

MODULE 11

DATA PERSISTENCE *INTRODUCTION*

TABLE OF CONTENT

- Relational Database
- Data Persistence
- Object Relational Mapping
- Persistence Framework
- Java Persistence API
- Hibernate
- MySQL

Relational Database

- ▶ Values such as integers and strings organized within tables
- ▶ A table is a relation
 - I.e, each value in a row is related to other values in the same row

Data Persistence

- ▶ **Persistence** is the act of automatically storing data contained in Java objects into a relational database

Object Relational Mapping

- ▶ **Object Relational Mapping (ORM)**
 - For converting data from relational databases to object-oriented programming languages
 - To bridge the gap between an object-oriented model and a relational database

Persistence Framework

- ▶ Framework for Object Relational Mapping (**ORM**)

Java Persistence API

- ▶ The Java Persistence API (JPA) is a Java standards-based solution for persistence
- ▶ JPA is an object-relational mapping (ORM) technology
 - Transparent to the developer
- ▶ By creating Java classes (**Entity Beans**) that mirror the data model
 - ⇒ Access these entities rather than directly accessing the DB

Hibernate

- ▶ Open source Java persistence framework
- ▶ ORM framework
- ▶ Is an implementation of the Java Persistence API (JPA) specification
 - Can be used in any environment supporting JPA
 - Including Java SE applications, Java EE application servers ...

Hibernate

- ▶ For mapping an object-oriented domain model to a relational database
 - By replacing direct persistence-related database accesses with high-level object handling function
 - By mapping
 - From Java classes to database tables
 - From Java data types to SQL data types
- ▶ Provides
 - Data queries
 - Generate SQL calls
 - Retrieval facilities
 - Relieves from manual result set handling and object conversion

MySQL

- ▶ To connect to MySQL with Spring Boot
 - Add a dependency in pom.xml

```
<dependency>  
    <groupId>mysql</groupId>  
    <artifactId>mysql-connector-java</artifactId>  
</dependency>
```

MySQL

- ▶ Create an instance of MySQL
 - E.g.,
 - *MySQL56*
 - Running on *localhost*
 - Port: *3306*
 - User: *root*
 - Password: ... // remember it
- ▶ Start this instance of MySQL
 - Task Manager ⇒ Services ⇒ *MySQL56* ⇒ Start

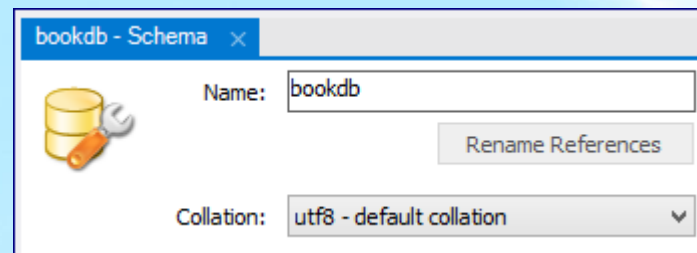
MySQL

- ▶ Use a visual tool for database management
- ▶ E.g, MySQL **Workbench**
 - For DBA, developer or data architect
 - To visually design, model, generate, and manage databases

MySQL

- ▶ Create a new schema

- E.g, *bookdb* schema



MySQL

- ▶ Configure Spring Boot to access MySQL database
- ▶ In **Application.yml**
 - Configure a spring datasource

```
# Local server
server:
  # port is used by spring-boot-admin
  port: 8080
  contextPath: /book

spring:
  datasource:
    password: ...
    url: jdbc:mysql://localhost:3306/bookdb?user=root
```

Root of the project

Password of the MySQL instance

User of the MySQL instance

Name of the database schema