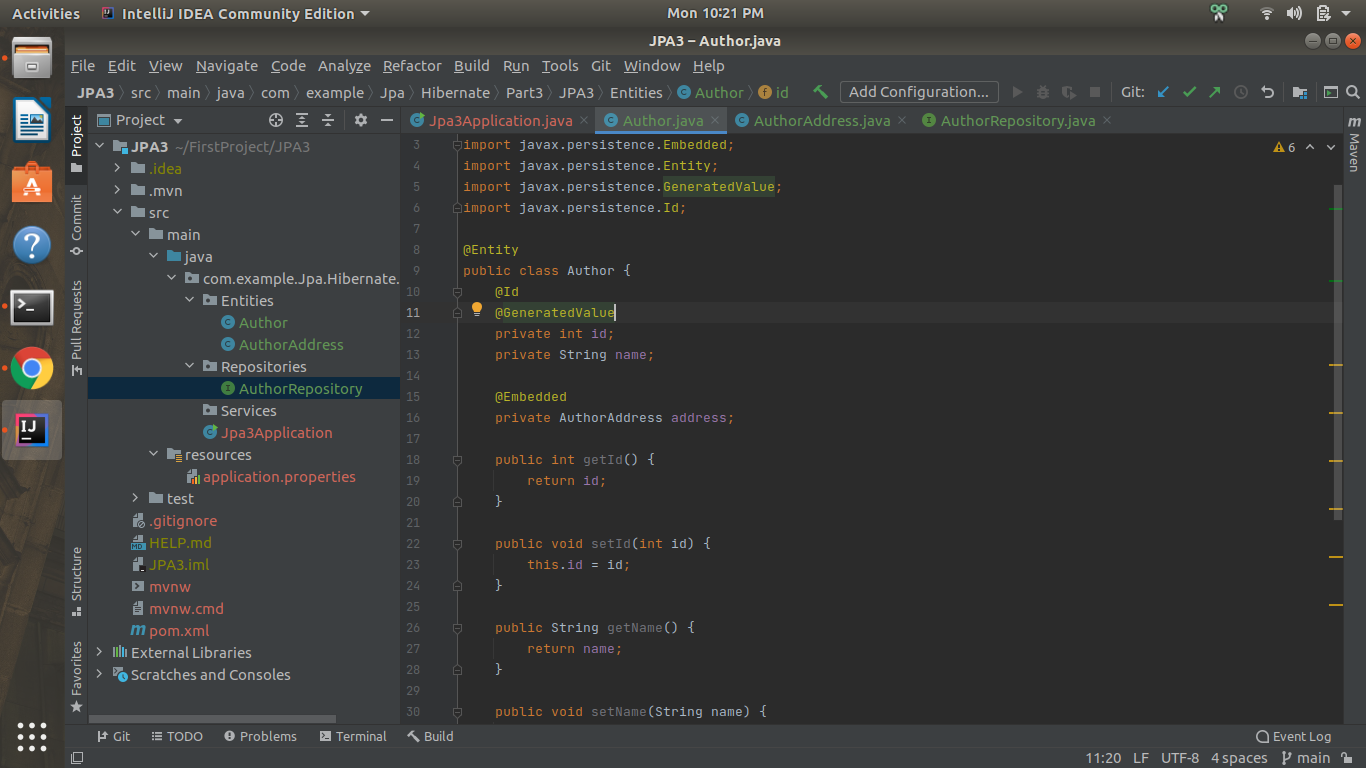
**ASSIGNMENT**

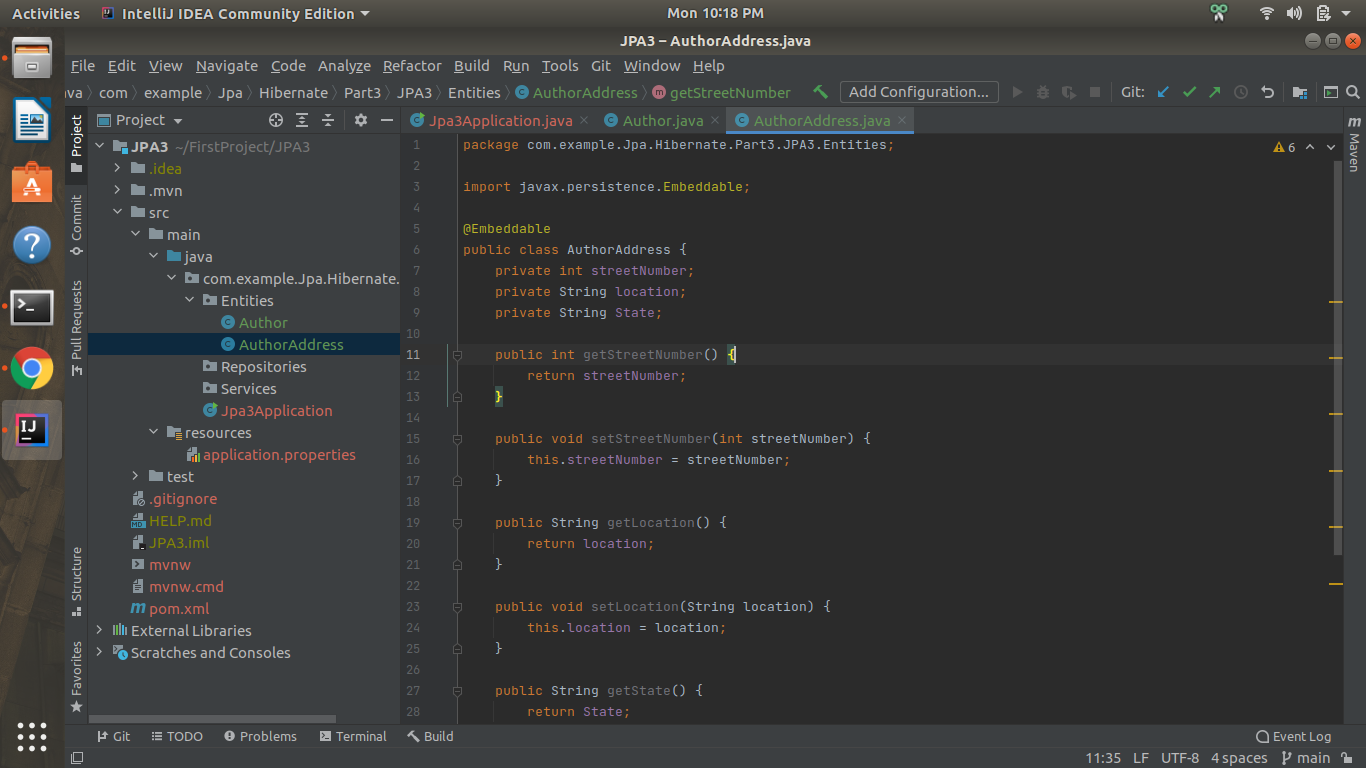
**JPA HIBERNATE PART-3**

**Doc link->** [**https://docs.google.com/document/d/1nUcwLOJvVfDEUpSHK4eCNe71WRdZviTTQ4Fvxqa7wjo/edit**](https://docs.google.com/document/d/1nUcwLOJvVfDEUpSHK4eCNe71WRdZviTTQ4Fvxqa7wjo/edit)

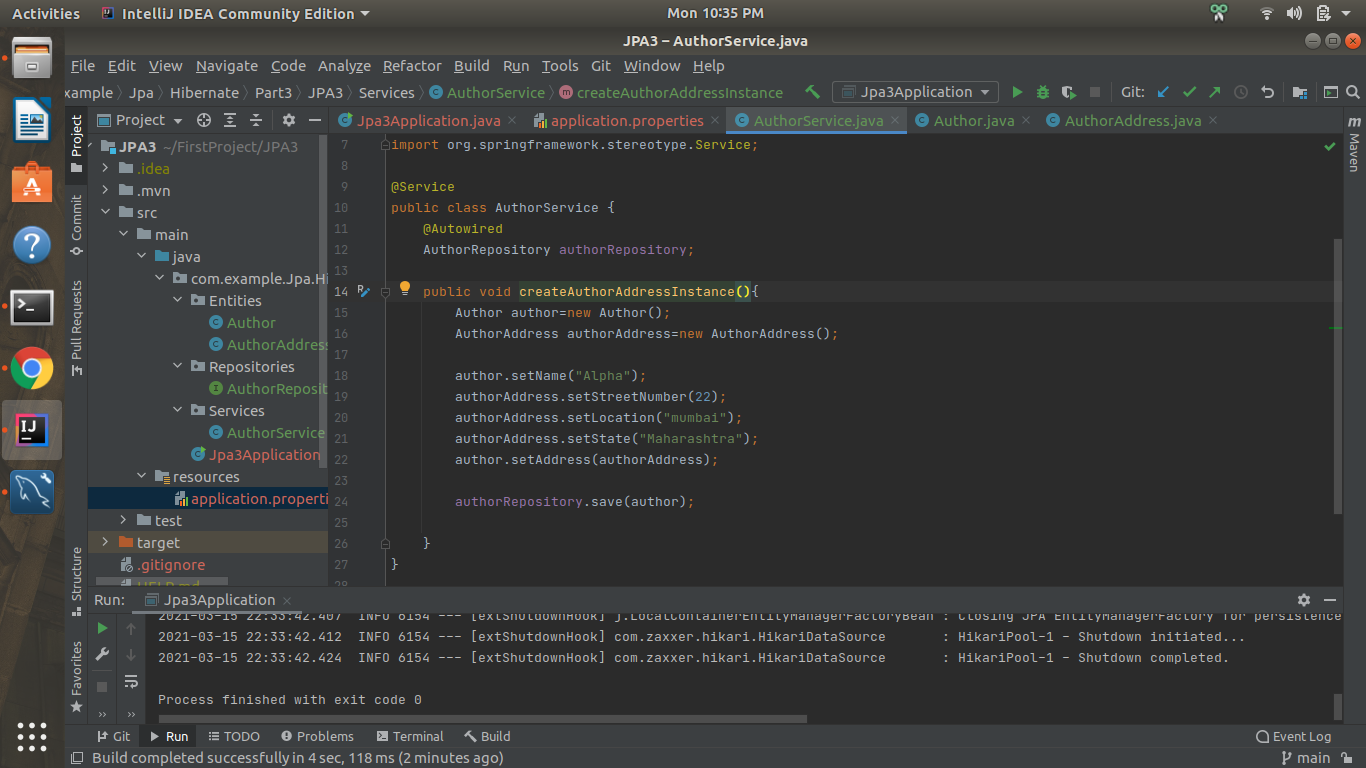
**Date-15/03/21**

Q1)Create a class Address for Author with instance variables streetNumber, location, State.

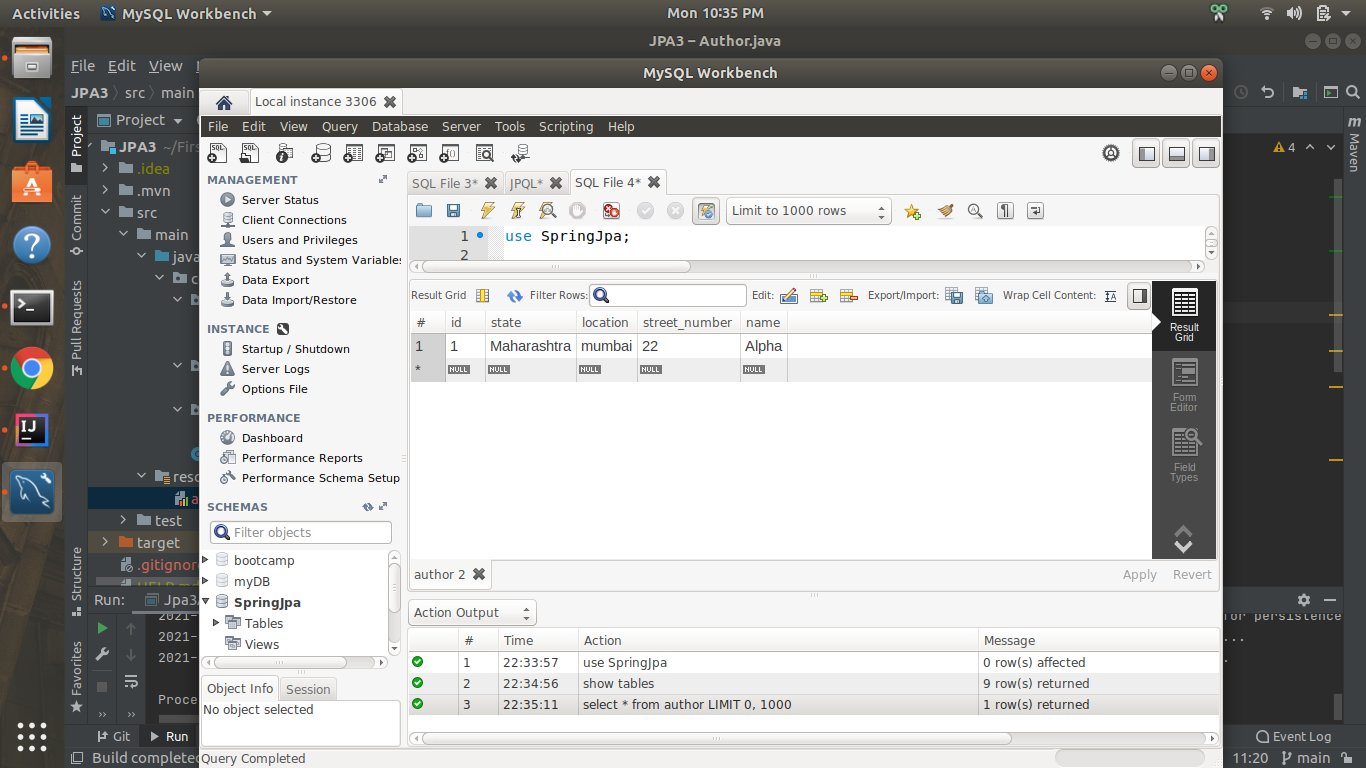




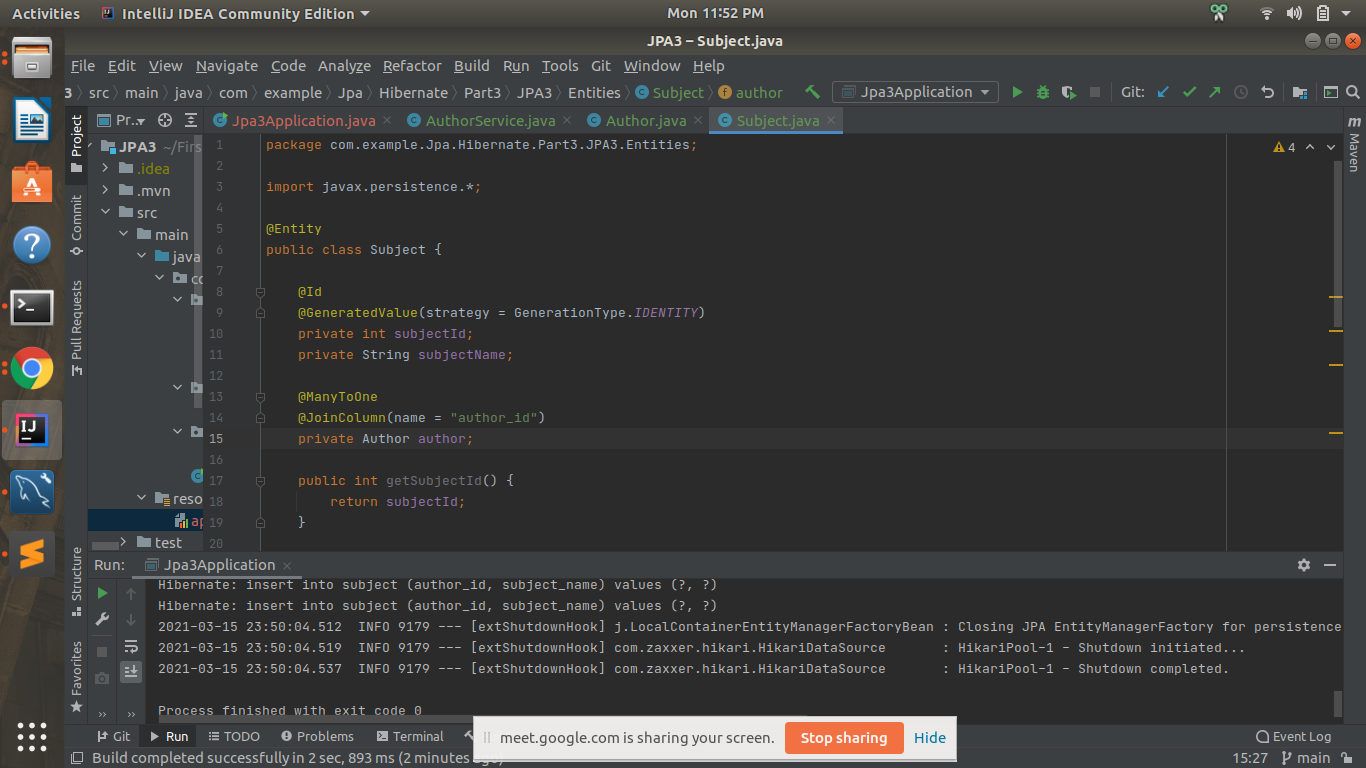
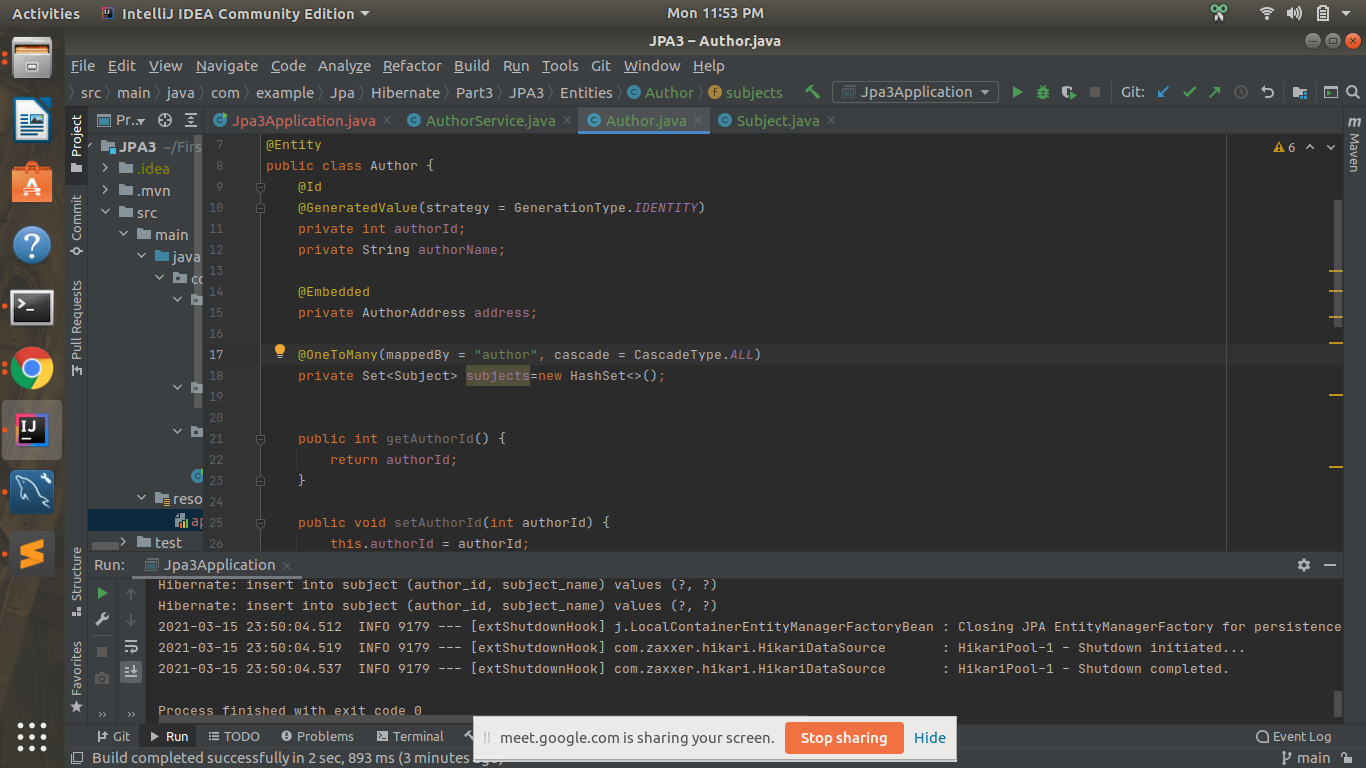
Q2)Create instance variable of Address class inside Author class and save it as embedded object.

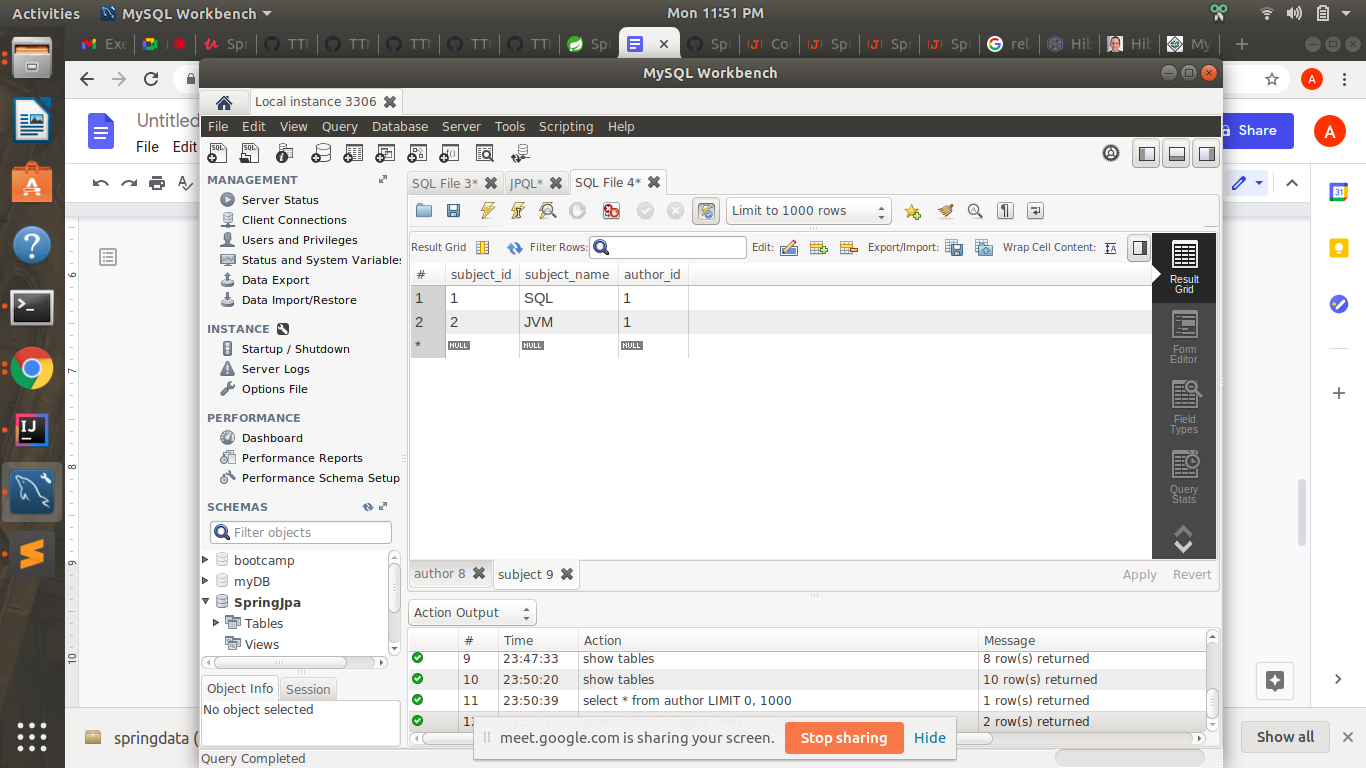


**author table**

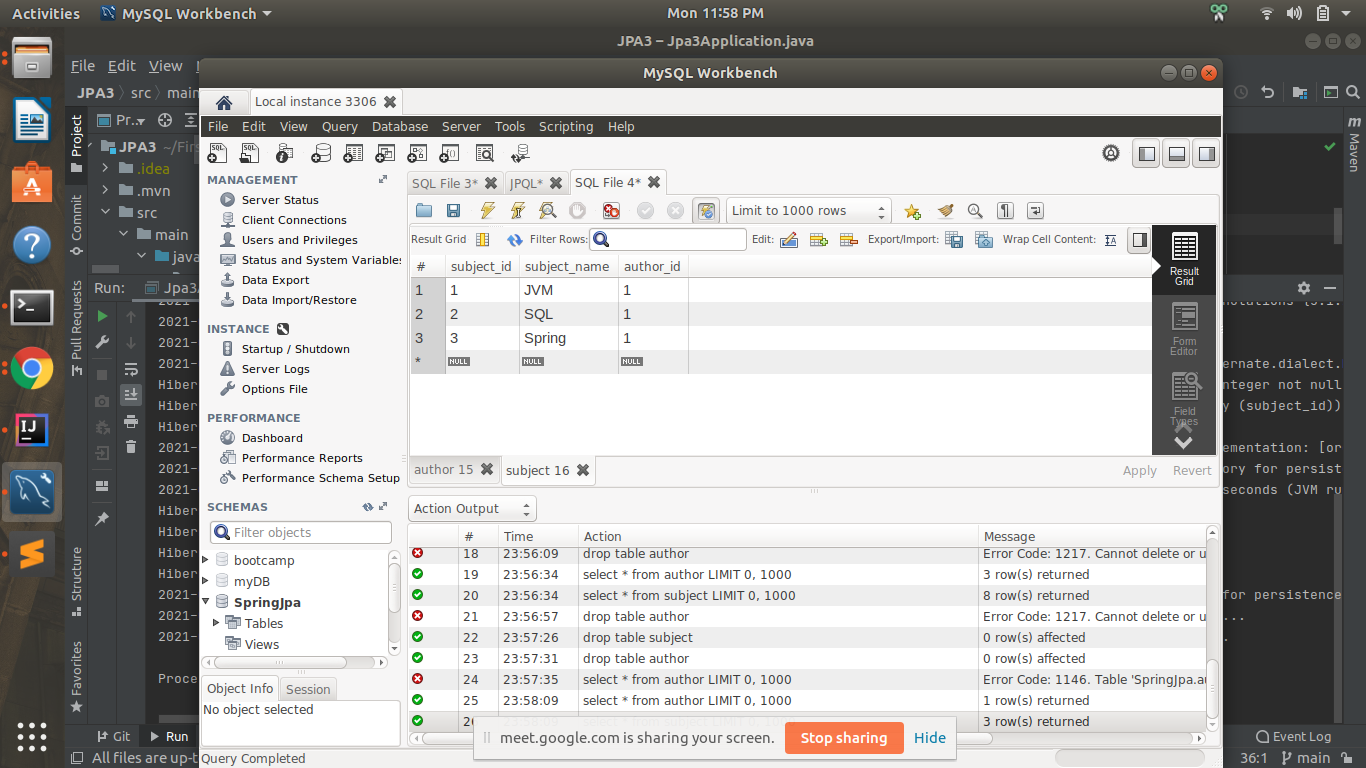


Q3)Introduce a List of subjects for author.

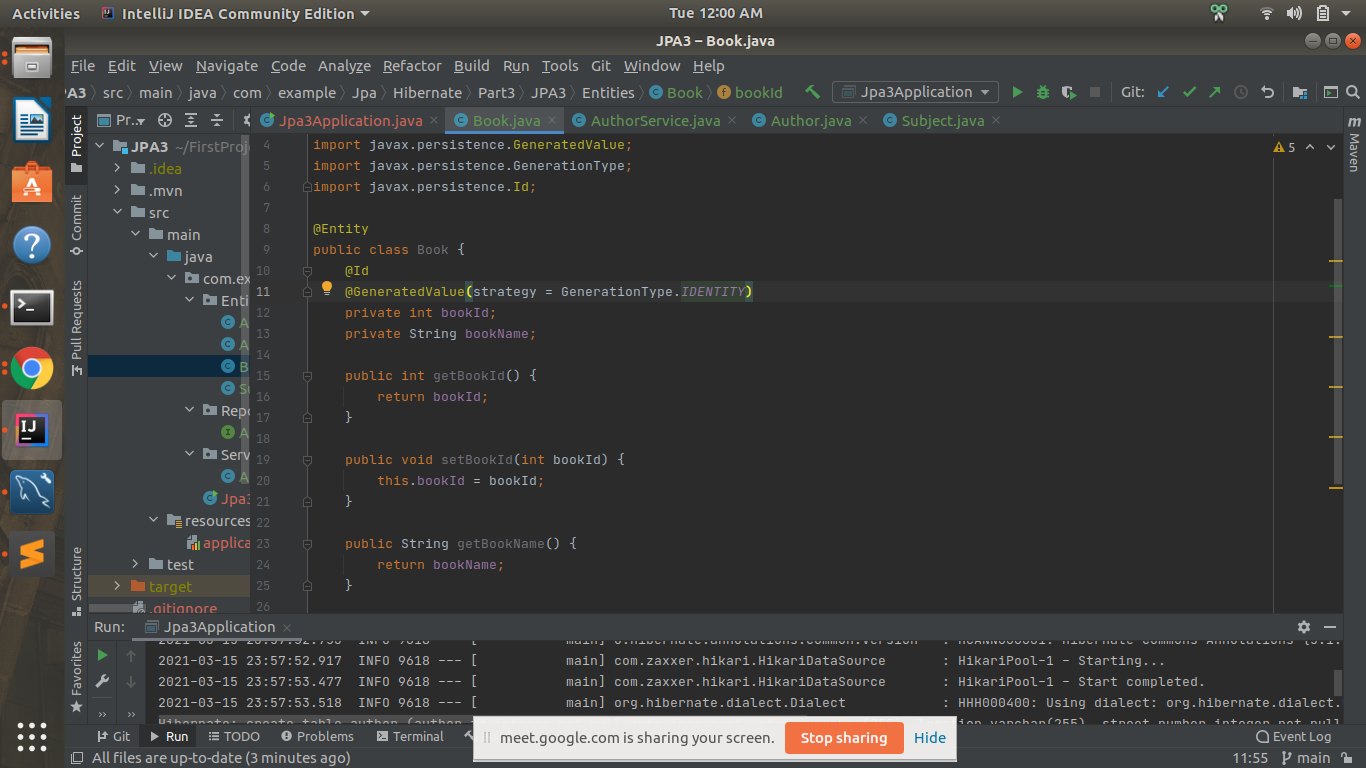




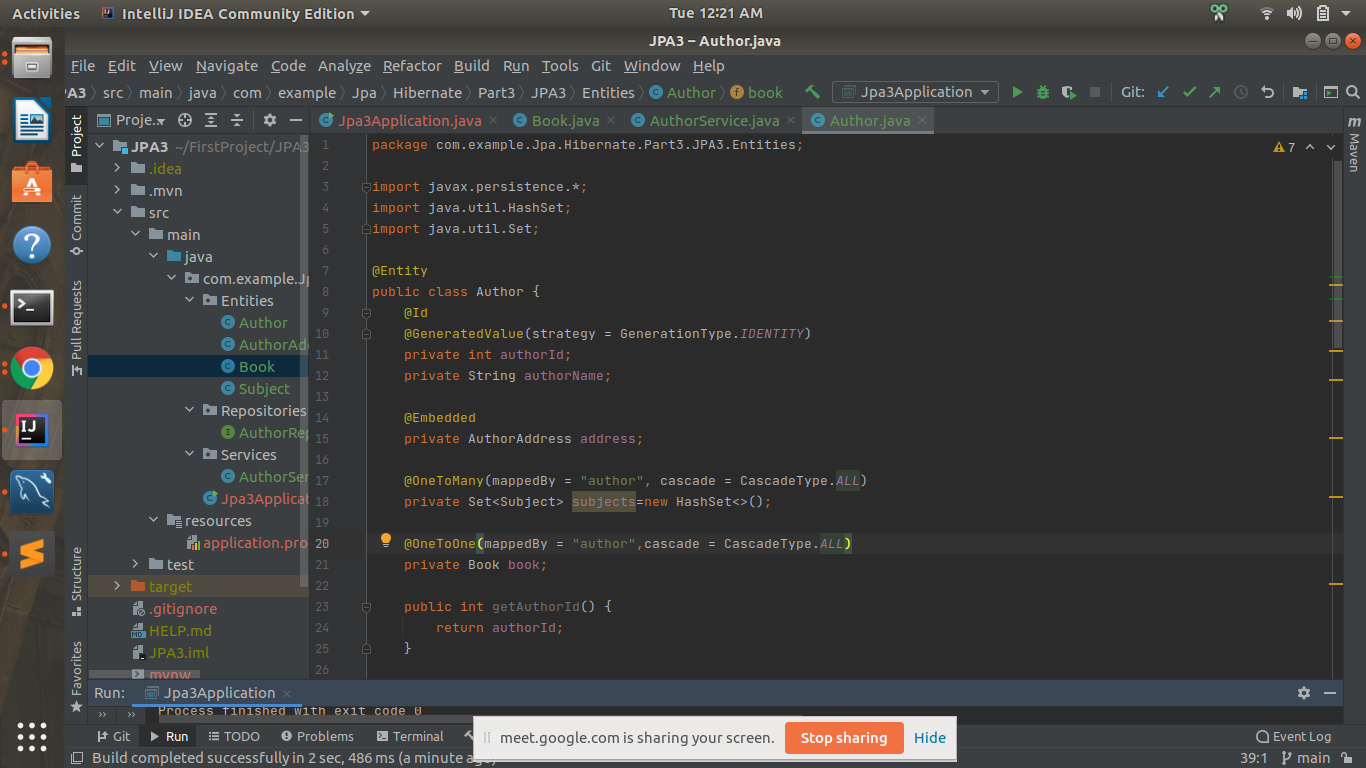
Q4)Persist 3 subjects for each author.

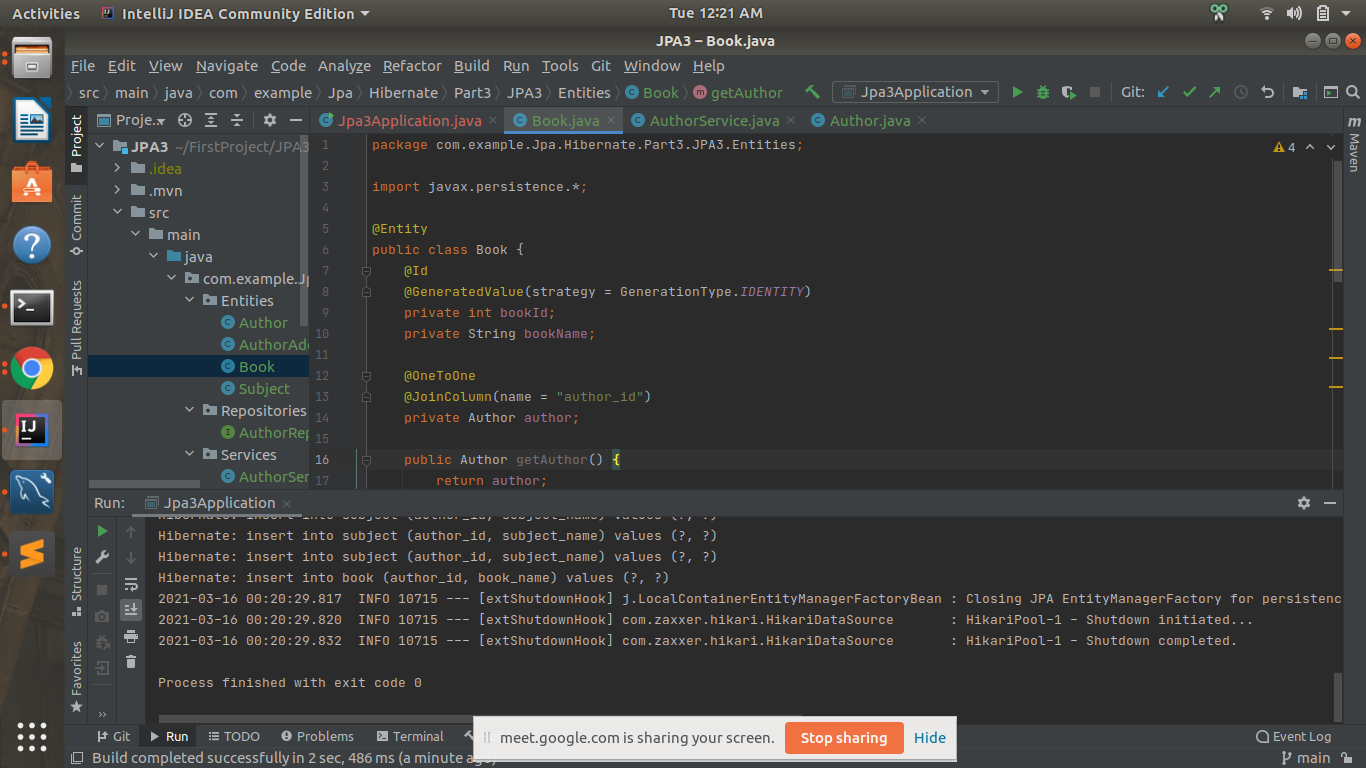


Q5)Create an Entity book with an instance variable bookName.

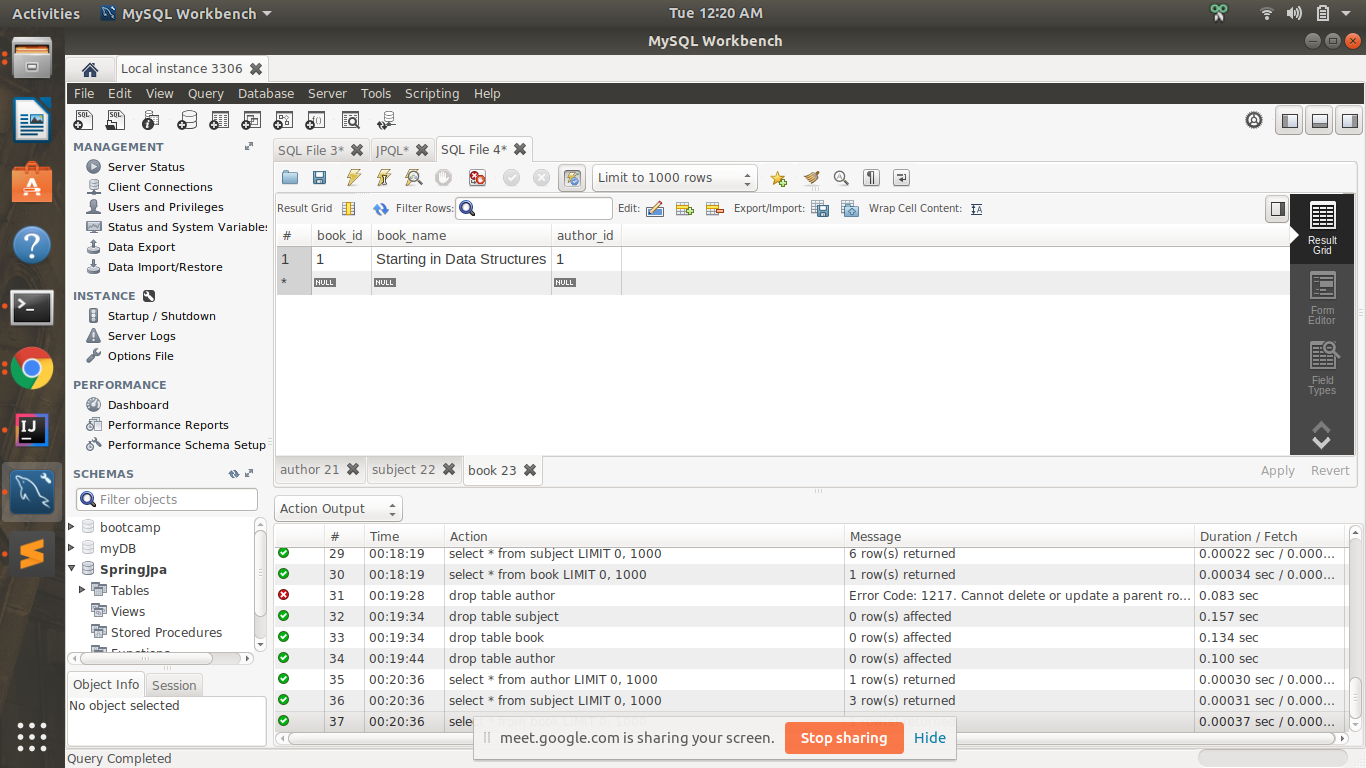


Q6)Implement One to One mapping between Author and Book.



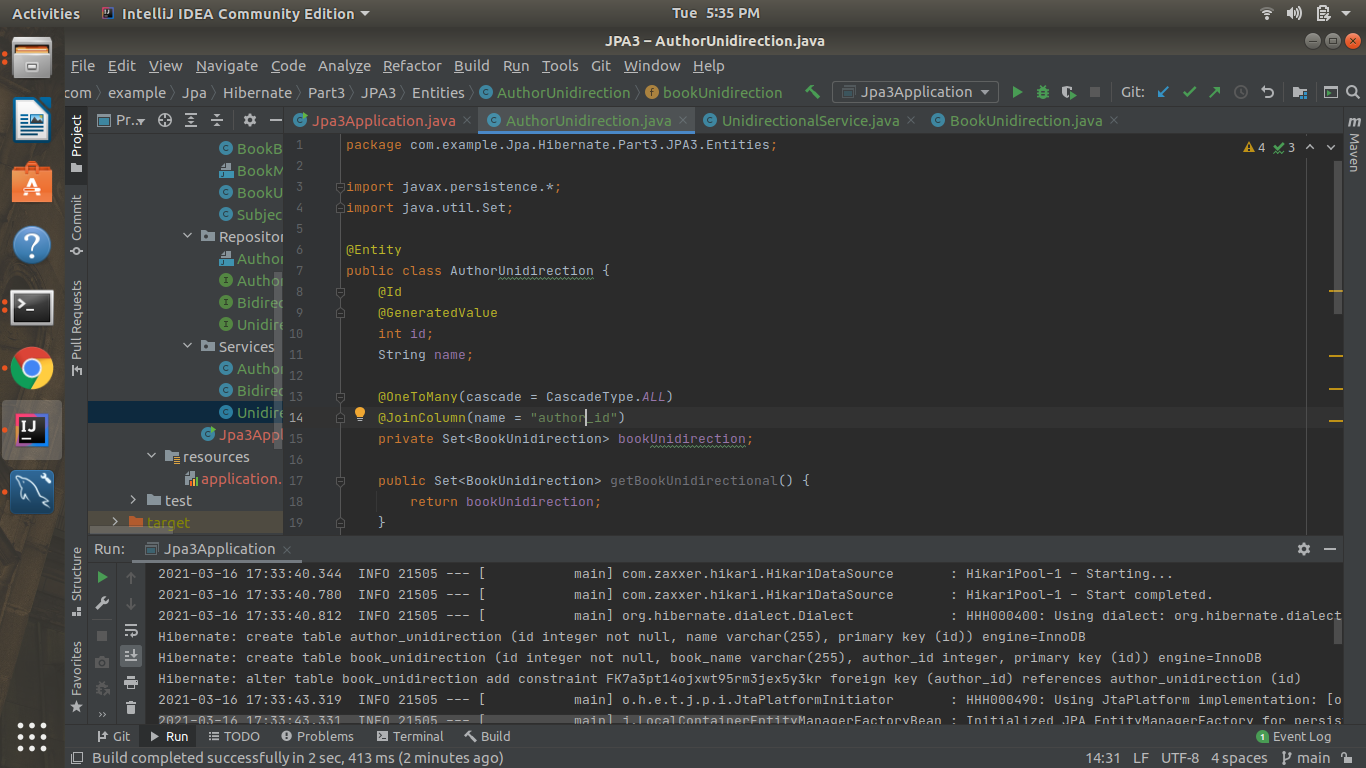


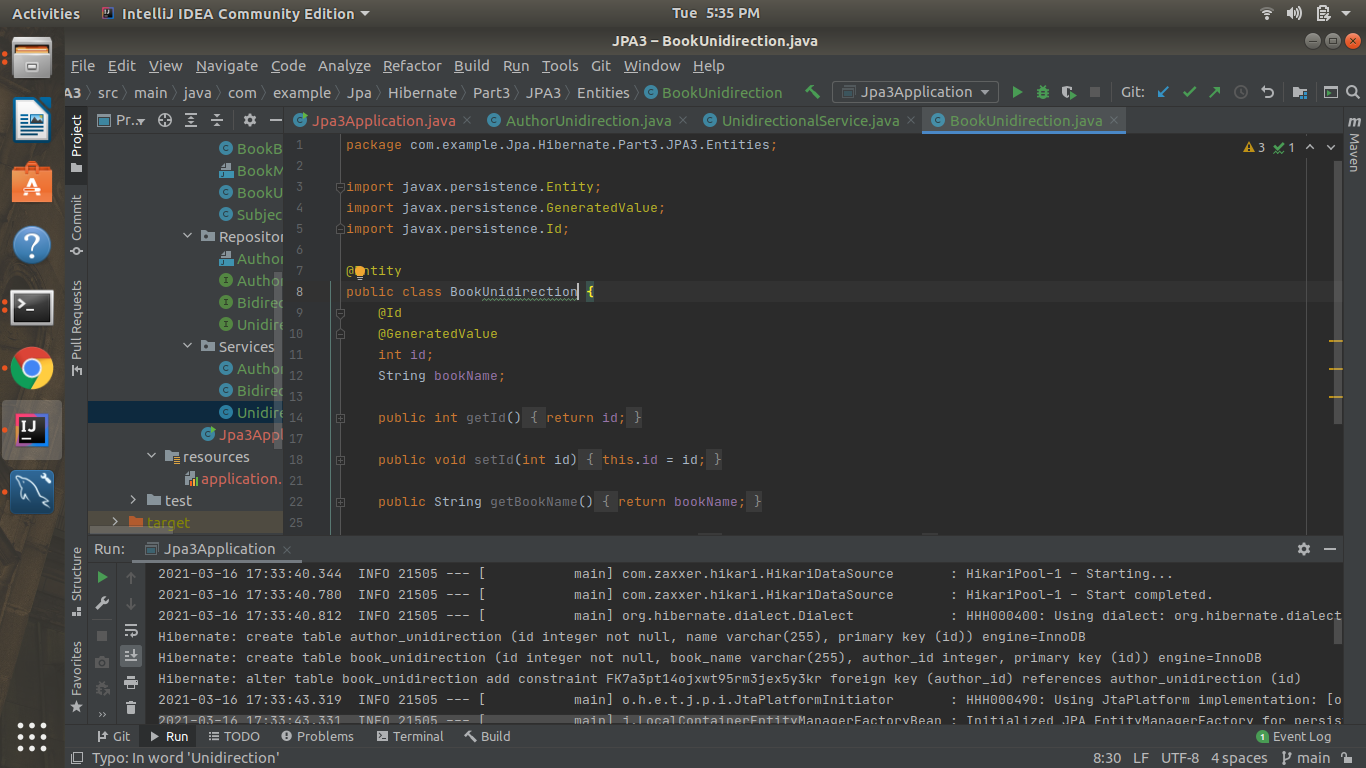
book table



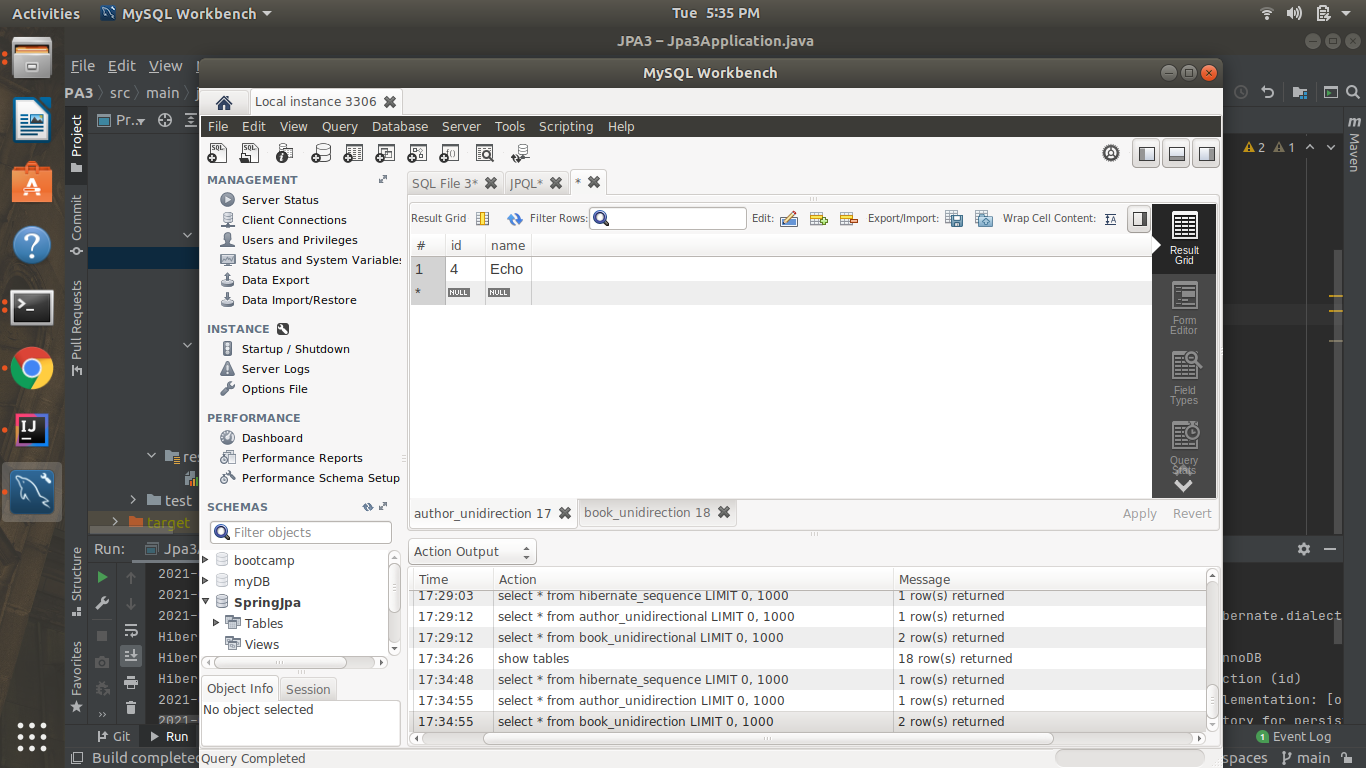
Q7)Implement One to Many Mapping between Author and Book(Unidirectional, BiDirectional and without additional table ) and implement cascade save.

**Unidirectional**

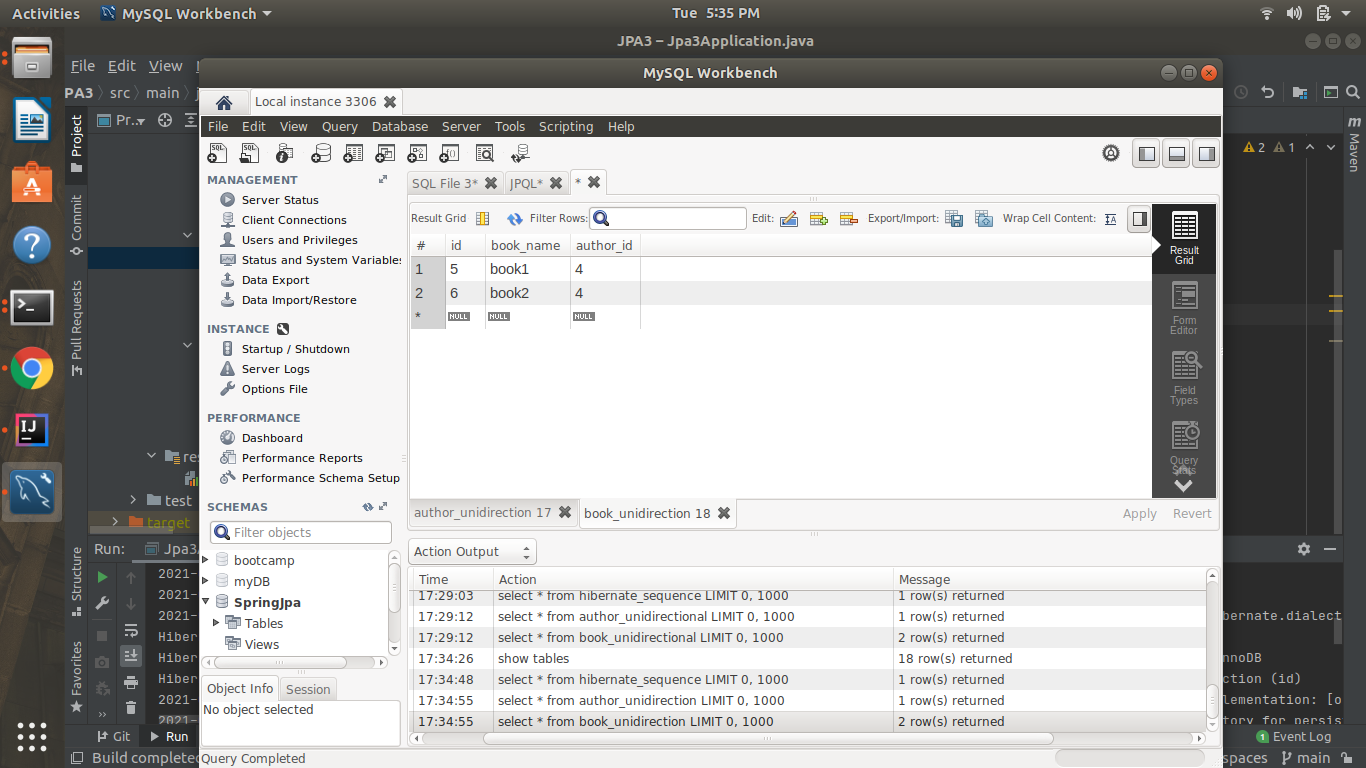
****

****

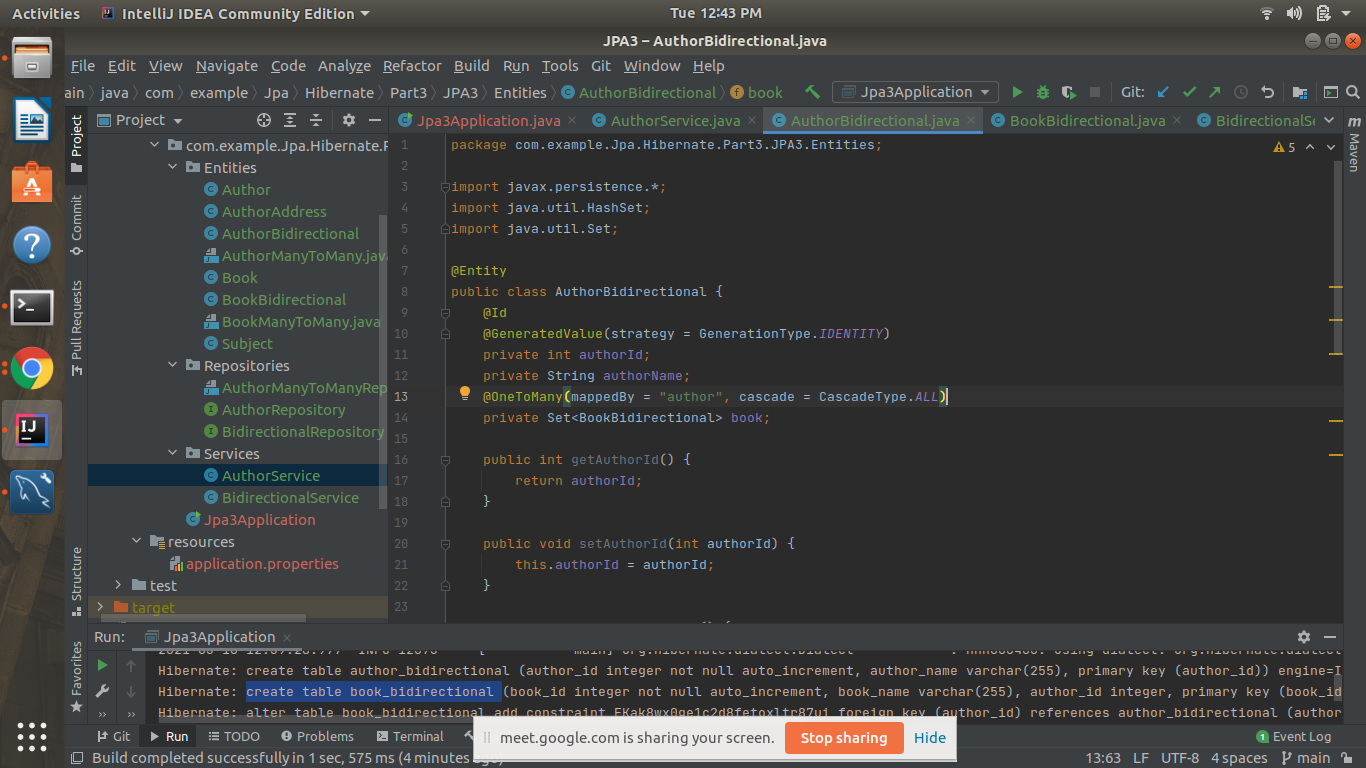
**author\_unidirection**

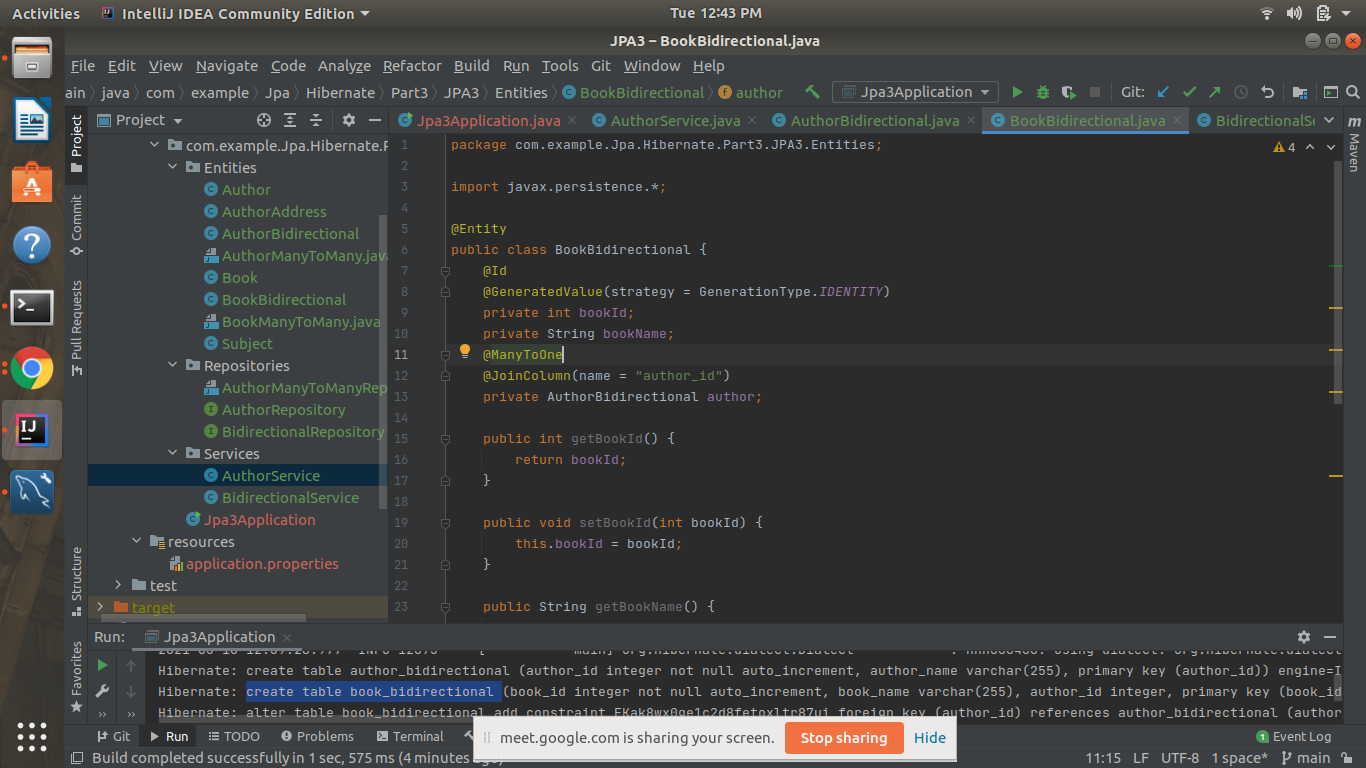
****

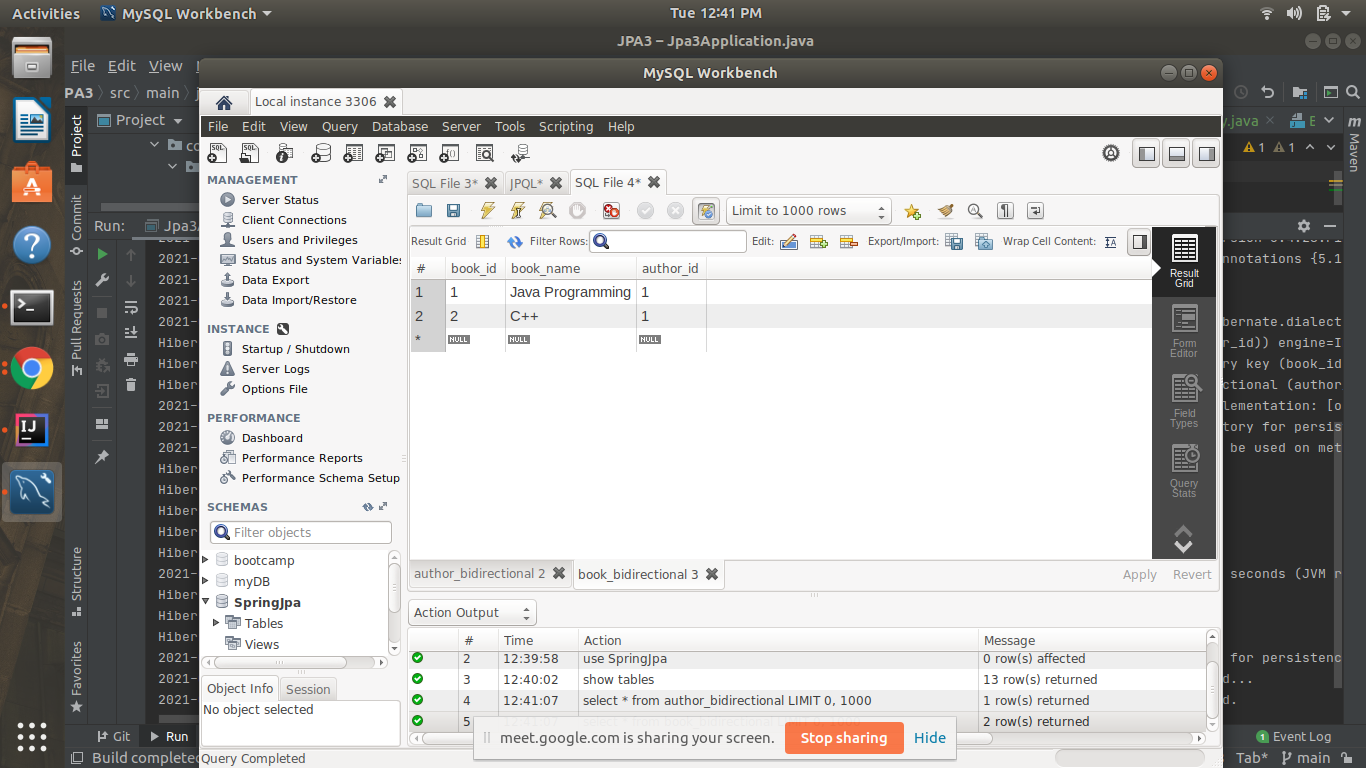
**book-unidirection**

****

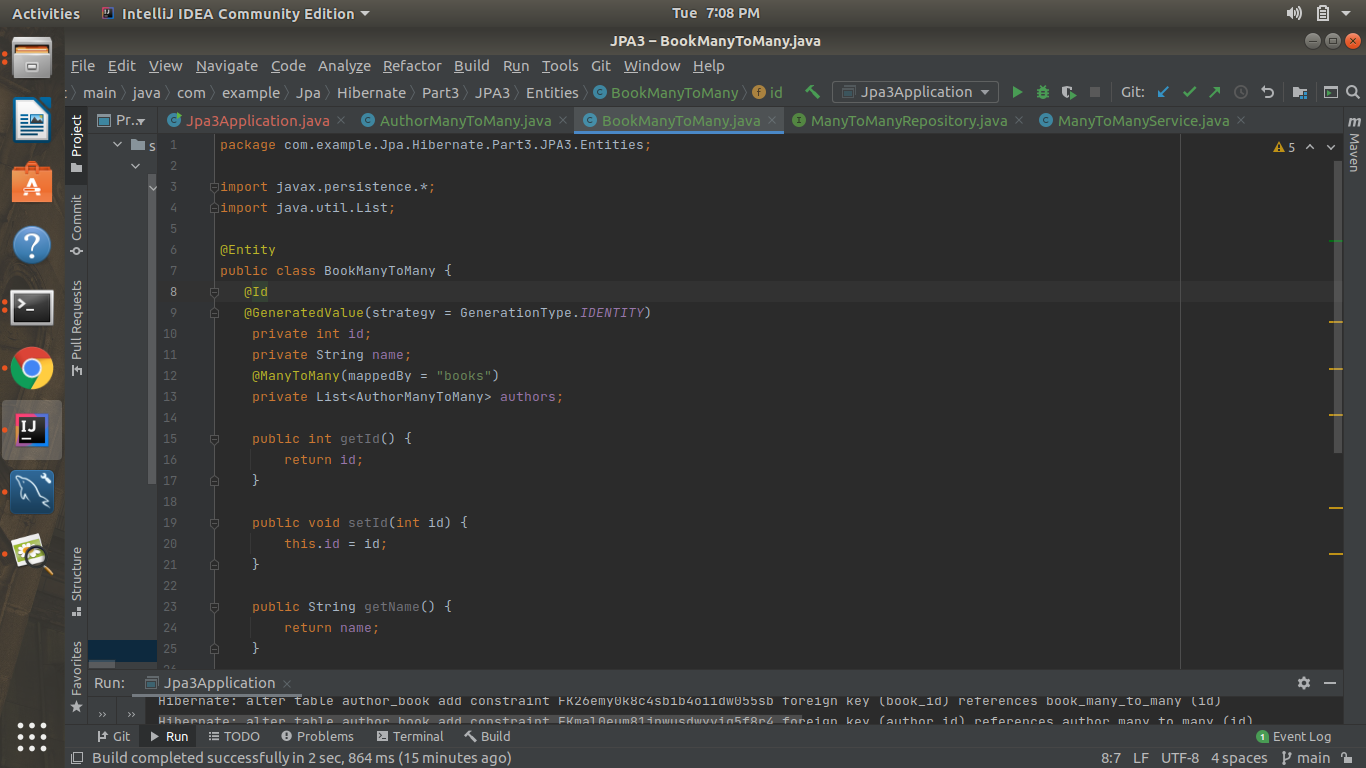
**Bidirectional**

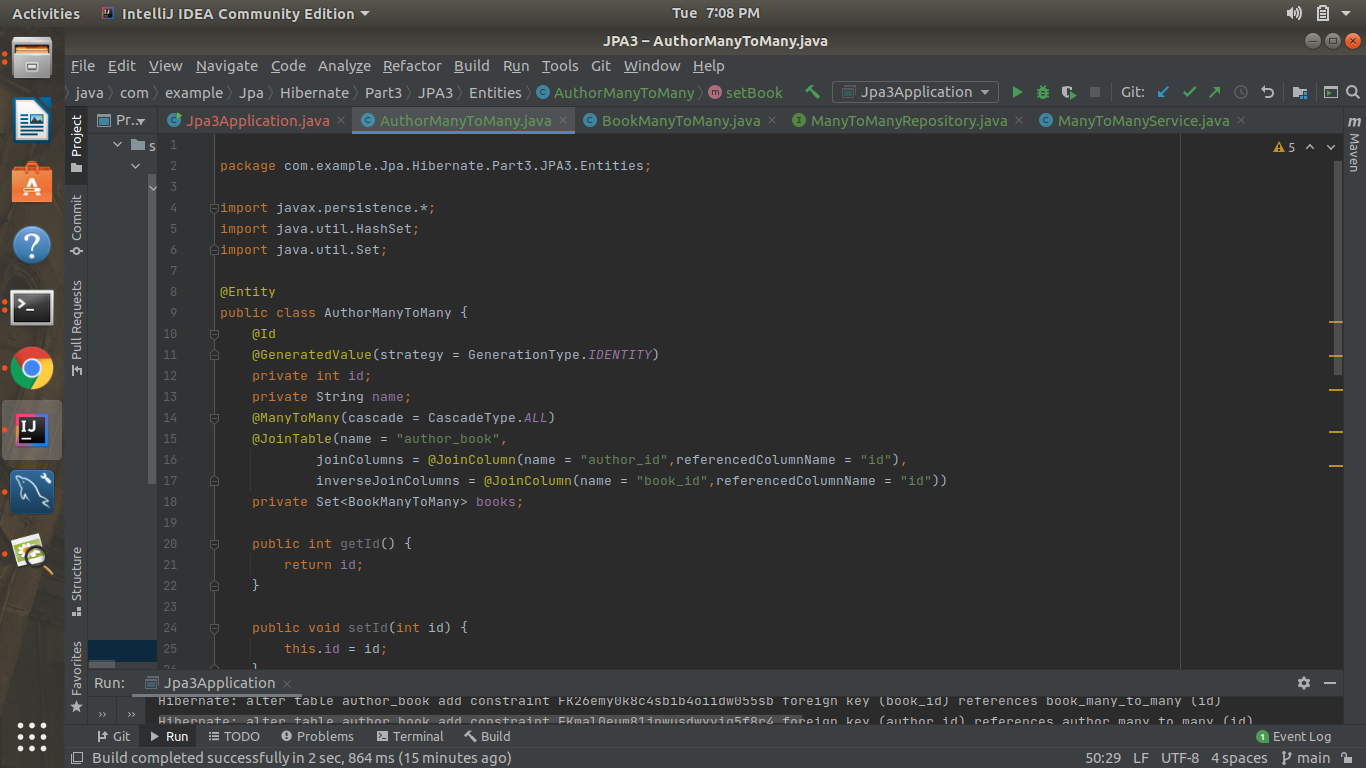




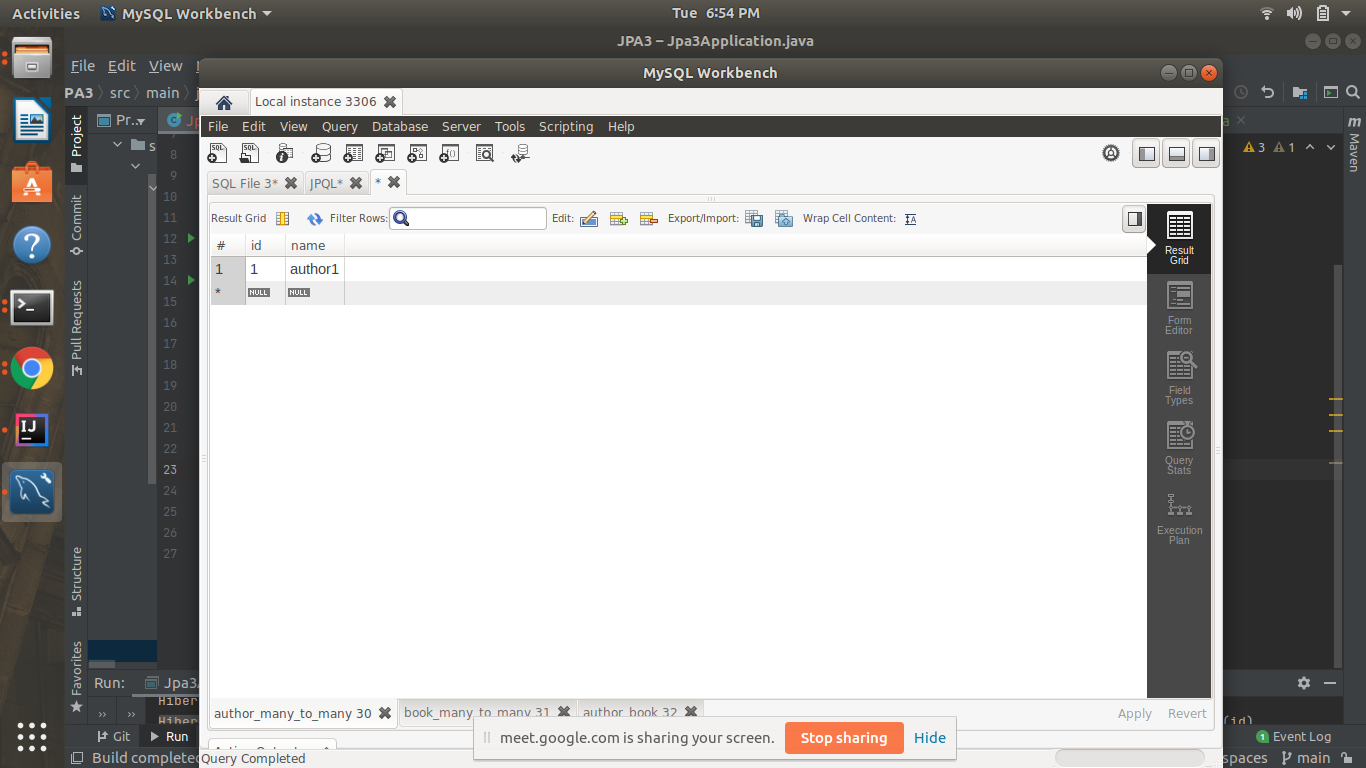


Q8)Implement Many to Many Mapping between Author and Book.

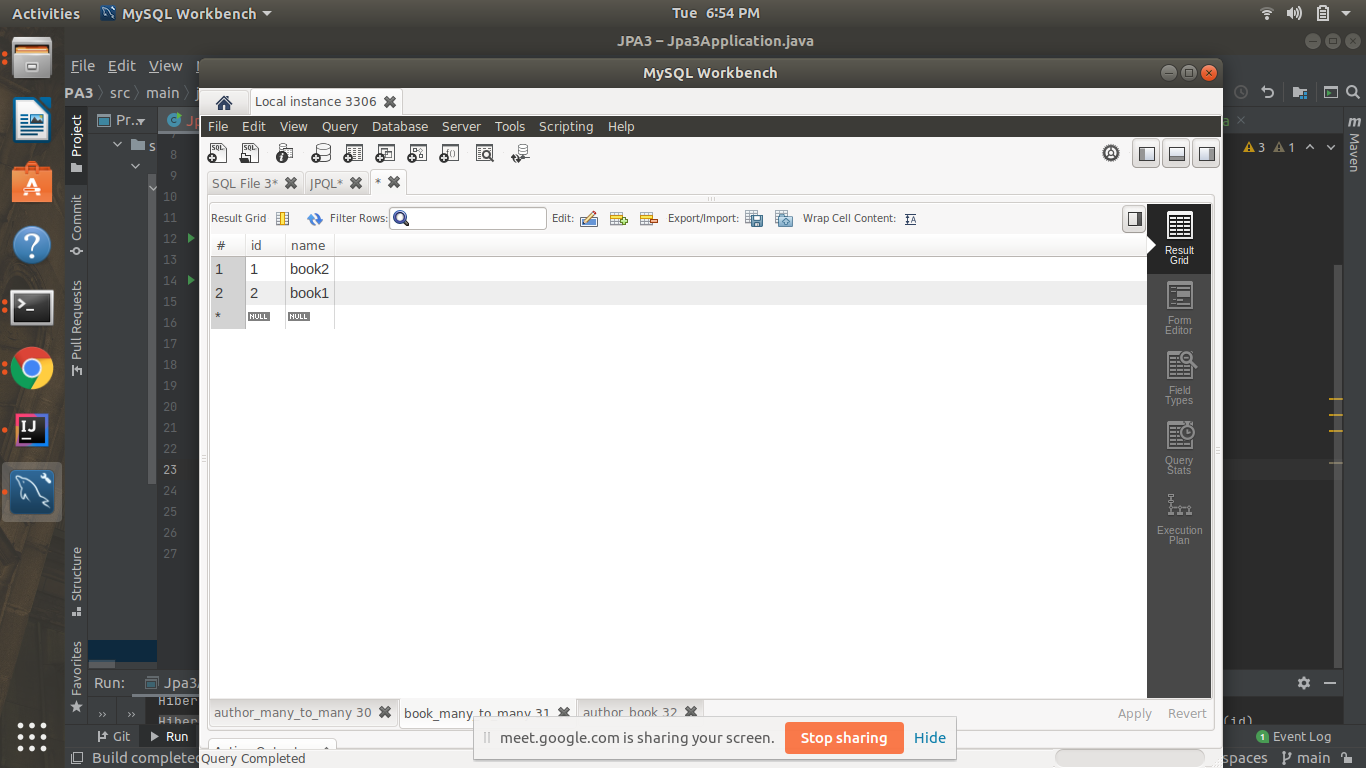




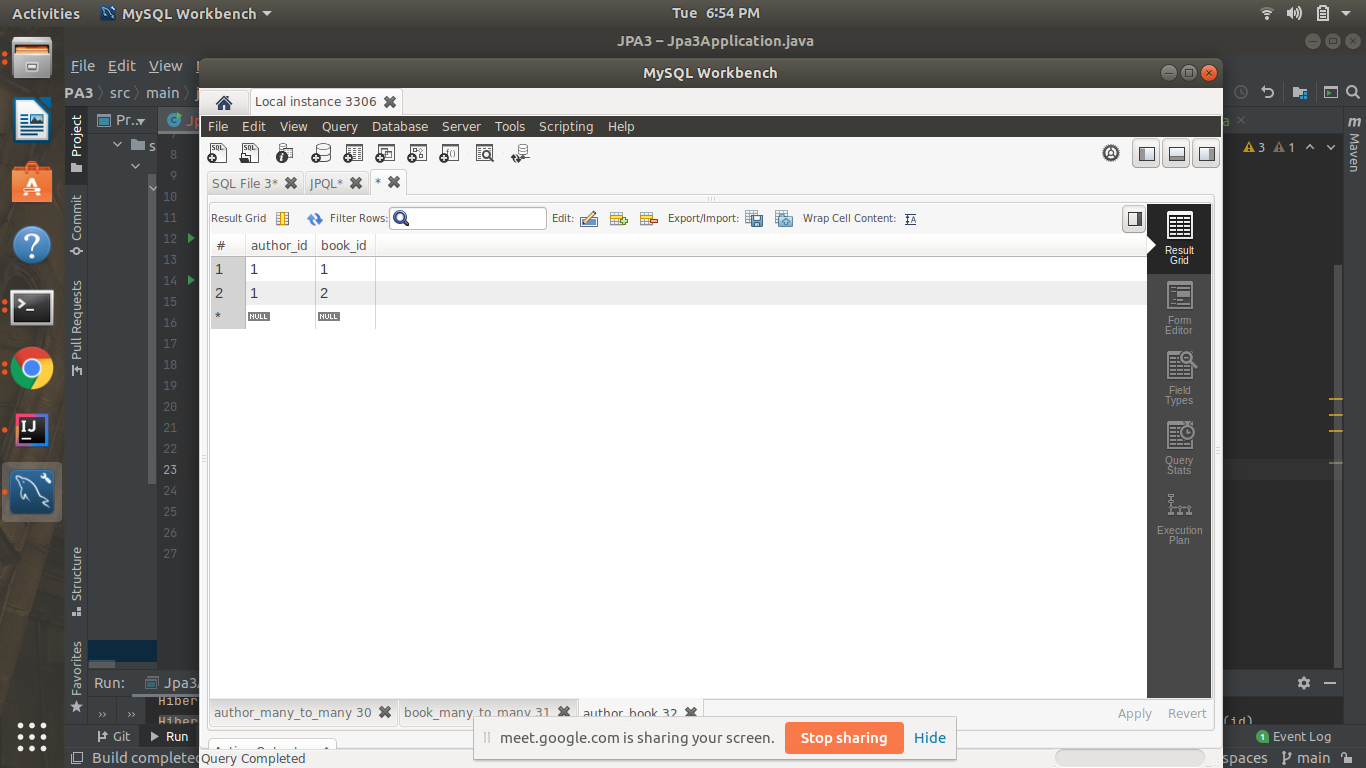
author\_many\_to\_many



book\_many\_to\_many

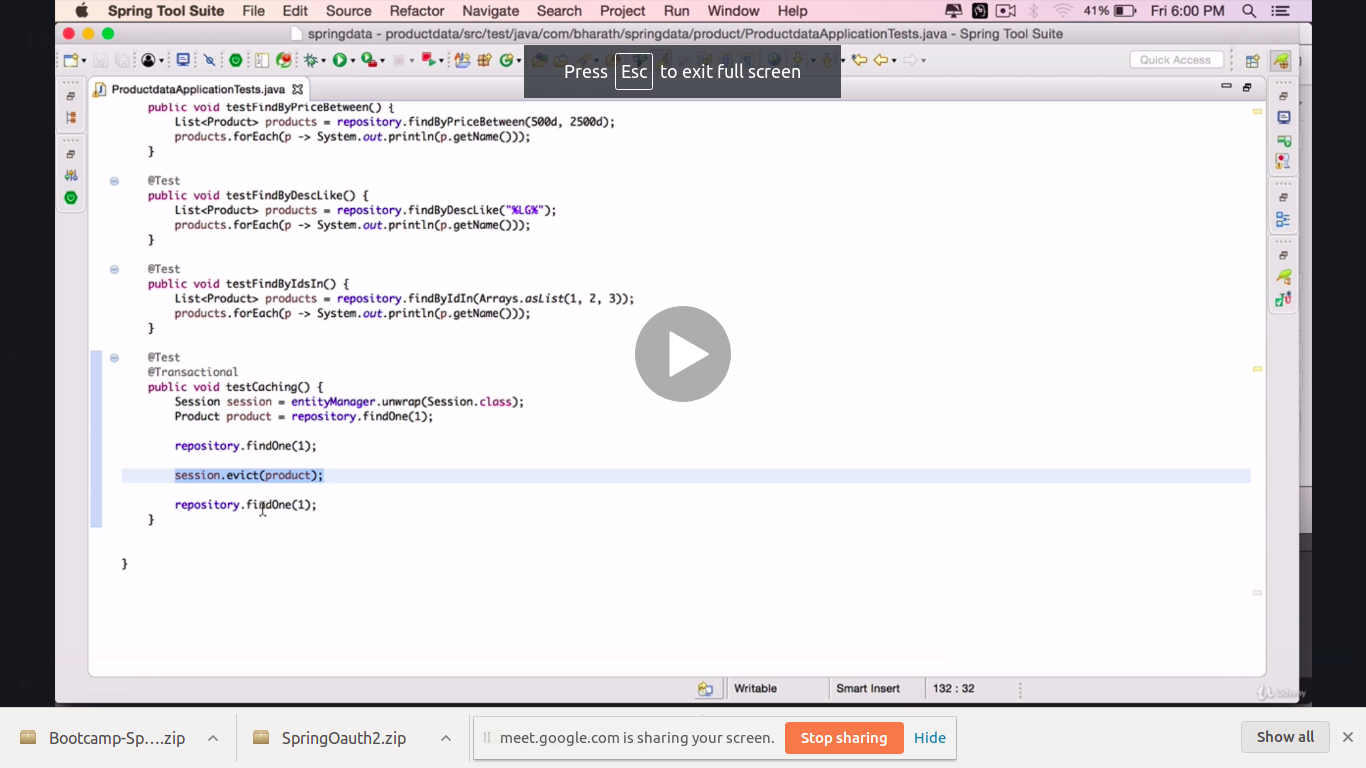


author\_book



Q9)Which method on the session object can be used to remove an object from the cache?

Answer-We use evict method of Session object and pass the object of an entity we want to remove from our cache



Here,considering only level 1 caching is enabled, repository.findOne(1) will run only once for first two but as the session.evict() method will remove the object from the cache, therefore it will run again for the last(third one) repository.findOne(1) and store the object in the cache again.

Q10)What does @transactional annotation do?

Answer-This annotation takes care of the database transactions. It starts the transaction with the database after the compilation of the taks it commits the transaction. If there is any exception in between the transaction it rolls back all the tasks leaving the database in stable state.It creates a proxy object during runtime that implements the same interface as the class with @Transactional and does the required task. When the task finishes, it commits or rollback accordingly.