How does Hibernate / JPA relate to JDBC?

Hibernate / JPA and JDBC

- Hibernate / JPA uses JDBC for all database communications

MySQL Database

MySQL Database

- In this course, we will use the MySQL Database

- MySQL includes two components
  - MySQL Database Server
  - MySQL Workbench

MySQL Database Server

- The MySQL Database Server is the main engine of the database
- Stores data for the database
- Supports CRUD features on the data

MySQL Workbench

- MySQL Workbench is a client GUI for interacting with the database
- Create database schemas and tables
- Execute SQL queries to retrieve data
- Perform insert, updates and deletes on data
- Handle administrative functions such as creating users
- Others ...

Install the MySQL software

- Step 1: Install MySQL Database Server
  - `https://dev.mysql.com/downloads/mysql/`
- Step 2: Install MySQL Workbench
  - `https://dev.mysql.com/downloads/workbench/`

Please install the MySQL software now

Setup Database Table

Two Database Scripts

1. Folder: **00-starter-sql-scripts**

   - **01-create-user.sql**
   - **02-student-tracker.sql**

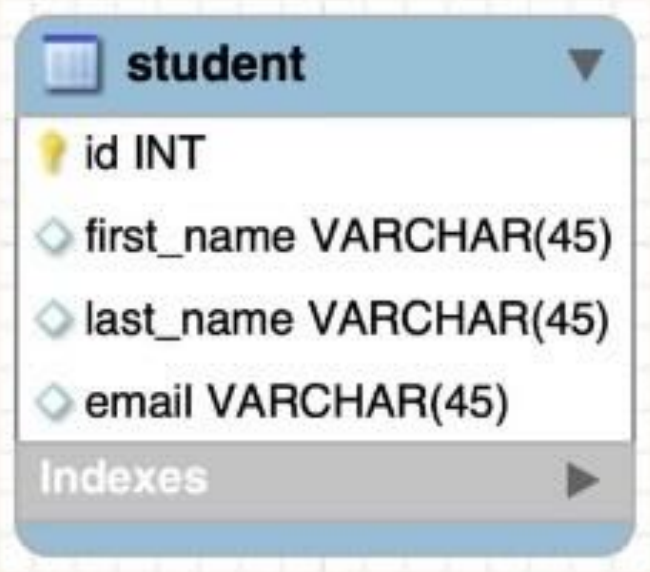
About: 01-create-user.sql

1. Create a new MySQL user for our application

- user id: **springstudent**
- password: **springstudent**

About: 02-student-tracker.sql

1. Create a new database table: **student**



Setting Up Spring Boot Project

Automatic Data Source Configuration

- In Spring Boot, Hibernate is the default implementation of JPA
- **`EntityManager`** is main component for creating queries etc …
- **`EntityManager`** is from Jakarta Persistence API (JPA)

Automatic Data Source Configuration

- Based on configs, Spring Boot will automatically create the beans:
  - `DataSource, EntityManager, ...`

- You can then inject these into your app, for example your DAO

Setting up Project with Spring Initialzr

- At Spring Initializr website, start.spring.io
- Add dependencies
  - MySQL Driver: **`mysql-connector-j`**
  - Spring Data JPA: **`spring-boot-starter-data-jpa`**

Spring Boot - Auto configuration

- Spring Boot will automatically configure your data source for you

- Based on entries from Maven pom file
  - JDBC Driver: **`mysql-connector-j`**
  - Spring Data (ORM): **`spring-boot-starter-data-jpa`**

- DB connection info from **`application.properties`**

application.properties

```
spring.datasource.url=jdbc:mysql://localhost:3306/student_tracker
spring.datasource.username=springstudent
spring.datasource.password=springstudent
```

No need to give JDBC driver class name
Spring Boot will automatically detect it based on URL

Creating Spring Boot - Command Line App

- We will create a Spring Boot - Command Line App
- This will allow us to focus on Hibernate / JPA
- Later in the course, we will apply this to a CRUD REST API

Creating Spring Boot - Command Line App

```
import org.springframework.boot.CommandLineRunner;
import org.springframework.boot.SpringApplication;
import org.springframework.boot.autoconfigure.SpringBootApplication;
import org.springframework.context.annotation.Bean;

@SpringBootApplication
public class CruddemoApplication {

    public static void main(String[] args) {
        SpringApplication.run(CruddemoApplication.class, args);
    }

    @Bean
    public CommandLineRunner commandLineRunner(String[] args) {
        return runner -> {
            System.out.println("Hello world");
        };
    }
}
```

Executed after the Spring Beans have been loaded

Lambda expression

Add our custom code