Date : 03/10/2020

Spring Boot 7AM

Mr. RAGHU

----------------------------------------------

**ORM : Object Relational Mapping**

Mapping: Linking :

Entity/Model class--- DB table

variable ---- column

object --- row

JPA Annotations:

@Entity

@Id (PK)

@Table (optional, if not provided default is class name)

@Column(optional, if not provided default is variable name)

\*)Embedded Database:- No Download + No Install DB, RAM(Temp), use only for DevEnv.

3 Embedded DB : H2, Apache Derby, Hyper SQL(HSQL).

========================================================================

driver class

url

username

password

dialect - class, Generate SQL query when we perform operation

show\_sql - if true, it will display SQL query on console/log

ddl-auto - create/update/validate/create-drop

Q) What is default value for ddl-auto?

A) For Embedded Database default is : create-drop

For Extrnal Database(Downloaded and Installed DB) default is: validate

---------------------------------------------------Data JPA Ex#1--------------------------------------------------------

#1. Create one Starter project

Name : SpringBoot2DataJpaH2Ex

Dep : Data Jpa, H2, Lombok, Spring web(This is used to view the embedded database such as h2 in console.

\*)Note: Spring web applications works using servers(default: Tomcat with PORT:8080)

#2. application.properties

server.port=9090

spring.datasource.url=jdbc:h2:mem:testdb

spring.h2.console.enabled=true

spring.jpa.show-sql=true

#3. Model class

package in.nareshit.raghu.model;

import javax.persistence.Entity;

import javax.persistence.Id;

import lombok.Data;

//ctrl+shift+O

@Data

@Entity

public class Employee {

@Id

private Integer empId;

private String empName;

private Double empSal;

}

#4 Define one Repository Interface that is used to perform insert/update/delete/select (No Code

by programmer)

package in.nareshit.raghu.repo;

import org.springframework.data.repository.CrudRepository;

import in.nareshit.raghu.model.Employee;

public interface EmployeeRepository

extends CrudRepository<Employee, Integer>{

}

**\*)Note:** For above interface one impl class is generated using Sun Dynamic Proxy Concept and Linked with its Reference variable.

#5. Runner class

package in.nareshit.raghu.runner;

import org.springframework.beans.factory.annotation.Autowired;

import org.springframework.boot.CommandLineRunner;

import org.springframework.stereotype.Component;

import in.nareshit.raghu.model.Employee;

import in.nareshit.raghu.repo.EmployeeRepository;

@Component

public class EmployeeTestRunner implements CommandLineRunner {

@Autowired

private EmployeeRepository repo;

@Override

public void run(String... args) throws Exception {

// to find Impl className using Reference variable

System.out.println(repo.getClass().getName());

//--- insert data------

repo.save(new Employee(101,"AA",2500.0));

repo.save(new Employee(102,"BB",3500.0));

repo.save(new Employee(103,"CC",4800.0));

//---print all rows--------

Iterable<Employee> data = repo.findAll();

//JDK 1.5 - forEach loop

for(Employee e:data) {

System.out.println(e);

}

}

}

-----------------------------------------------------------------------

\*) To open any pre-defined file : ctrl+shift+T

\*) To view all members inside class/interface : ctrl+O

**Q) What is a Proxy class generator in java?**

A) Based on Interface and Design class, one class is generated at runtime that works for multiple inputs.

https://docs.oracle.com/javase/8/docs/technotes/guides/reflection/proxy.html

interface Sample {

void m1();

}

interface A extends Sample {};

class CodeGen {

generate(){

A a = proxy(A);

setBody(a.m1()=>{ sysout("HELLO"); }

}

}

\*)Dynamic Proxy Example : https://github.com/javabyraghu/DynamicProxyExample