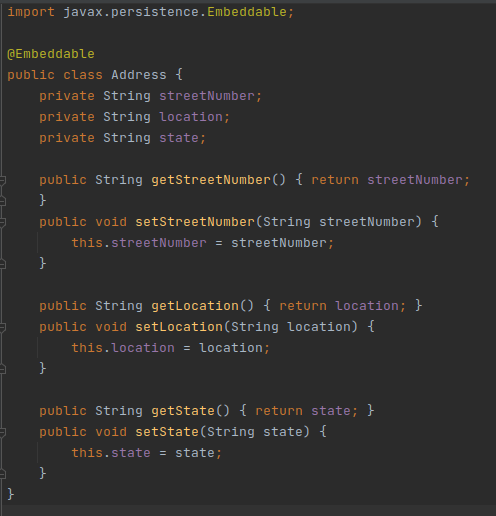
**Spring data JPA using Hibernate-3**

**Name:Akanksha Tyagi**

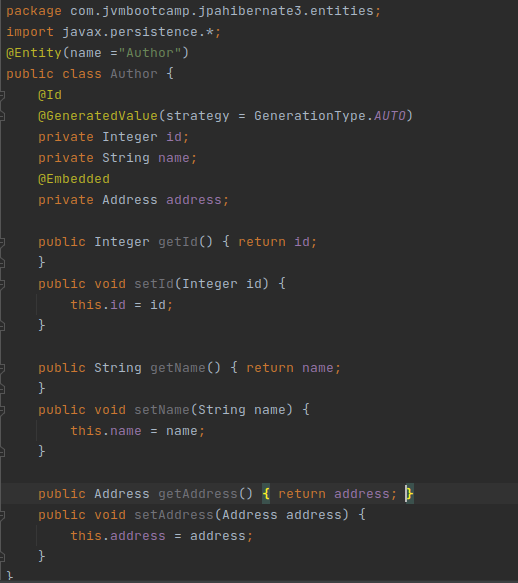
**id=4701**

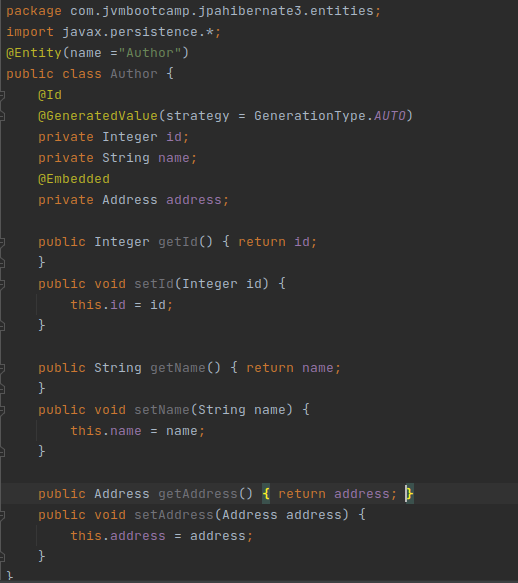
**1. Create a class Address for Author with instance variables streetNumber, location, State.**

Address.java



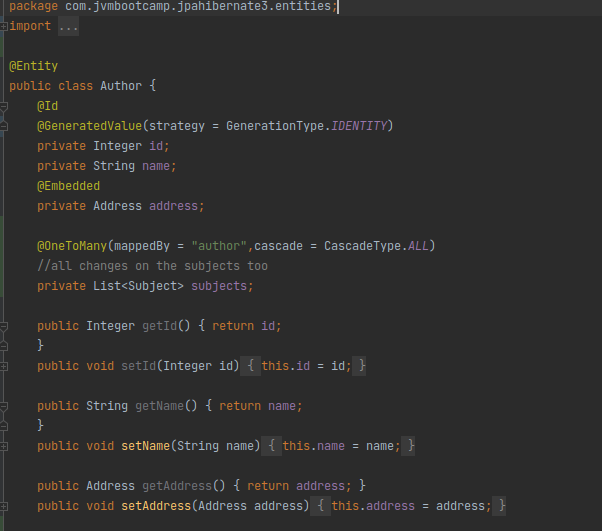
**2. Create an instance variable of the Address class inside the Author class and save it as an embedded object.**

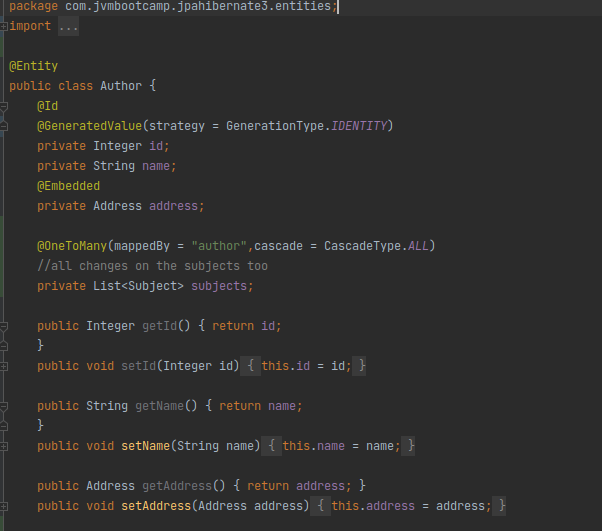
****

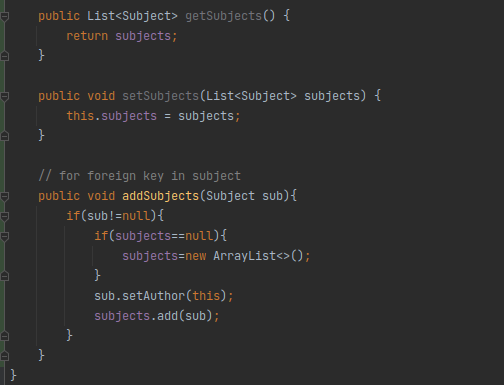
****

**3. Introduce a List of subjects for the author.**

Author.java





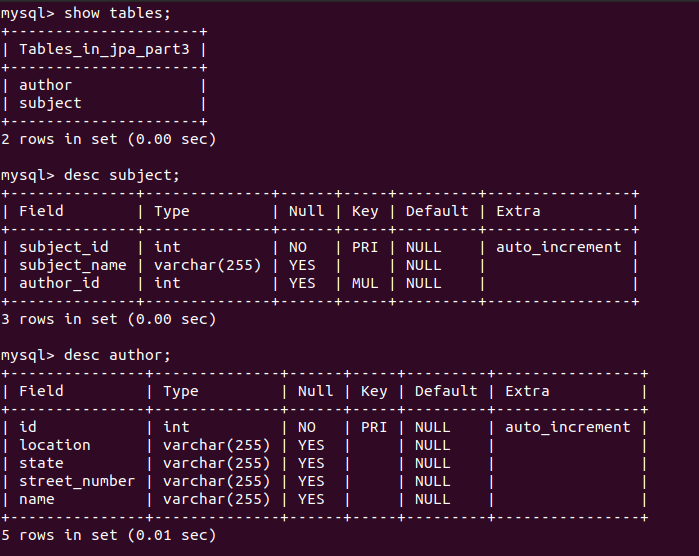


Subject.java





**Output :**

****

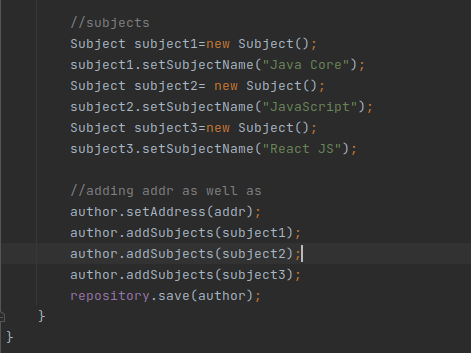
**4. Persist 3 subjects for each author.**

AuthorController.java

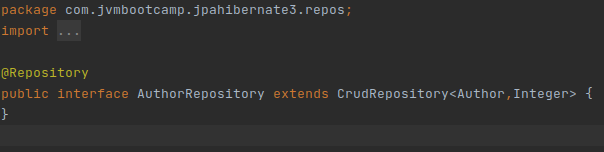


AuthorService.java

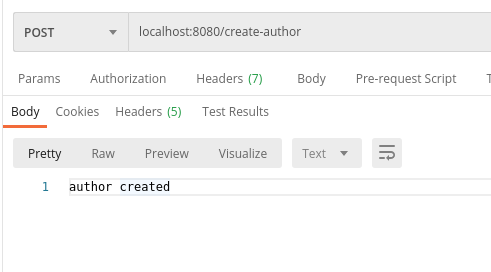


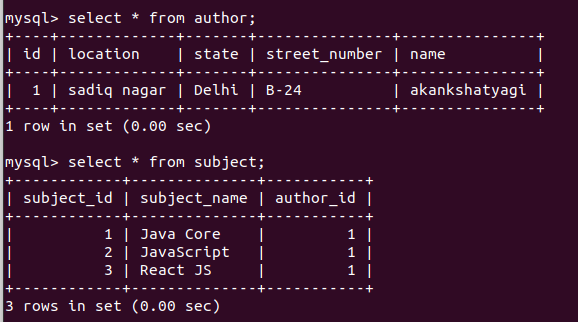


AuthorRepository.java



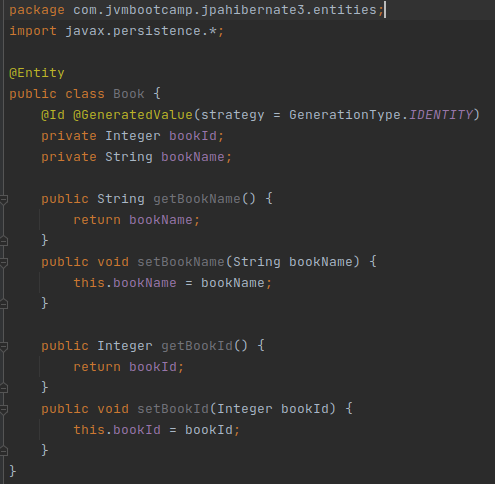
**Output:**

****

****

**5. Create an Entity book with an instance variable bookName.**

Book.java



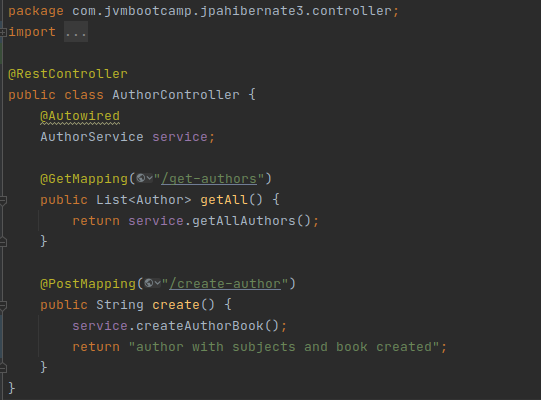
Author.java





**6. Implement One to One mapping between Author and Book.**

AuthorController.java:

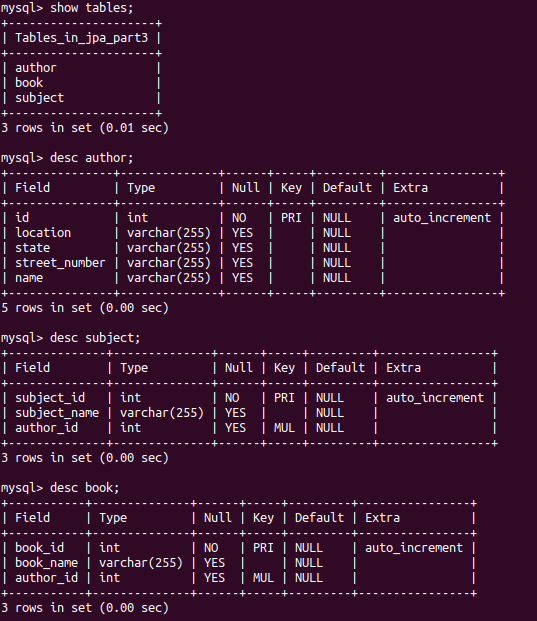


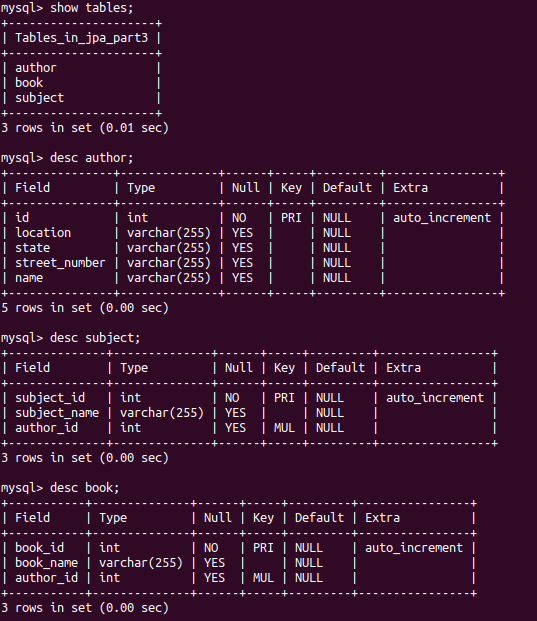
AuthorService.java:



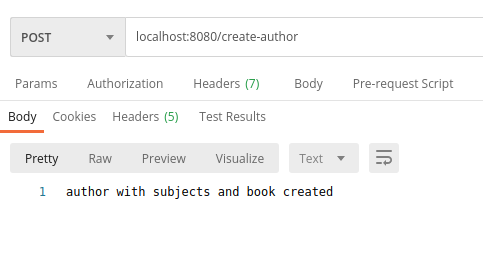


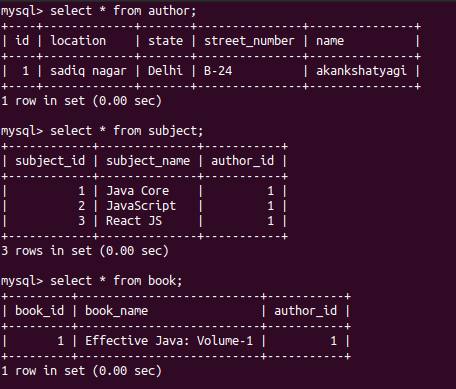
Output:

****

****

**Inserted data--**

****

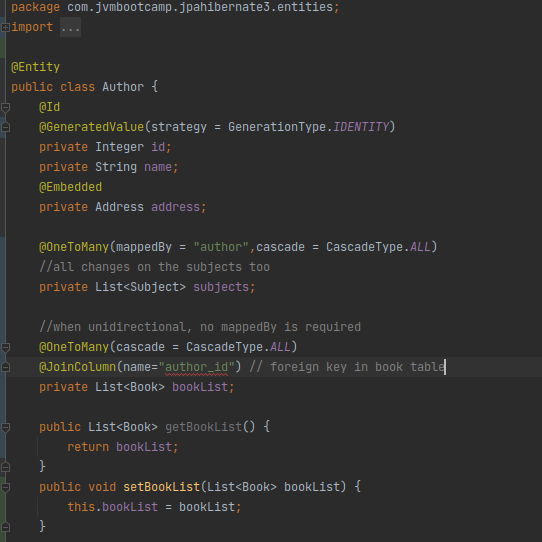
****

**7. Implement One to Many Mapping between Author and Book(Unidirectional, BiDirectional and without additional table ) and implement cascade save.**

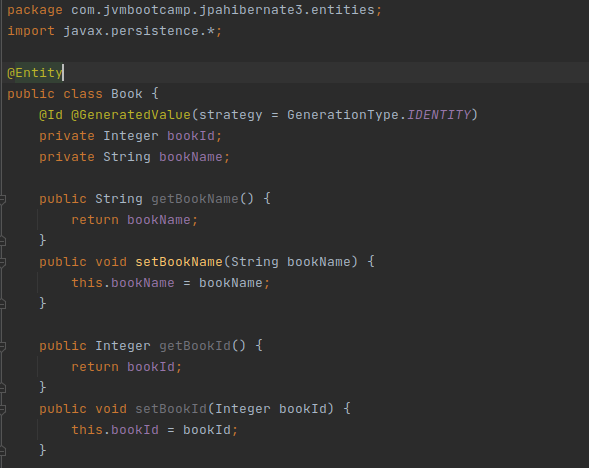
**Unidirectional:**

**CODE:**

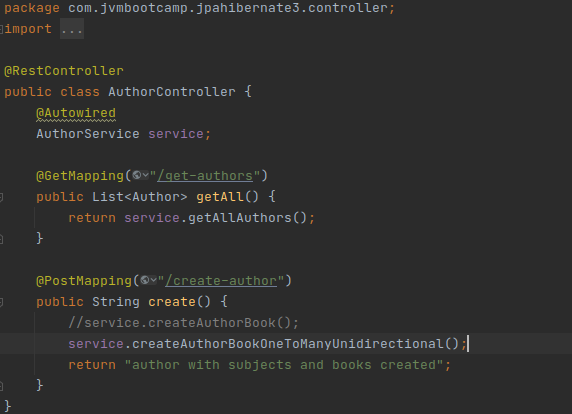
Author.java



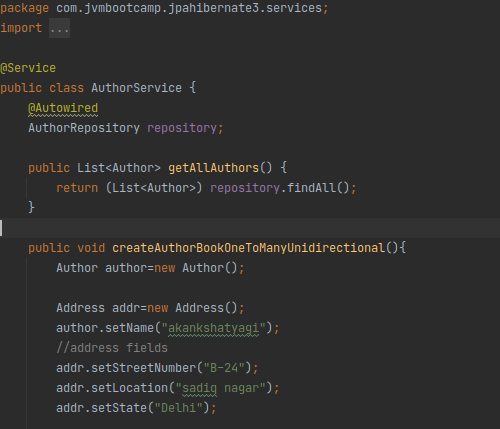
Book.java

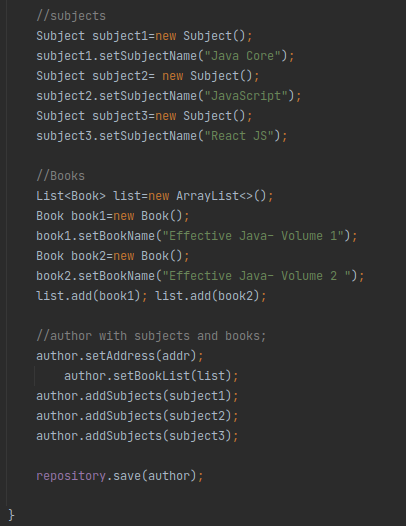


AuthorController.java



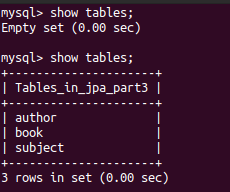
AuthorService.java:



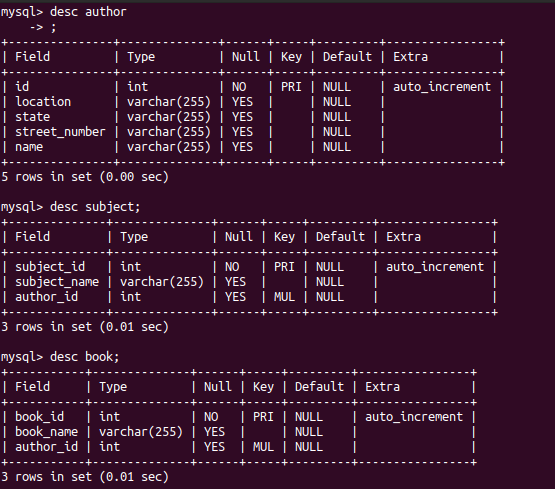


**Output:**

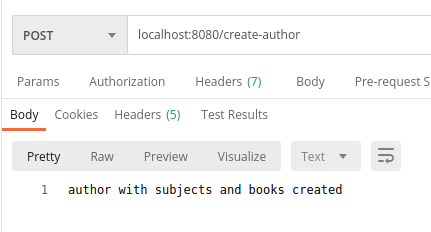
**Tables-**

****

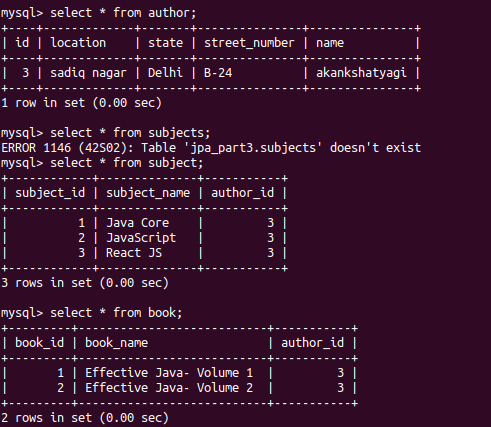
**Description:**

****

**Inserting data:**

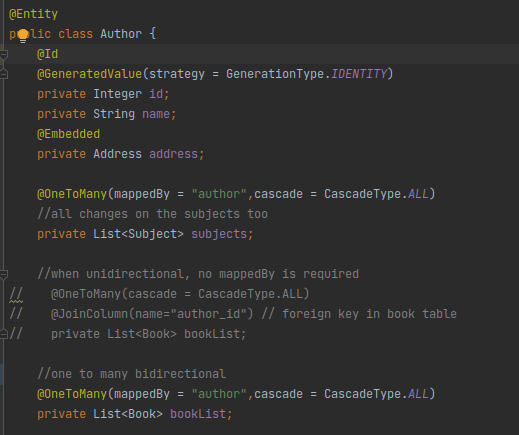
****

**Select \* from various tables:**

****

**Bidirectional:**

Author.java

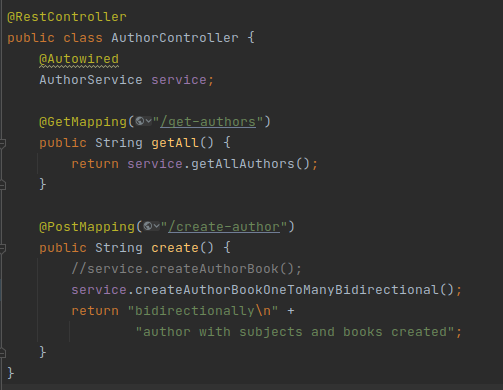




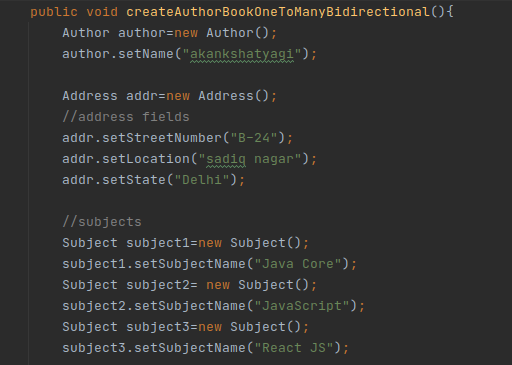
Book.java

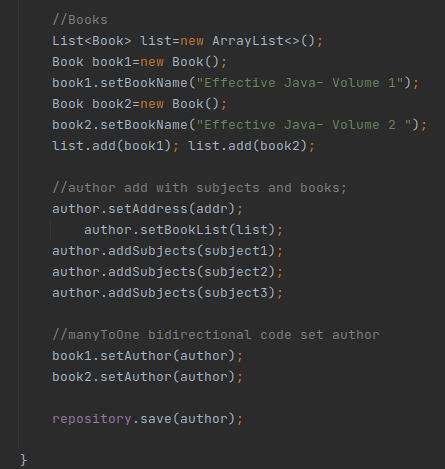


AuthorController.java



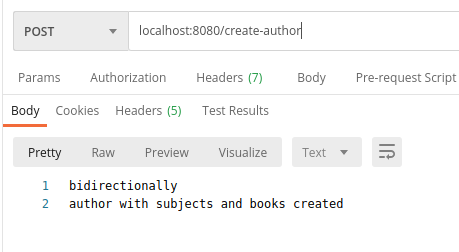
AuthorService.java



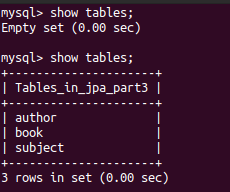


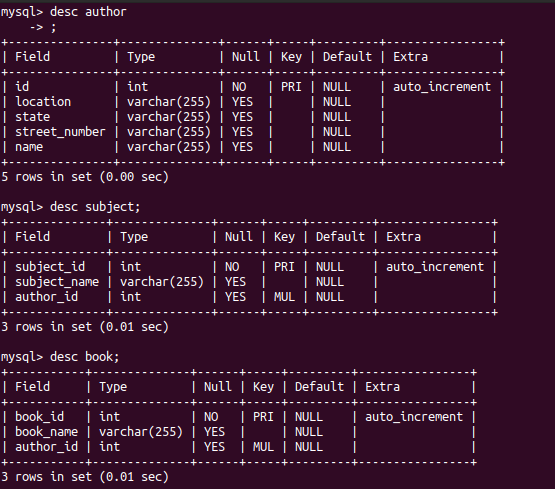
**Output:**

Data inserted:

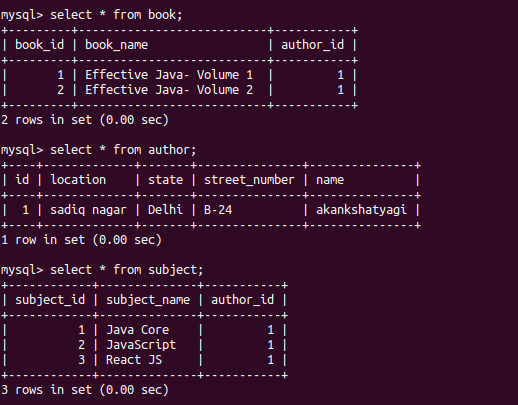


Tables:





Select \* from various tables.

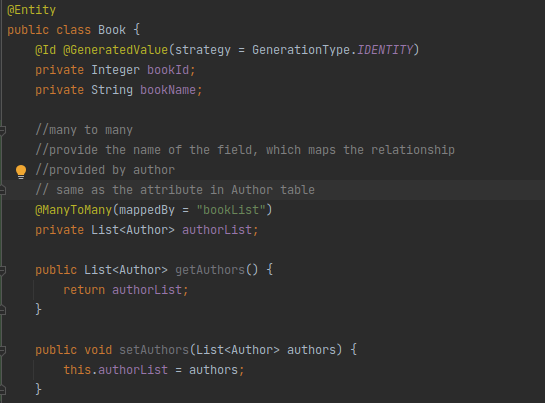


**8.Implement Many to Many Mapping between Author and Book.**

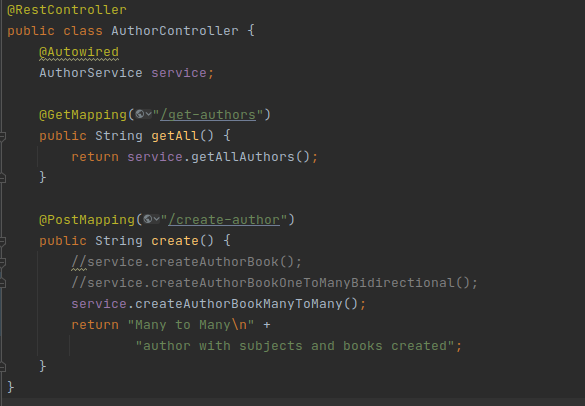
Author.java



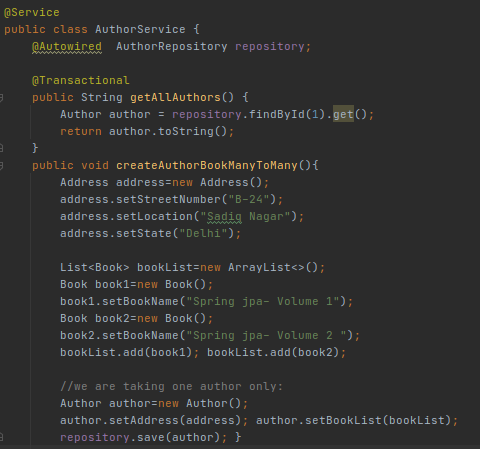
Book.java

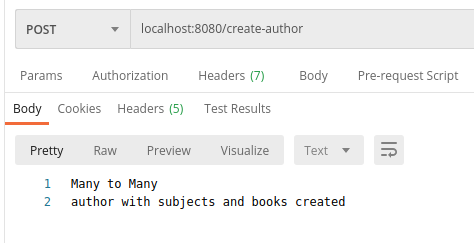


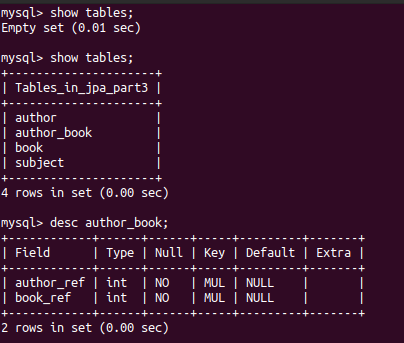
AuthorController.java

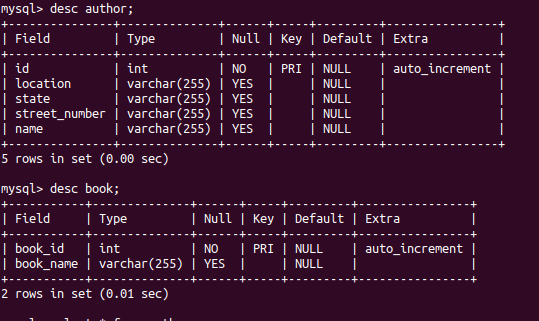


AuthorService.java

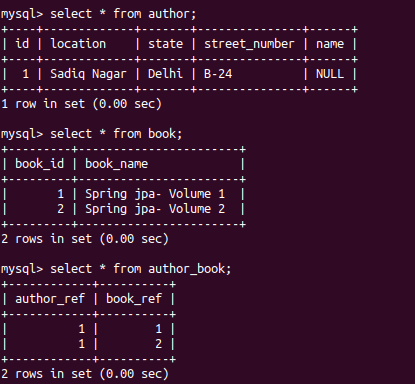


**Output: **

****

****

**Select statements:**

****

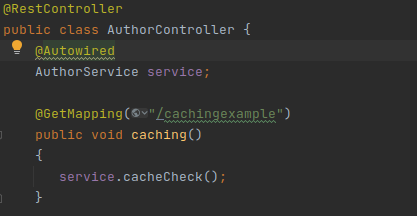
**9. Which method on the session object can be used to remove an object from the cache?**

**First Level Cache:**

**AuthorService.java**

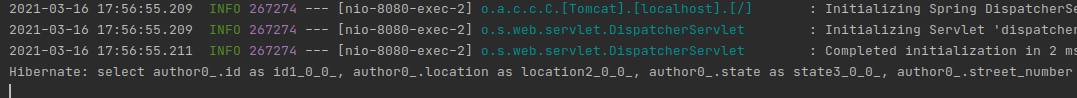
****

**AuthorController.java**

****

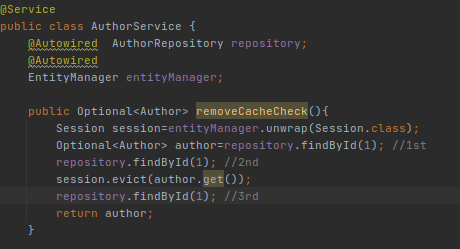
**Output:**

findById(1) is executed only once.

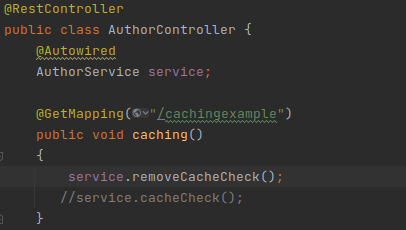
****

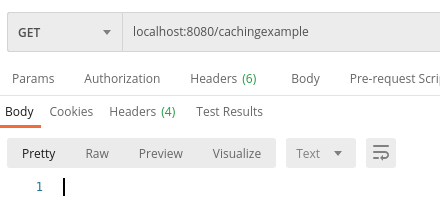
**→ evict()- to remove an object from the cache.**

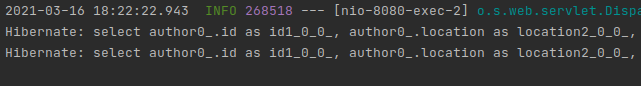
AuthorService.java:



AuthorController.java

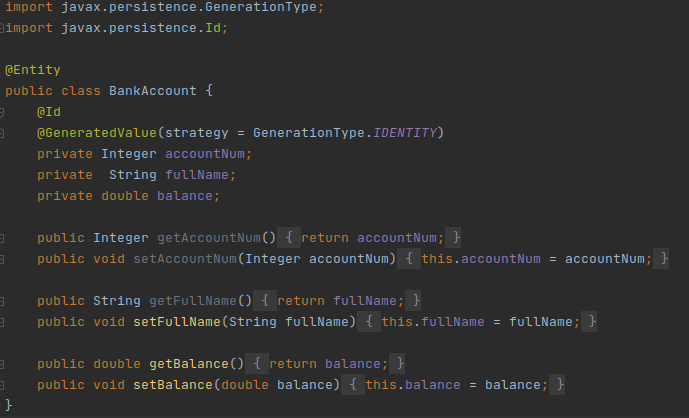


****

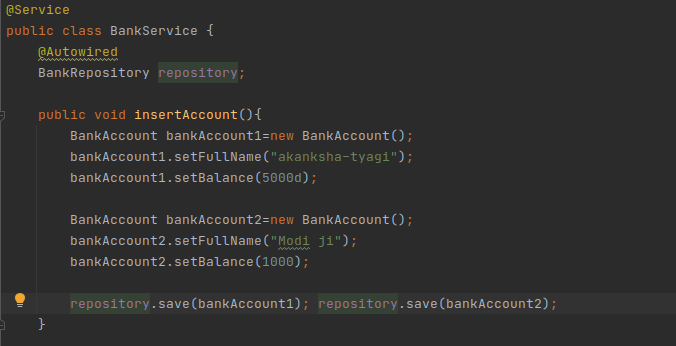
****

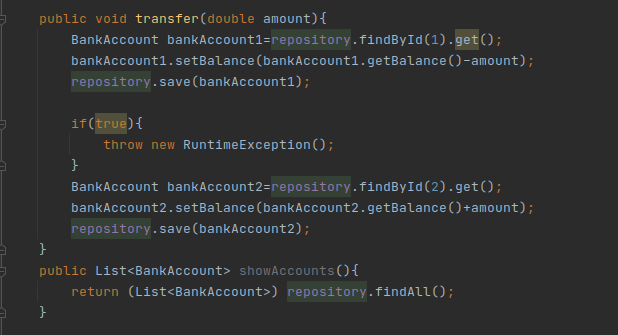
**10. What does @transactional annotation do?**

**BankAccount.java**

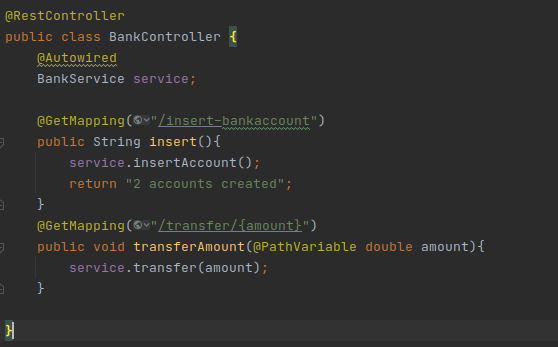
****

**BankService.java**

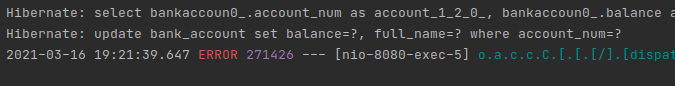
****

****

**BankController.java**

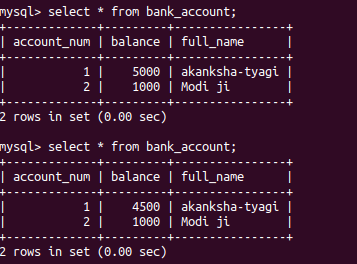


There is only one update statement on the console but there should be 2.



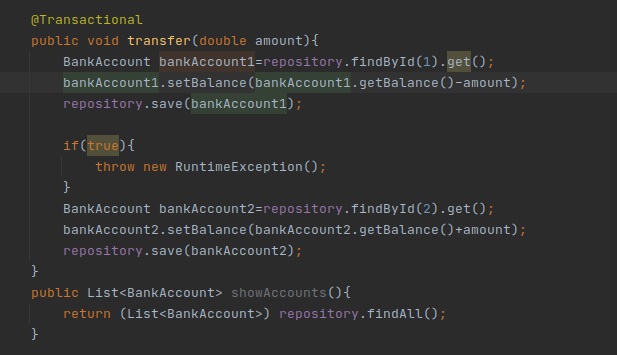
**Without Transactional**

**THE AMOUNT IS DEDUCTED FROM ONE ACCOUNT BUT NOT UPDATED IN ANOTHER ACCOUNT AS EXCEPTION IS THROWN IN BETWEEN.**

****

**With Transactional**

BankService.java



BankController.java



There is no update statement(atomicity concept)



**THE AMOUNT IS NOT DEDUCTED FROM ONE ACCOUNT AS EXCEPTION IS THROWN IN BETWEEN.**

