

DAMOS A FINISH

* PASO 2

Rellenamos el archive application. Properties con estos datos

server.servlet.context-path=/ejemplopersona

spring.jpa.database=MYSQL

spring.jpa.show-sql=false

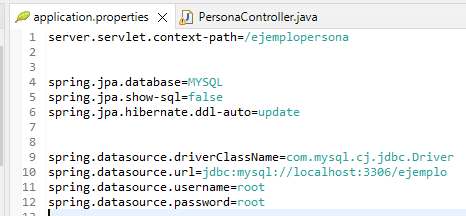
spring.jpa.hibernate.ddl-auto=update

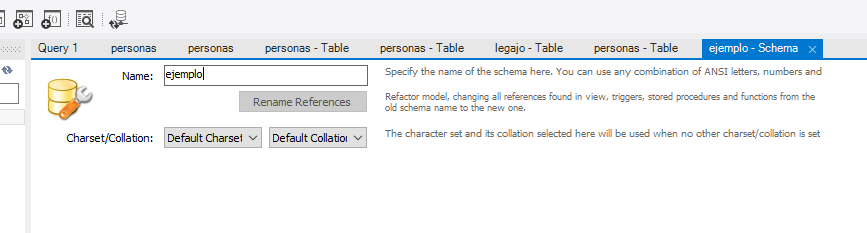
spring.datasource.driverClassName=com.mysql.cj.jdbc.Driver

spring.datasource.url=jdbc:mysql://localhost:3306/ejemplo

spring.datasource.username=root

spring.datasource.password=root



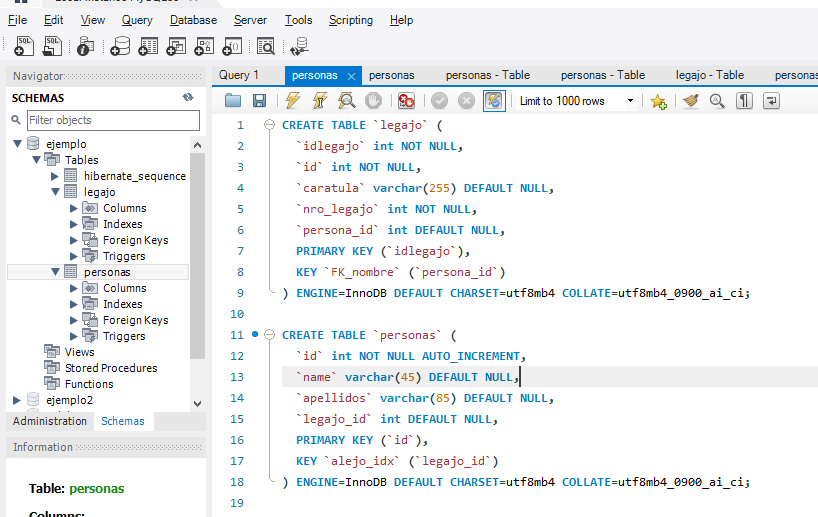
* REVISAR AQUÍ EL ONMBRE DE USUARIO Y PASSWORD Y NOMBRE DE BASE DE DATOS UTILIZADA
* PASO 3: CREAR BASE DE DATOS LLAMADA ejemplo
* 
* Y luego correr los siguientes scripts sql

CREATE TABLE `legajo` (  
  `idlegajo` int NOT NULL,  
  `id` int NOT NULL,  
  `caratula` varchar(255) DEFAULT NULL,  
  `nro\_legajo` int NOT NULL,  
  `persona\_id` int DEFAULT NULL,  
  PRIMARY KEY (`idlegajo`),  
  KEY `FK\_nombre` (`persona\_id`)  
) ENGINE=InnoDB DEFAULT CHARSET=utf8mb4 COLLATE=utf8mb4\_0900\_ai\_ci;

CREATE TABLE `personas` (  
  `id` int NOT NULL AUTO\_INCREMENT,  
  `name` varchar(45) DEFAULT NULL,  
  `apellidos` varchar(85) DEFAULT NULL,  
  `legajo\_id` int DEFAULT NULL,  
  PRIMARY KEY (`id`),  
  KEY `alejo\_idx` (`legajo\_id`),  
  CONSTRAINT `alejo` FOREIGN KEY (`legajo\_id`) REFERENCES `legajo` (`idlegajo`)  
) ENGINE=InnoDB AUTO\_INCREMENT=2 DEFAULT CHARSET=utf8mb4 COLLATE=utf8mb4\_0900\_ai\_ci;

ALTER TABLE legajo

ADD CONSTRAINT `FK\_nombre` FOREIGN KEY (`persona\_id`) REFERENCES `personas` (`id`);



PASO 5: agregar y/o crear las siguientes clases

PersonaController.java

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

**import** org.springframework.stereotype.Controller;

**import** org.springframework.web.bind.annotation.CrossOrigin;

**import** org.springframework.web.bind.annotation.RequestMapping;

**import** org.springframework.web.bind.annotation.RequestMethod;

**import** org.springframework.web.bind.annotation.ResponseBody;

**import** org.springframework.web.bind.annotation.RestController;

@RestController

@CrossOrigin(origins ="\*")

**public** **class** PersonaController {

@Autowired

**private** PersonaService personaService;

@RequestMapping(value = "/personas", method = RequestMethod.***GET***)

@ResponseBody

**public** Iterable<Persona> getPersonas() {

**return** personaService.findAll();

}

}

Persona.java

**import** javax.persistence.Column;

//JPA

**import** javax.persistence.Entity;

**import** javax.persistence.GeneratedValue;

**import** javax.persistence.GenerationType;

**import** javax.persistence.Id;

**import** javax.persistence.OneToOne;

**import** javax.persistence.Table;

@Entity

@Table(name = "personas")

**public** **class** Persona {

@Id

@GeneratedValue(strategy = GenerationType.***IDENTITY***)

**int** id;

@Column(name = "name")

String nombre;

String apellidos;

@OneToOne

Legajo legajo;

**public** **int** getId() {

**return** id;

}

**public** **void** setId(**int** id) {

**this**.id = id;

}

**public** String getNombre() {

**return** nombre;

}

**public** **void** setNombre(String nombre) {

**this**.nombre = nombre;

}

**public** String getApellidos() {

**return** apellidos;

}

**public** **void** setApellidos(String apellidos) {

**this**.apellidos = apellidos;

}

**public** Legajo getLegajo() {

**return** legajo;

}

**public** **void** setLegajo(Legajo legajo) {

**this**.legajo = legajo;

}

}

Legajo.java

**import** javax.persistence.Entity;

**import** javax.persistence.GeneratedValue;

**import** javax.persistence.Id;

**import** javax.persistence.OneToOne;

**import** javax.persistence.Table;

@Entity

@Table

**public** **class** Legajo {

@Id

@GeneratedValue

**int** id;

**protected** **int** nroLegajo;

**protected** String caratula;

@OneToOne

Persona persona;

**public** **int** getNroLegajo() {

**return** nroLegajo;

}

**public** **void** setNroLegajo(**int** nroLegajo) {

**this**.nroLegajo = nroLegajo;

}

**public** String getCaratula() {

**return** caratula;

}

**public** **void** setCaratula(String caratula) {

**this**.caratula = caratula;

}

}

Interface PersonaRepository.java

**import** org.springframework.data.repository.CrudRepository;

**import** org.springframework.stereotype.Repository;

@Repository

**public** **interface** PersonaRepositorio **extends** CrudRepository<Persona, Integer> {

}

PersonaService.java

**import** java.util.Optional;

**public** **interface** PersonaService {

<S **extends** Persona> S save(S entity);

<S **extends** Persona> Iterable<S> saveAll(Iterable<S> entities);

Optional<Persona> findById(Integer id);

**boolean** existsById(Integer id);

Iterable<Persona> findAll();

Iterable<Persona> findAllById(Iterable<Integer> ids);

**long** count();

**void** deleteById(Integer id);

**void** delete(Persona entity);

**void** deleteAllById(Iterable<? **extends** Integer> ids);

**void** deleteAll(Iterable<? **extends** Persona> entities);

**void** deleteAll();

}

PersonaServiceImpl.java

**import** java.util.Optional;

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

**import** org.springframework.stereotype.Service;

@Service

**public** **class** PersonaServiceImpl **implements** PersonaService {

@Autowired

PersonaRepositorio personaRepositorio;

@Override

**public** <S **extends** Persona> S save(S entity) {

**return** personaRepositorio.save(entity);

}

@Override

**public** <S **extends** Persona> Iterable<S> saveAll(Iterable<S> entities) {

**return** personaRepositorio.saveAll(entities);

}

@Override

**public** Optional<Persona> findById(Integer id) {

**return** personaRepositorio.findById(id);

}

@Override

**public** **boolean** existsById(Integer id) {

**return** personaRepositorio.existsById(id);

}

@Override

**public** Iterable<Persona> findAll() {

**return** personaRepositorio.findAll();

}

@Override

**public** Iterable<Persona> findAllById(Iterable<Integer> ids) {

**return** personaRepositorio.findAllById(ids);

}

@Override

**public** **long** count() {

**return** personaRepositorio.count();

}

@Override

**public** **void** deleteById(Integer id) {

personaRepositorio.deleteById(id);

}

@Override

**public** **void** delete(Persona entity) {

personaRepositorio.delete(entity);

}

@Override

**public** **void** deleteAllById(Iterable<? **extends** Integer> ids) {

personaRepositorio.deleteAllById(ids);

}

@Override

**public** **void** deleteAll(Iterable<? **extends** Persona> entities) {

personaRepositorio.deleteAll(entities);

}

@Override

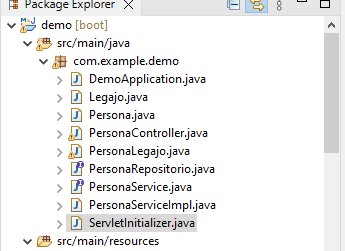
**public** **void** deleteAll() {

personaRepositorio.deleteAll();

}

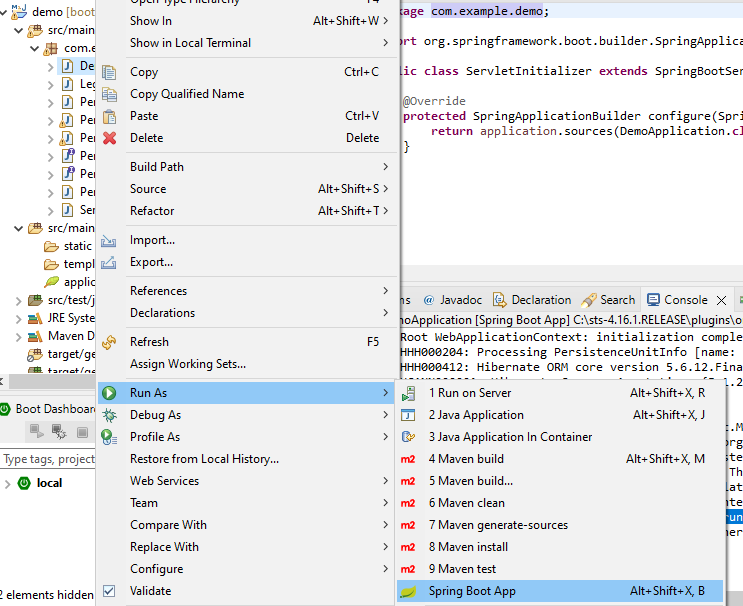
}

DEBERIAN TENER ALGO ASI

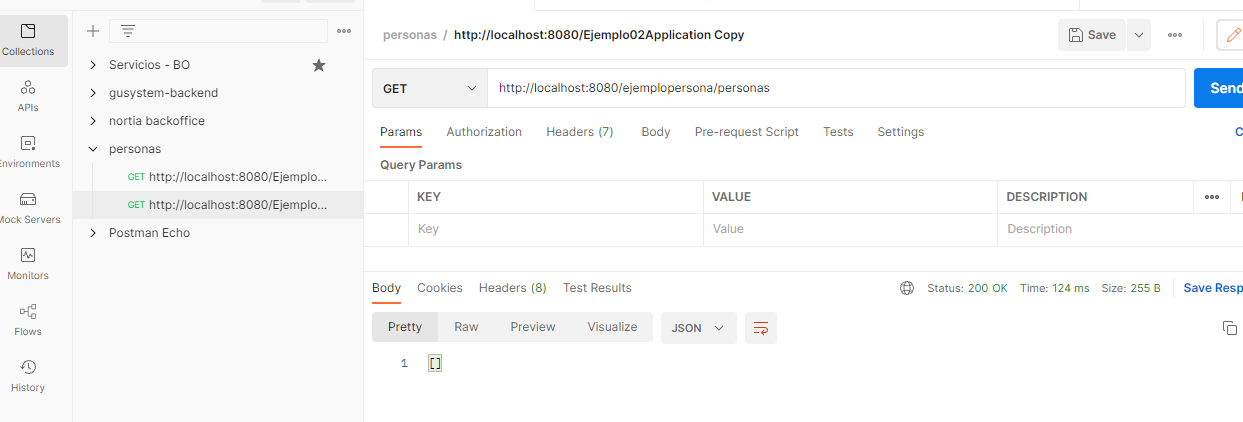


PASO 5- CORRER EL PROYECTO

EjemploApplication.java click derecho -> Run As -> Spring Boot App



* PASO 6: prueba en Postman



Enunciado para sprint boot  
  
-   SOBRE EL PROYECTO CREADO,  
  AGREGAR 2 NUEVOS ENDPOINTS:  
      - POST  que permita persistir una nueva persona  
      - DELETE que permita borrar una persona por su ID  
  
PARA USAR POSTMAN  
<https://www.youtube.com/watch?v=qsejysrhJiU&ab_channel=DominiCode>