**Task 04 || Spring Boot, ORM (JPA, Hibernate)**

1. **Duration:** 3 working days
2. **Description:**

* This task has 2 parts.
  1. Connect with a Relational Database (MySQL) in place of H2 DB.
  2. In Task 03, Book DTO and Book Entity has the same properties. In this task we will add some properties to the BookEntity to trace and version our data.
* create feature branch for task 04 from release branch:

go to release branch terminal and execute,

*git checkout -b feature/bim/atique-202015/task-04*

*[Note: use your nick name and emp\_id in place of “atique-202015”]*

* Part 01: Connect with MySQL DB
* Add below properties in *application.properties* file to configure MySQL DB

spring.datasource.driver-class-name=com.mysql.cj.jdbc.Driver  
spring.datasource.url=jdbc:mysql://localhost:3306/bookinfo  
spring.datasource.username=root  
spring.datasource.password=password  
spring.jpa.properties.hibernate.dialect=org.hibernate.dialect.MySQL5Dialect  
spring.jpa.hibernate.ddl-auto=update  
spring.jpa.show-sql=true

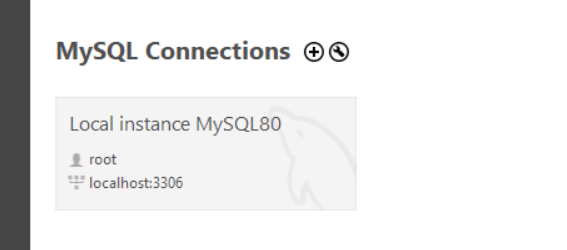
*[Note: we have already used Some of above properties in previous task, just replace their values. Some are new, simply add them, and notice* spring.datasource.url, *our database name will be* bookinfo*]*

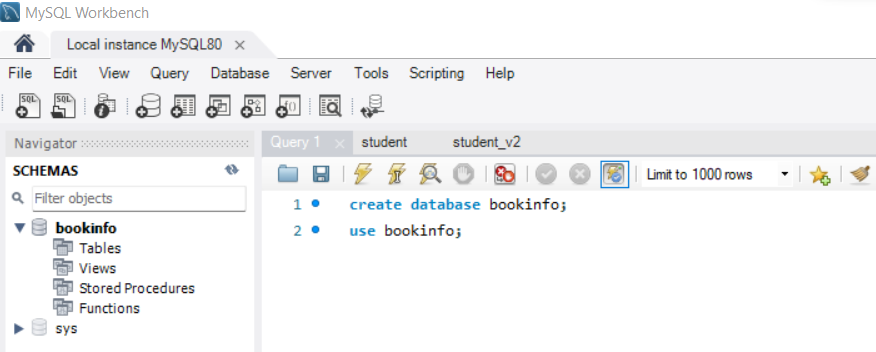
* Remove H2 DB driver dependency from *build.gradle → dependencies*

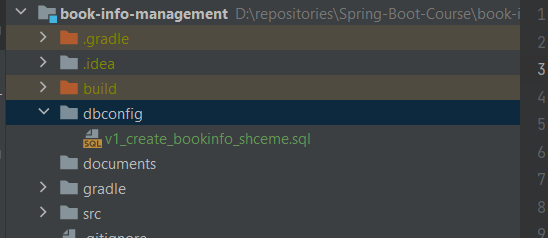
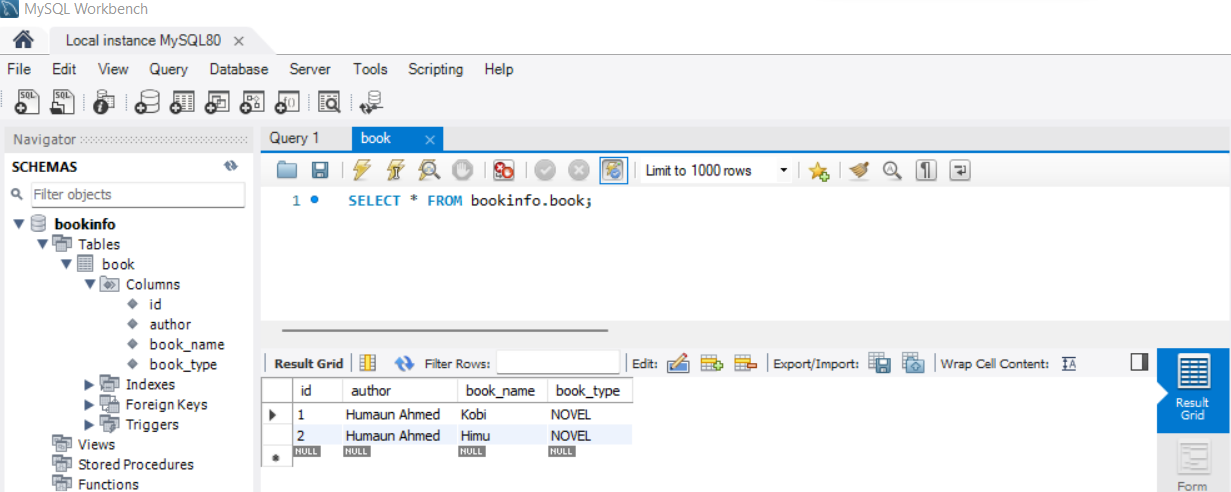
runtimeOnly 'com.h2database:h2'

* Add MySQL DB driver dependency to *build.gradle → dependencies*

runtimeOnly 'mysql:mysql-connector-java'

* You need to create the database manually, to do so first Log in to the ***"root"*** user in MySQL Workbench.
* Create bookinfo database/scheme in MySQL Workbench query console by following the screenshot bellow. bookinfo database will appear in left side Navigation panel. Press refresh button in case database not appear.



* Create *“dbconfig”* directoryunder *src/main/java/com.example.bookinfomanagement* and keep database creation query there in a .sql file like
* Run the Application, it will automatically create required tables in bookinfo database.
* Hit POST localhost:8084/book-info-manager-1.0/api/v1/student API with proper request, see data in *book* table.
* Done. Did you have noticed the beauty of JPA? We changed our entire DBMS from H2 to MySQL with some configuration changes in *application.properties* and MySQL driver dependency in *build.gradle*, without any change in codebase and everything just worked out of the box.
* We have used 2 new properties

1. spring.jpa.hibernate.ddl-auto=update
2. spring.jpa.show-sql=true

for first one, if we have set the value to “*update*” so, hibernate will update DB based on our annotated entity classes. That is why our table “*book*” created automatically when we started our application.

What are the other options for this property? see: <https://stackoverflow.com/questions/42135114/how-does-spring-jpa-hibernate-ddl-auto-property-exactly-work-in-spring>

for second one, the SQL generated by JPA will be logged in console, thus we can see what JPA is doing under the hood. See the console log

* open terminal, add & commit your changes for part 01

*Git add .*

*git commit -m "[bim] task 04 part 01"*

1. **Learning metatrails**
2. <https://www.baeldung.com/spring-boot-h2-database>
3. <https://www.youtube.com/watch?v=9SGDpanrc8U&ab_channel=Amigoscode>
4. <https://www.baeldung.com/learn-jpa-hibernate>
5. <https://www.baeldung.com/transaction-configuration-with-jpa-and-spring>
6. **Helper Project**

* This is an already done project to help you.
* Simple CRUD operations to manage student info is done in this project using H2 Database and JPA in 4 layers.

1. You have already cloned this project in task 02

2. fetch branches: *git fetch*

3. checkout branch: *git checkout hp/student-info-manage-h2-jpa*

4. Description can be found in README.md file.

5. Postman collection kept in *documents* package.

* You can see the project structure and browse the code.

1. **What to submit**

Prepare a doc with the answer of these two questions:

* What is ORM?
* Describe Persistence Life Cycle.
* Why JPA?
* How Builder Pattern work?
* Write about the newly added dependencies in *build.gradle* file.
* Write about the newly added properties in *application.properties* file.
* Write about new annotations we used:

@Builder, @Repository, @Entity, @Table, @Id, @Column, @Enumerated

* Submit your code for task 03 in GitHub. See how to submit section.

1. **How to submit**

* Name your doc as: <nick name>\_<emp\_id>\_<task id>.

Ex: atique\_202015\_task\_03.docx

* Upload your doc/pptx in documents folder
* commit your changes in your task 02 feature branch:

*git commit -m "[bim] task 03"*

* Push the changes to your feature branch: *git push*
* Create a merge req in GitLab [your feature branch→ your release branch]

Ex: *feature/bim/atique-202015/task-03 → bim/atique-202015/release-1.0*

* Follow up instructors’ feedback on GitHub merge request.