Spring Boot MVC JPA H2

Agenda

1 **Spring Boot MVC JPA H2**

Objectives

At the end of this module, you will be able to:

- Understand, How to create a MVC web application by using JPA and H2 database to perform CRUD operations in spring boot.

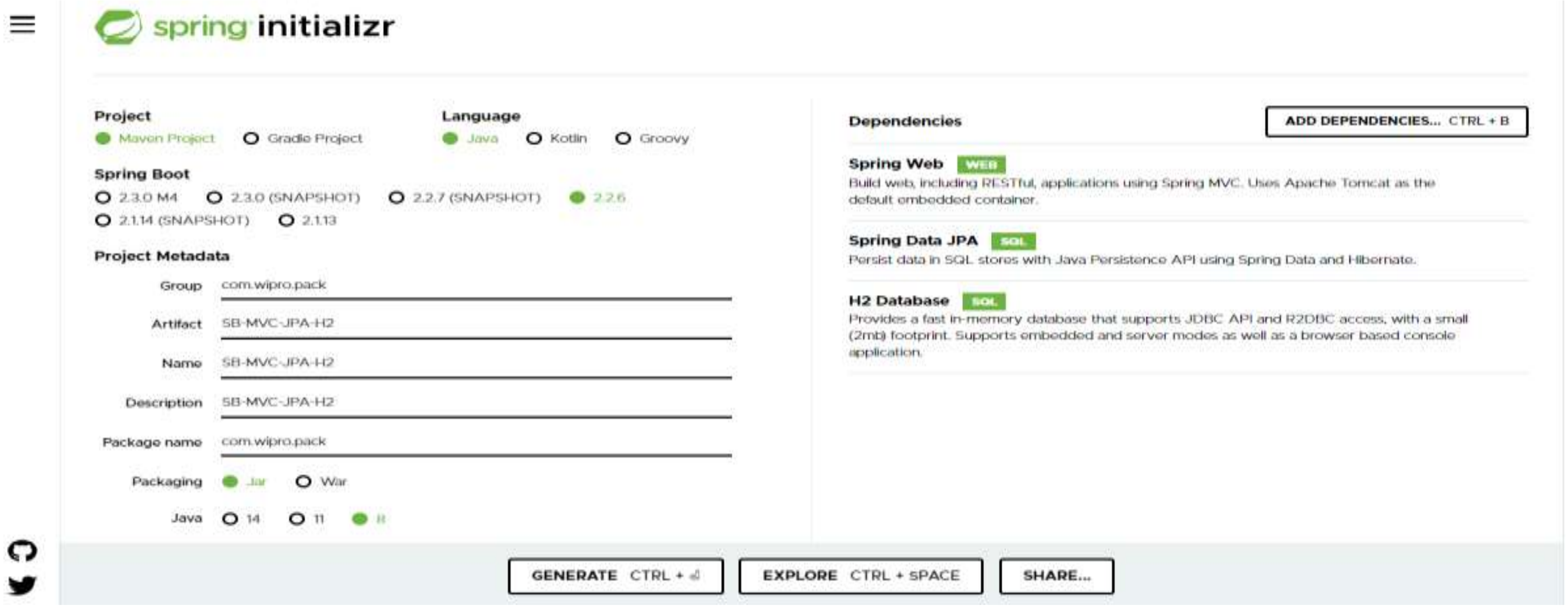
Spring Boot MVC JPA H2

What is H2?

- H2 is an open-source lightweight Java database.
- It can be embedded in Java applications or run in the client-server mode.
- Mainly, H2 database can be configured to run as inmemory database, which means that data will not persist on the disk.
- Because of embedded database it is not used for production development, but mostly used for development and testing.

Spring Boot MVC JPA H2.

- We will create a spring boot project along with web, jpa and h2 dependencies. Like below.



spring initializr

Project
[✓ Maven Project] Gradle Project

Language
[✓ Java] Kotlin Groovy

Spring Boot
2.3.0 M4 2.3.0 (SNAPSHOT) 2.2.7 (SNAPSHOT) [✓ 2.2.6]
2.1.14 (SNAPSHOT) 2.1.13

Project Metadata

Group	com.wipro.pack
Artifact	SB-MVC-JPA-H2
Name	SB-MVC-JPA-H2
Description	SB-MVC-JPA-H2
Package name	com.wipro.pack
Packaging	[✓ Jar] War
Java	14 11 [✓ 8]

Dependencies

Spring Web WEB
Build web, including RESTful, applications using Spring MVC. Uses Apache Tomcat as the default embedded container.

Spring Data JPA SQL
Persist data in SQL stores with Java Persistence API using Spring Data and Hibernate.

H2 Database SQL
Provides a fast in-memory database that supports JDBC API and R2DBC access, with a small (2mb) footprint. Supports embedded and server modes as well as a browser based console application.

Spring Boot MVC JPA H2.

- We will create a model class “Employee” and Home.jsp like below.

```
package com.wipro.pack.model;
public class Employee {
             private int eid;
             private String ename;
             private int esalary;
             //Getters & Setters
             @Override
             public String toString() {
                          return "Employee [eid=" + eid + ", ename=" + ename + ", esalary=" + esalary + "]";
             }
}
```

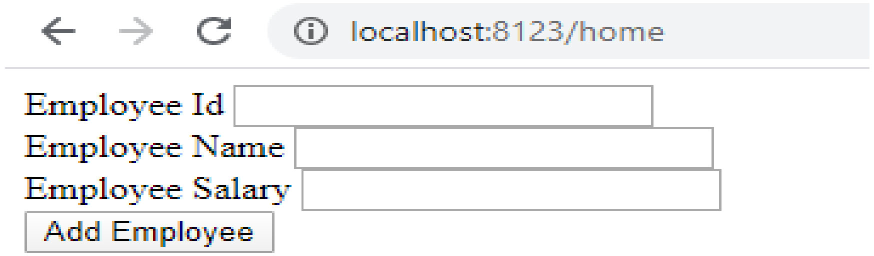
```
<body>
<form action="AddEmployee">
Employee Id <input type="text" name="eid"/><br>
Employee Name <input type="text" name="ename"/><br>
Employee Salary <input type="text" name="esalary"/><br>
<input type="Submit" value="Add Employee"/>
</form>
</body>
```

Spring Boot MVC JPA H2.

- We will create controller “EmployeeController” like below.

```
package com.wipro.pack.controller;
import org.springframework.stereotype.Controller;
import org.springframework.web.bind.annotation.RequestMapping;
@Controller
public class EmployeeController {
             @RequestMapping("/home")
             public String home() {
                          return "Home";
             }
}
```

- Now our output will be like below.



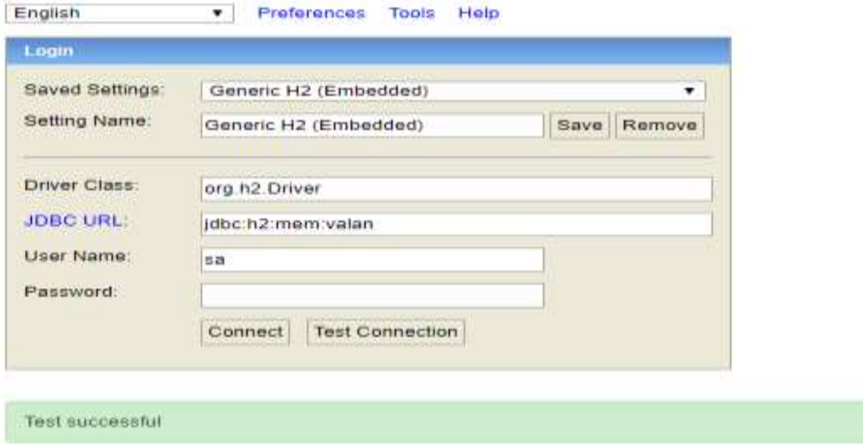
Employee Id	
Employee Name	
Employee Salary	

Spring Boot MVC JPA H2.

- Now we will enable H2 data base in properties file like below.

```
server.port=8123
spring.mvc.view.suffix=.jsp
spring.h2.console.enabled=true
spring.datasource.platform=h2
spring.datasource.url=jdbc:h2:mem:valan
```

- Relaunch our application and check the url - localhost:8123/h2-console and test connection.

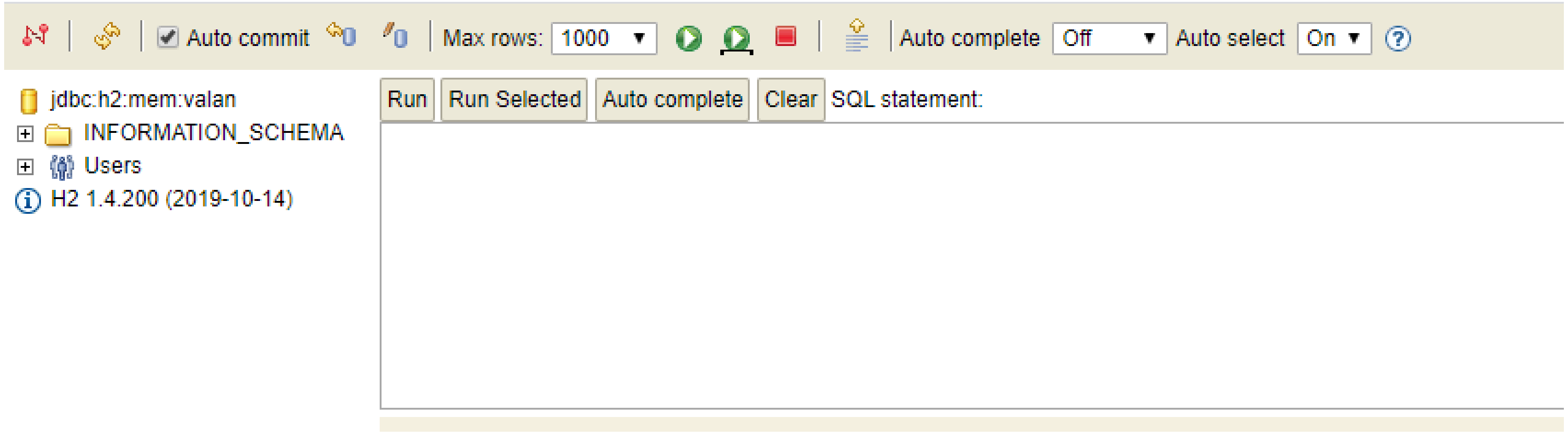


Saved Settings: Generic H2 (Embedded)
Setting Name: Generic H2 (Embedded)
Driver Class: org.h2.Driver
JDBC URL: jdbc:h2:mem:valan
User Name: sa
Password:

Test successful

Spring Boot MVC JPA H2.

- Now click connect to connect with our database. You will get like below.



jdbc:h2:mem:valan
INFORMATION_SCHEMA
Users
H2 1.4.200 (2019-10-14)

- You can look at our database, there is no table is created.

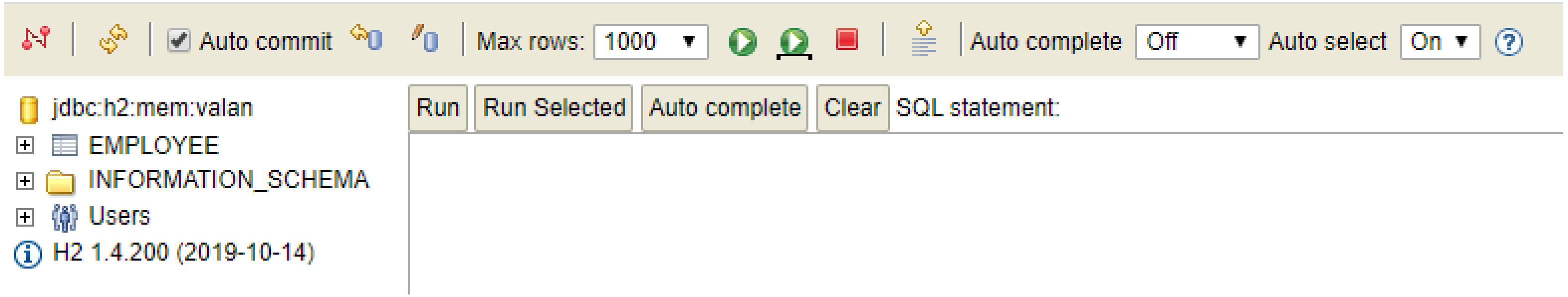
Spring Boot MVC JPA H2.

- To create a table in H2 data base by using our Employee model, We will use two annotations @Entity and @Id like below.

```
package com.wipro.pack.model;
import javax.persistence.Entity;
import javax.persistence.Id;
@Entity
public class Employee {
             @Id
             private int eid;
             private String ename;
             private int esalary;
             //Getters & Setters
             @Override
             public String toString() {
                          return "Employee [eid=" + eid + ", ename=" + ename + ", esalary=" + esalary + "]";
             }
}
```

Spring Boot MVC JPA H2.

- Relaunch our application and check the H2 data base. Now we are getting the table.



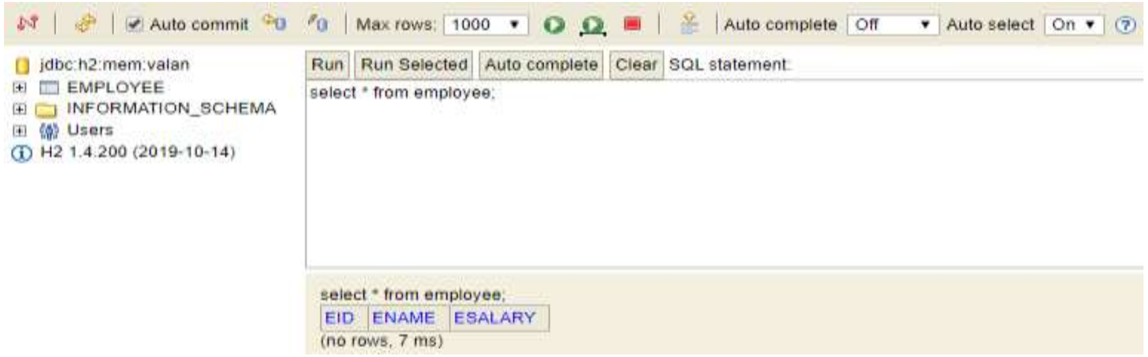
Auto commit   Max rows: 1000   Auto complete Off   Auto select On

Run   Run Selected   Auto complete   Clear   SQL statement:

jdbc:h2:mem:valan
- EMPLOYEE
- INFORMATION_SCHEMA
- Users

H2 1.4.200 (2019-10-14)

- But there is no record.



Auto commit   Max rows: 1000   Auto complete Off   Auto select On

Run   Run Selected   Auto complete   Clear   SQL statement:

jdbc:h2:mem:valan
- EMPLOYEE
- INFORMATION_SCHEMA
- Users

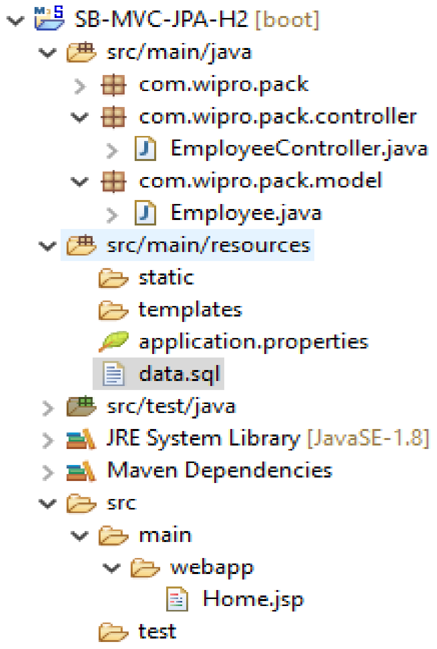
H2 1.4.200 (2019-10-14)

```
select * from employee;
```

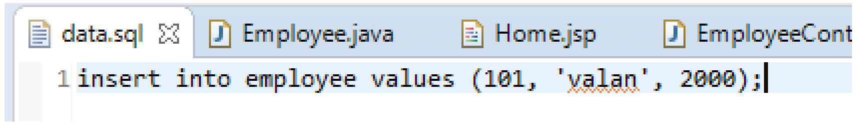
```
select * from employee;
EID  ENAME  ESALARY
(no rows, 7 ms)
```

Spring Boot MVC JPA H2.

- In case, If we want to execute any pre handed queries. Then we need to create a file “data.sql” and inside this file we can have our pre handed queries. Like below.



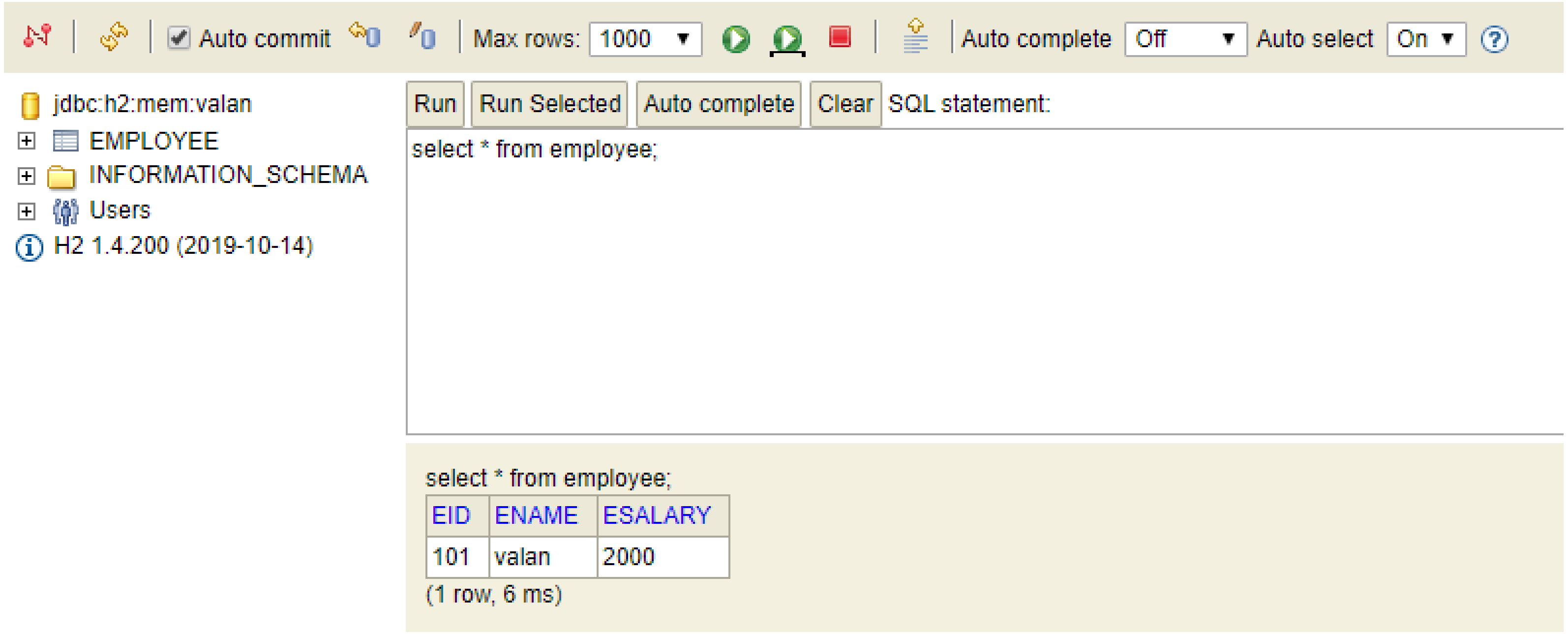
```
SB-MVC-JPA-H2 [boot]
  src/main/java
    com.wipro.pack
    com.wipro.pack.controller
      EmployeeController.java
    com.wipro.pack.model
      Employee.java
  src/main/resources
    static
    templates
    application.properties
    data.sql
  src/test/java
  JRE System Library [JavaSE-1.8]
  Maven Dependencies
  src
    main
      webapp
        Home.jsp
    test
```



```sql
insert into employee values (101, 'valan', 2000);
```

Spring Boot MVC JPA H2.

- Now relaunch our application and test our data base, we will get one record as like below.



```
select * from employee;
```

select * from employee;

EID	ENAME	ESALARY
101	valan	2000

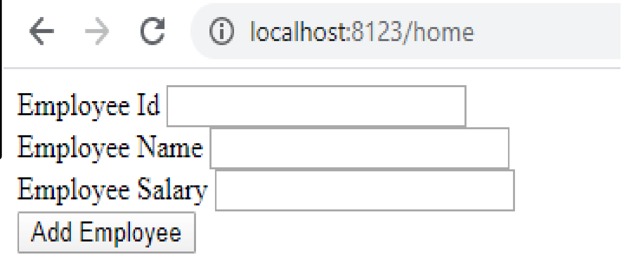
(1 row, 6 ms)

Spring Boot MVC JPA H2.

- Just we have look on our “EmployeeController” and “Home.jsp”. And we will insert a employee details via our application.

```java
package com.wipro.pack.controller;
import org.springframework.stereotype.Controller;
import org.springframework.web.bind.annotation.RequestMapping;
@Controller
public class EmployeeController {
		@RequestMapping("/home")
		public String home() {
			return "Home";
		}
}
```

```html
<body>
<form action="AddEmployee">
Employee Id <input type="text" name="eid"/><br>
Employee Name <input type="text" name="ename"/><br>
Employee Salary <input type="text" name="esalary"/><br>
<input type="Submit" value="Add Employee"/>
</form>
</body>
```


localhost:8123/home

Employee Id
Employee Name
Employee Salary

Spring Boot MVC JPA H2.

- Now we will create a “EmployeeDao” interface like below.

```
package com.wipro.pack.dao;

import org.springframework.data.repository.CrudRepository;

import com.wipro.pack.model.Employee;

public interface EmployeeDao extends CrudRepository<Employee, Integer>{

}
```

- Here we are extending an interface “CrudRepository”. It will provide all CRUD operations functionality. We need not write any code for this CRUD operations.

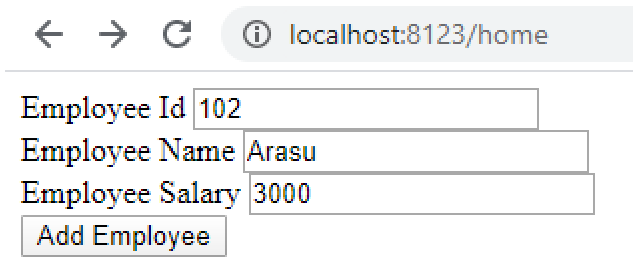
Spring Boot MVC JPA H2.

- Now we will use this “EmployeeDao” interface in our “EmployeeController” to insert a record into our H2 database employee table like below.

```
@Controller
public class EmployeeController {
		@Autowired
		EmployeeDao dao;
		@RequestMapping("/home")
		public String home() {
			return "Home";
		}
		@RequestMapping("/AddEmployee")
		public ModelAndView addEmployee(Employee emp) {
			ModelAndView mv = new ModelAndView();
			dao.save(emp);
			mv.addObject("message", "Record Inserted");
			mv.setViewName("Home");
			return mv;
		}
}
```

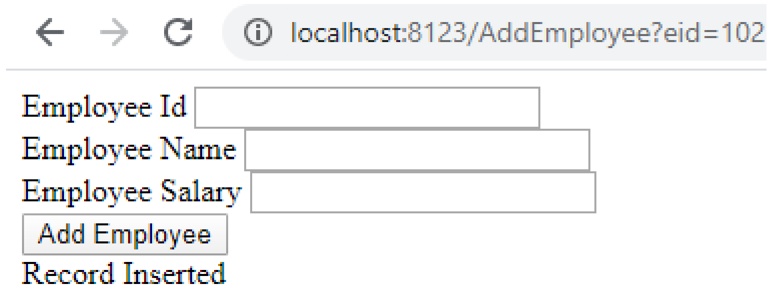
Spring Boot MVC JPA H2.

- Relaunch our application and test the output.



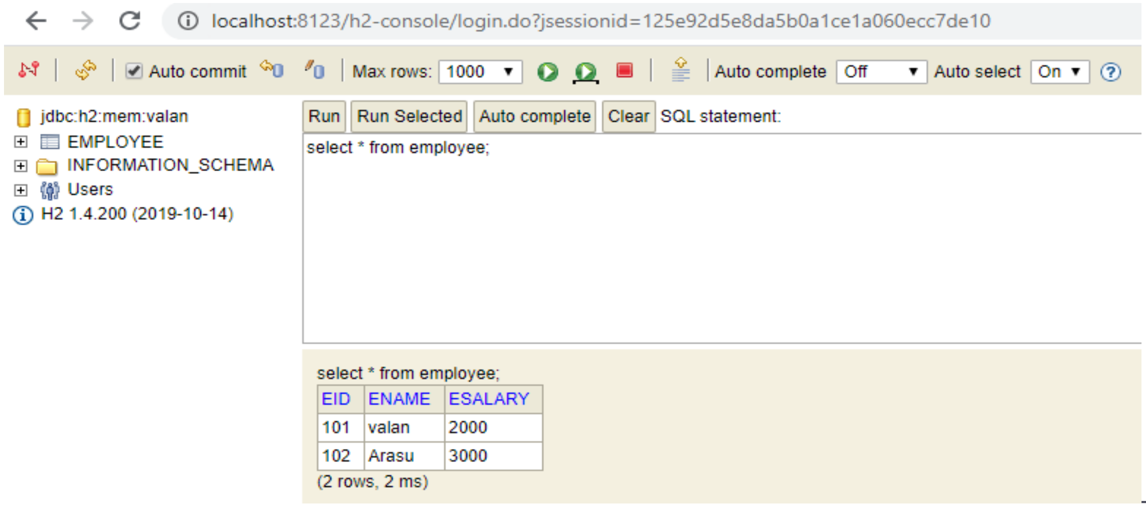
localhost:8123/home

Employee Id 102
Employee Name Arasu
Employee Salary 3000
Add Employee



localhost:8123/AddEmployee?eid=102

Employee Id
Employee Name
Employee Salary
Add Employee
Record Inserted



localhost:8123/h2-console/login.do?jsessionid=125e92d5e8da5b0a1ce1a060ecc7de10

Auto commit ✓ | Max rows: 1000 | Auto complete: Off | Auto select: On

jdbc:h2:mem:valan
EMPLOYEE
INFORMATION_SCHEMA
Users
H2 1.4.200 (2019-10-14)

Run | Run Selected | Auto complete | Clear | SQL statement:

```
select * from employee;
```

select * from employee;

EID	ENAME	ESALARY
101	valan	2000
102	Arasu	3000

(2 rows, 2 ms)

Spring Boot MVC JPA H2.

- Other CRUD methods from “CrudRepository” interface.
    - count()
    - delete(Entity entity)
    - deleteAll()
    - deleteById(Integer id)
    - findAll()
    - findById(Integer id)

Summary

In this session, you have learned about:

- How to create a MVC web application by using JPA and H2 database to perform CRUD operations in spring boot.

Thank you

Sensitivity: Internal & Restricted