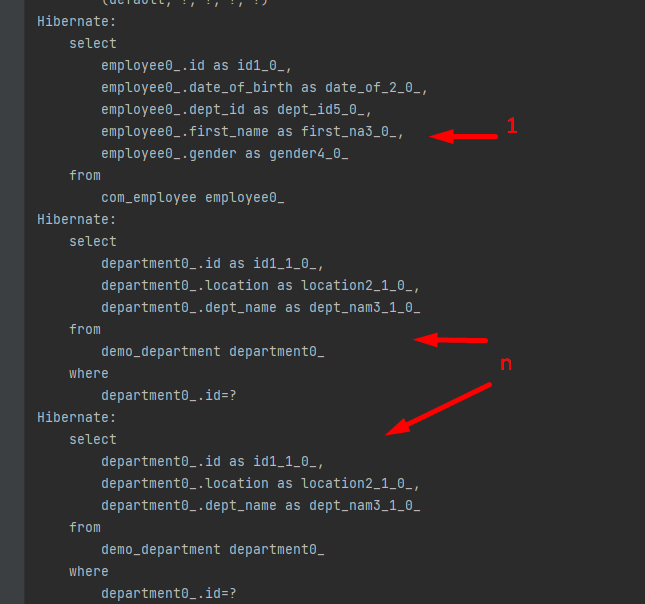
As following picture:



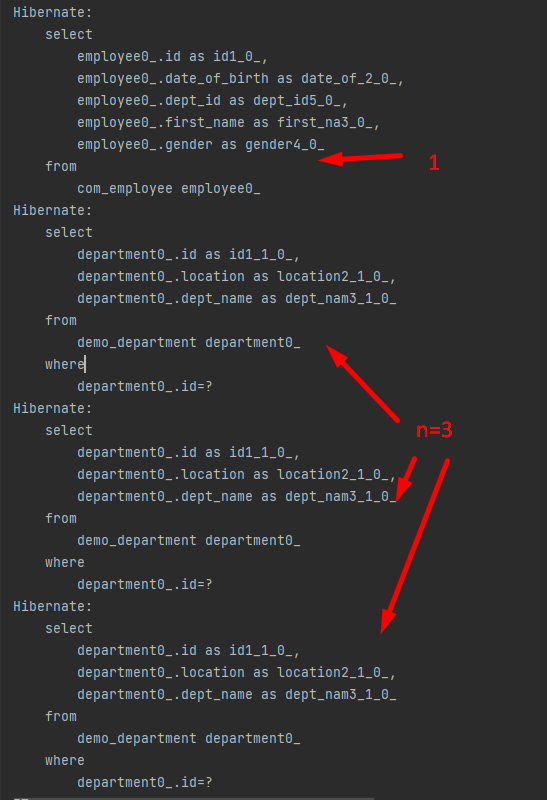
Remember to set log level to Info in properties file in Test Resource;



Run the test and SQL show like this: n = 2 (because we set 3 employees in **2 department** (IT and Marketing)



If I set Employee2 belong to Department “Admin”, n will be **3 as follows:**



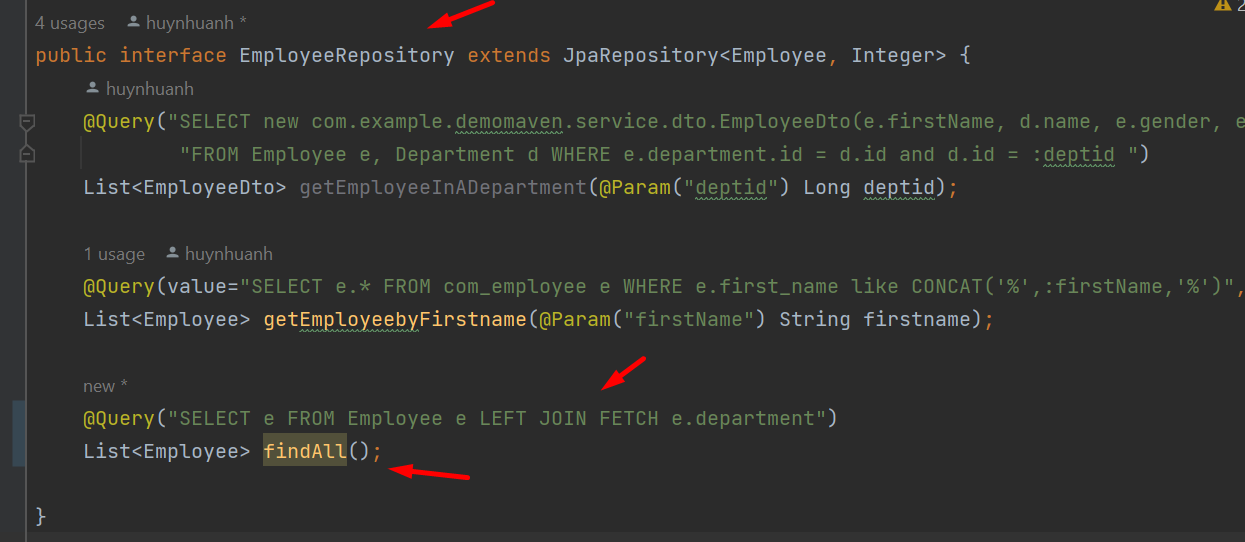
**How to solve this Problem?**

<https://thorben-janssen.com/5-ways-to-initialize-lazy-relations-and-when-to-use-them/>

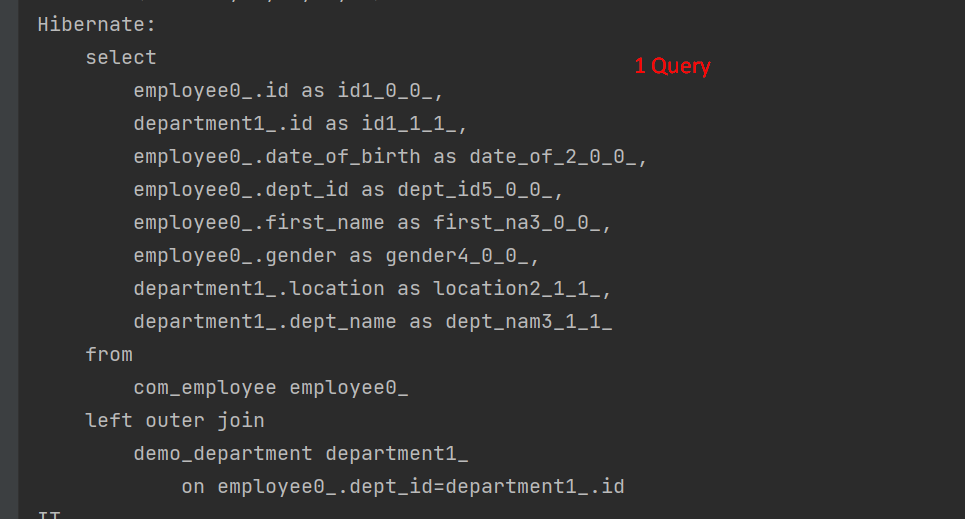
* Using **lazy load** not **eager fetch mode**; but we have problem with getDepartment() 🡺 return null
* Fetch join in JPQL or Criteria API
* **Named Entity Graph** or **Dynamic Entity Graph** (> JPA 2.1 New features)

**Part 1 -1: demo using Fetch join in JPQL:**

Step 1: rewrite the findAll function in EmployeeRepositorie:

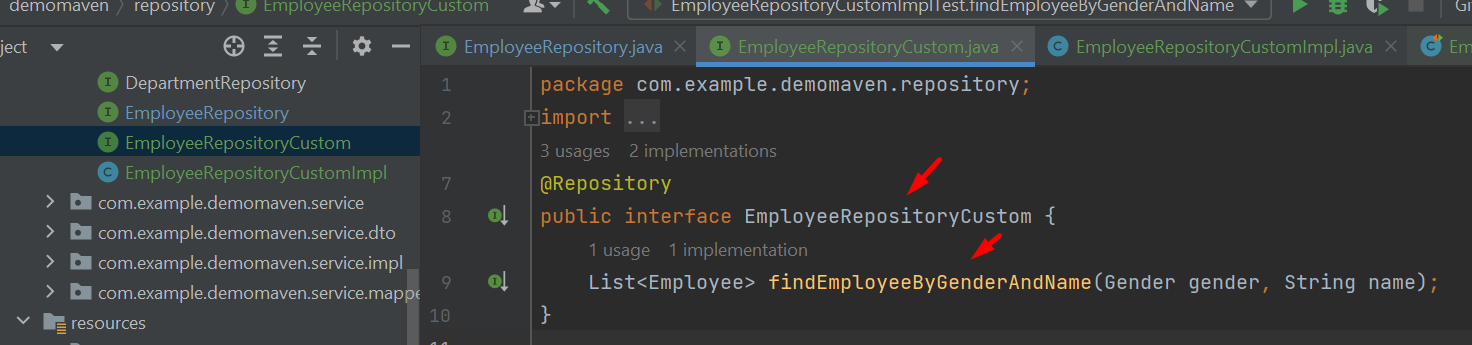


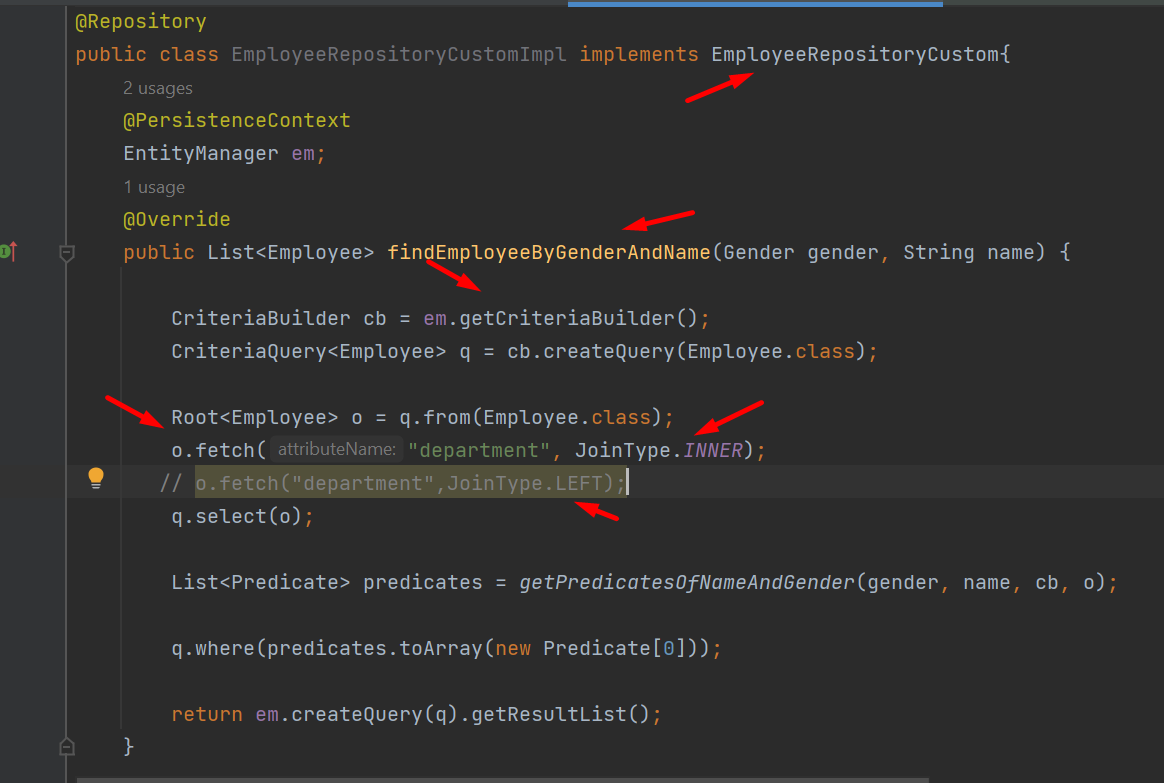
Step 2: Run the unit test again to see the generated sql:



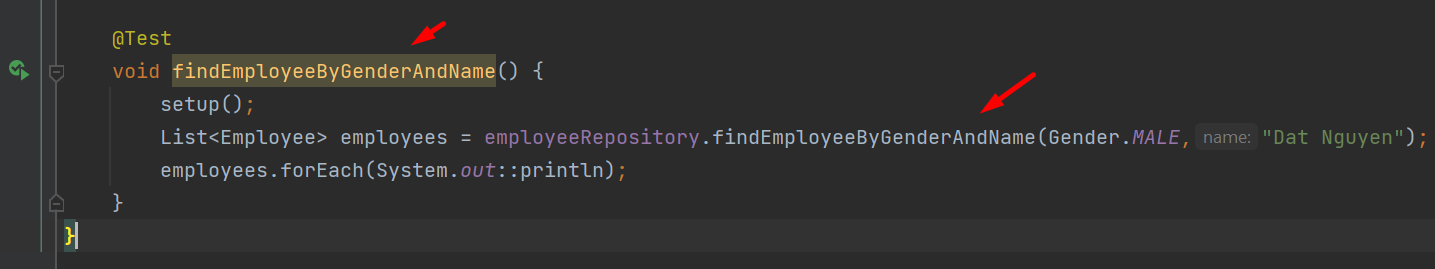
**Part 1 -2: demo using Criteria**

Step 1: Create an custom interface for using Entity Manager & a Class implement it

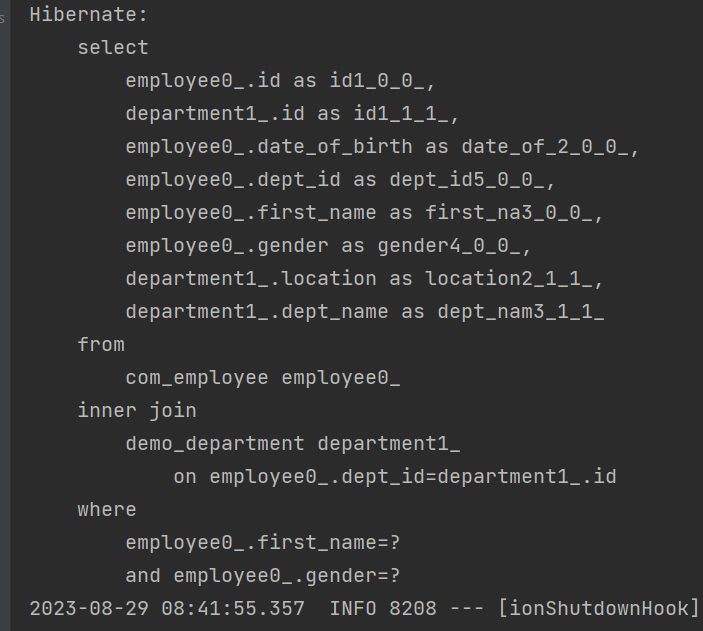




Step 2: write a test case to call that function:



Step 3: Run the unit test again to see the generated sql:

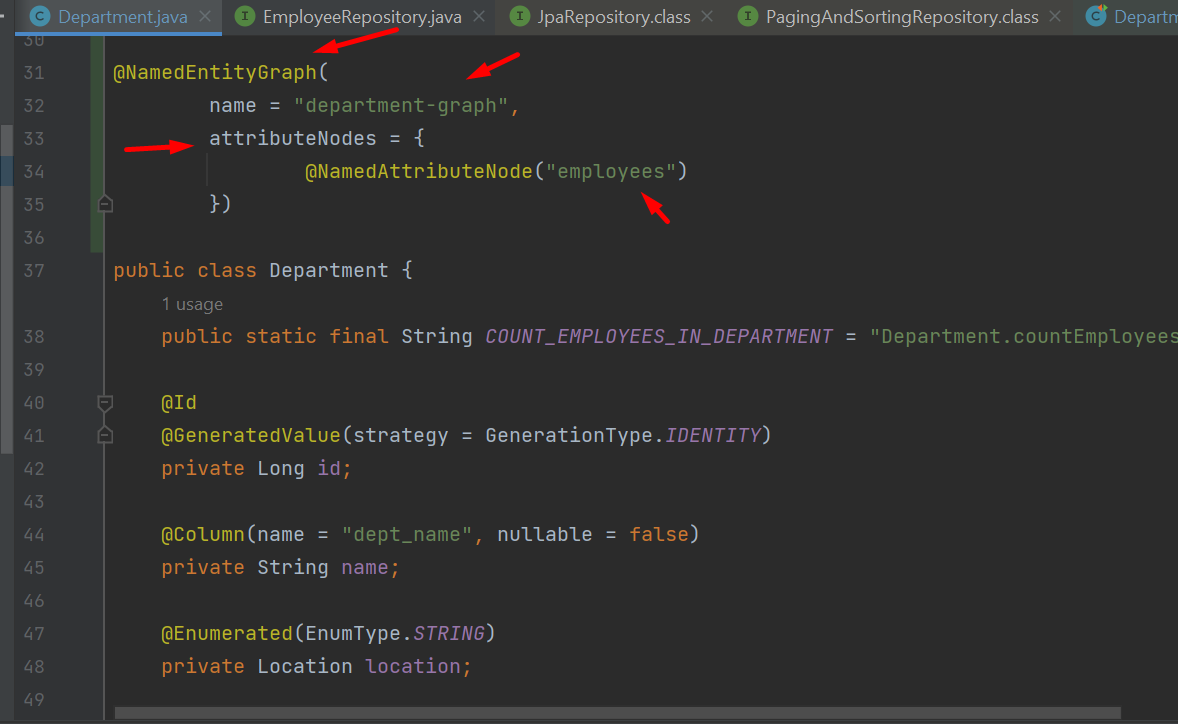


**Part 2-1: demo using “Named Entity Graph”**

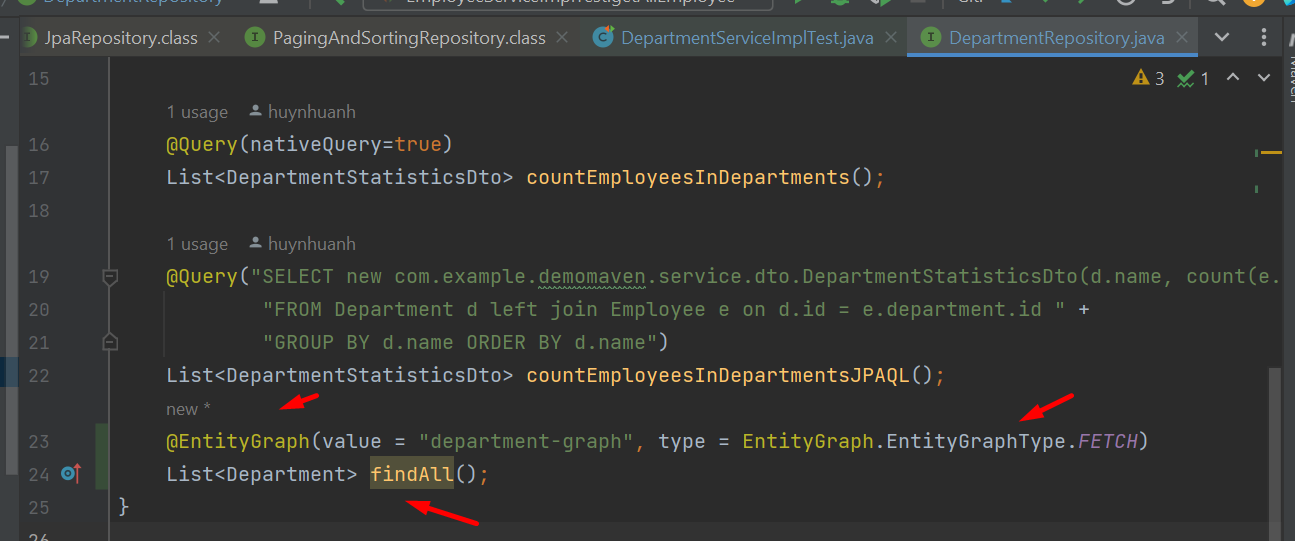


In Department entity, we add bidirectional mapping with @OneToMany to employees.

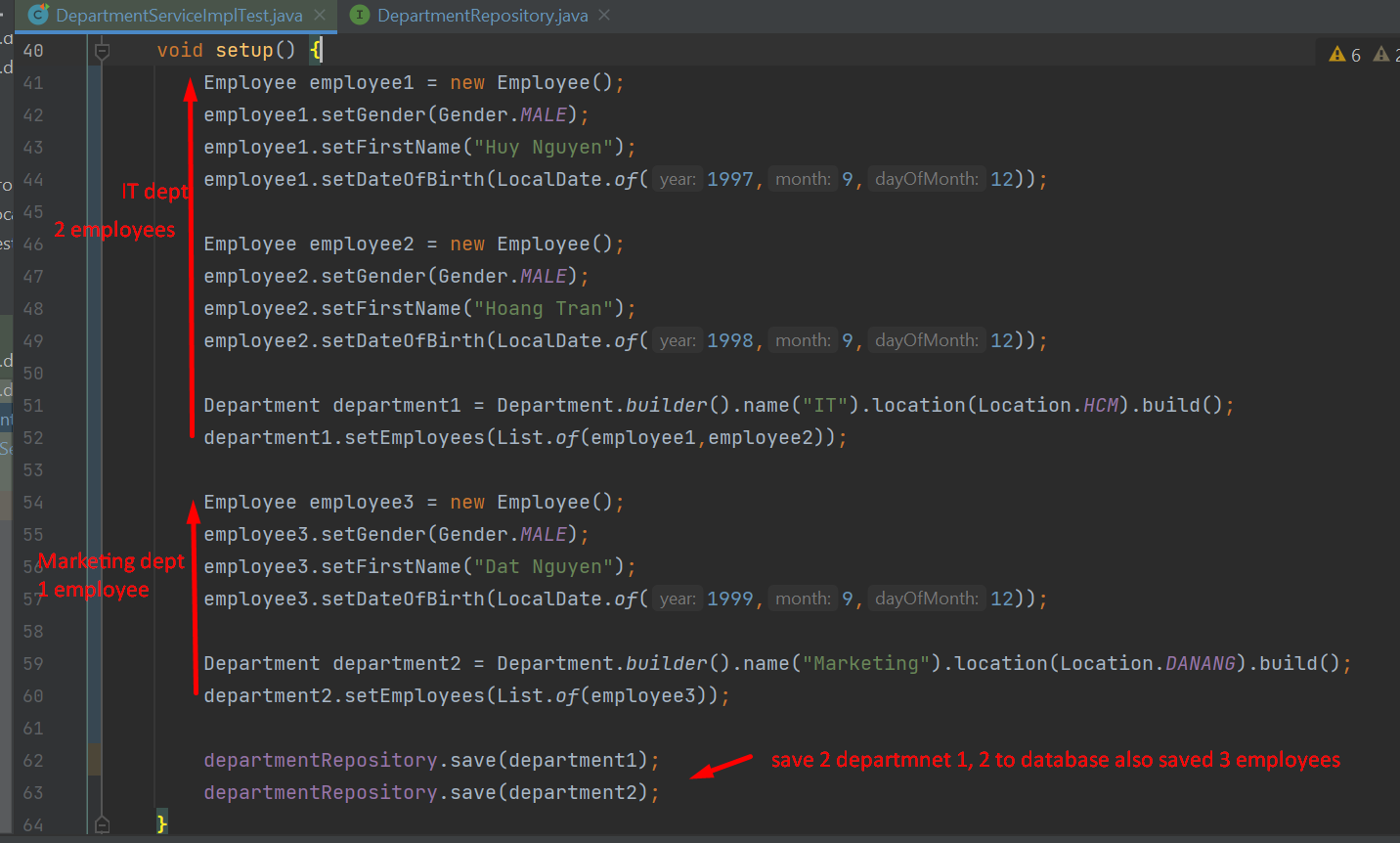
Then we declare Named Entity Graph and its attribute Nodes as follow:

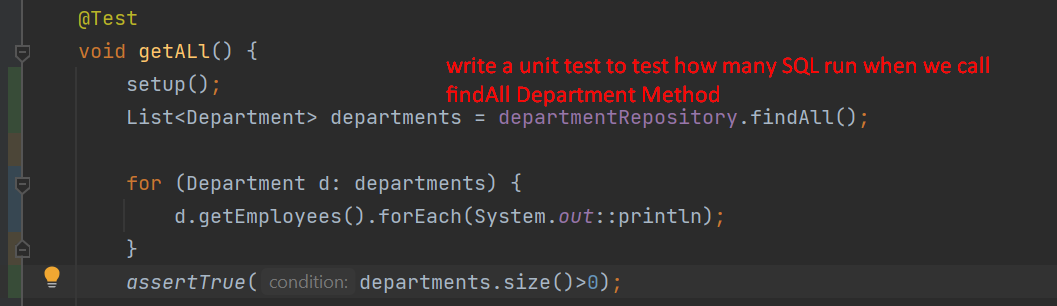


In the Department Repositories, we override findAll method as follows:

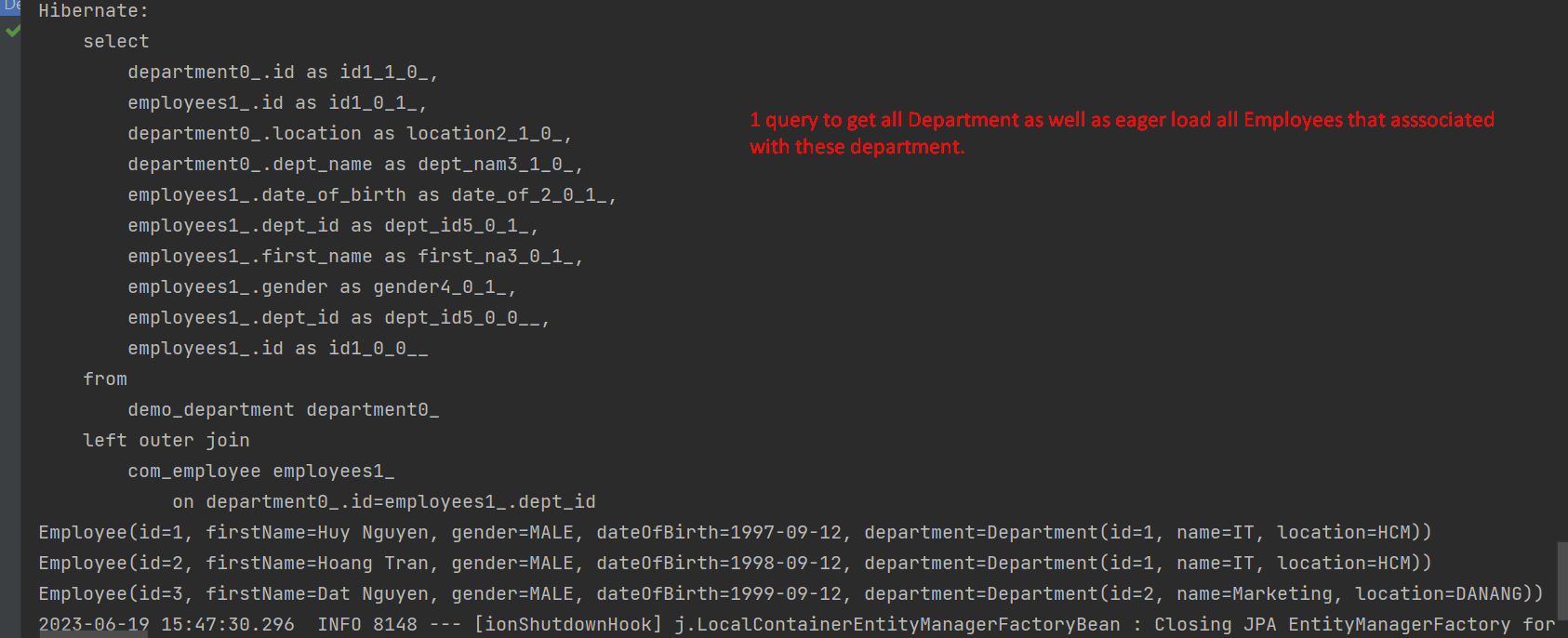


Setup method will create 2 departments as well as employees in them.



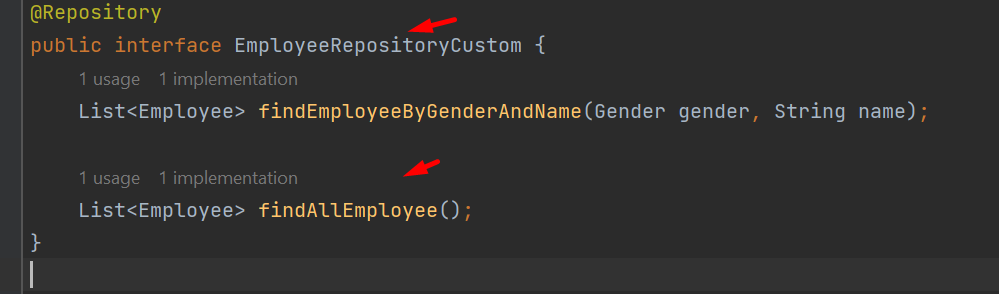


And console log will show that only 1 query run as following picture:



**Part 2-2: demo using “Dynamic Entity Graph”**

Step 1: create 1 more function getAllEmployee in custom interface



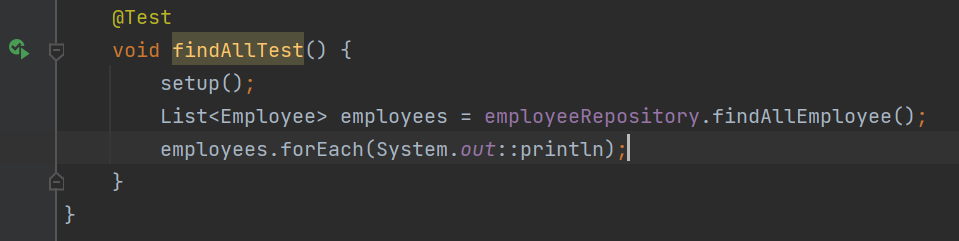
Step 2: add dynamic entity graph in the implementation class



(loadgraph & fetchgraph)

<https://docs.oracle.com/javaee/7/tutorial/persistence-entitygraphs001.htm>

Step 3: Run the unit test again to see the generated sql:



==================Thank You ==================