

AD-4. Tarea en equipo. Microservicios web con Spring Boot



GRUPO 12

Jose San Pastor

David Menéndez

Raúl Moreno

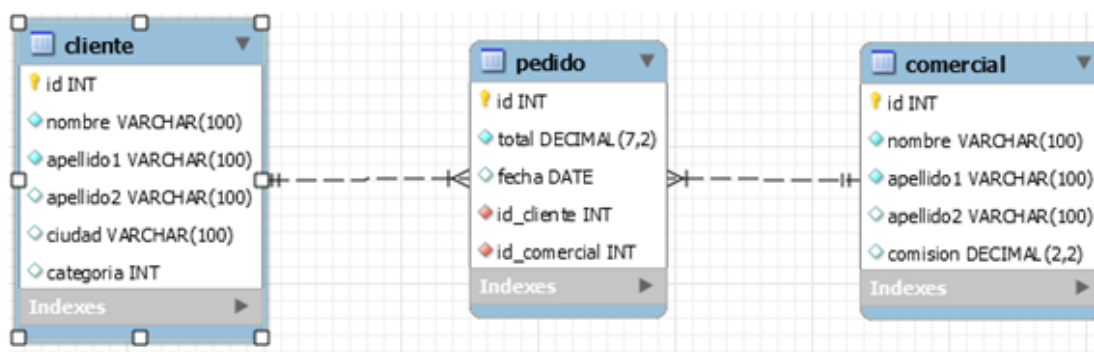
Github: <https://github.com/Davasda/AD-4-GRUPO>

ENUNCIADO.....	2
EJERCICIO A REALIZAR	3
RESULTADOS	4
1.Crear @RequestMapping("/comercial")	4
2...../alta para Dar de alta el comercial	5
3./eliminar Eliminar de la bbdd el comercial cuyo id coincida.	7
4./uno/{id} Devolver los datos del comercial cuyo id coincida	9
5./bycliente/{id} Devolver la lista de los comerciales que han atendido pedidos del cliente que coincida con ese id.	11
6./conpedidos Devolver la lista de los comerciales que han atendido algún pedido.....	13
7./pedidos/{id} Devolver la lista de pedidos gestionados por el comercial que coincida con ese id.	15

Una Empresa tiene recogido en una base de datos los pedidos solicitados por Clientes y el Comercial asociado a la venta. La tabla pedidos recoge el importe total del pedido realizado.

La aplicación la van a usar los jefes de comerciales para ver la información, tanto de clientes como de sus comerciales.

Tablas:



Se adjunta el script del sql (en la unidad formativa UF6) con la creación de la base de datos (ventasbdd_2023) de las tablas, insert para pruebas y creación del usuario uventas_2023 con la password uventas.

EJERCICIO A REALIZAR

3

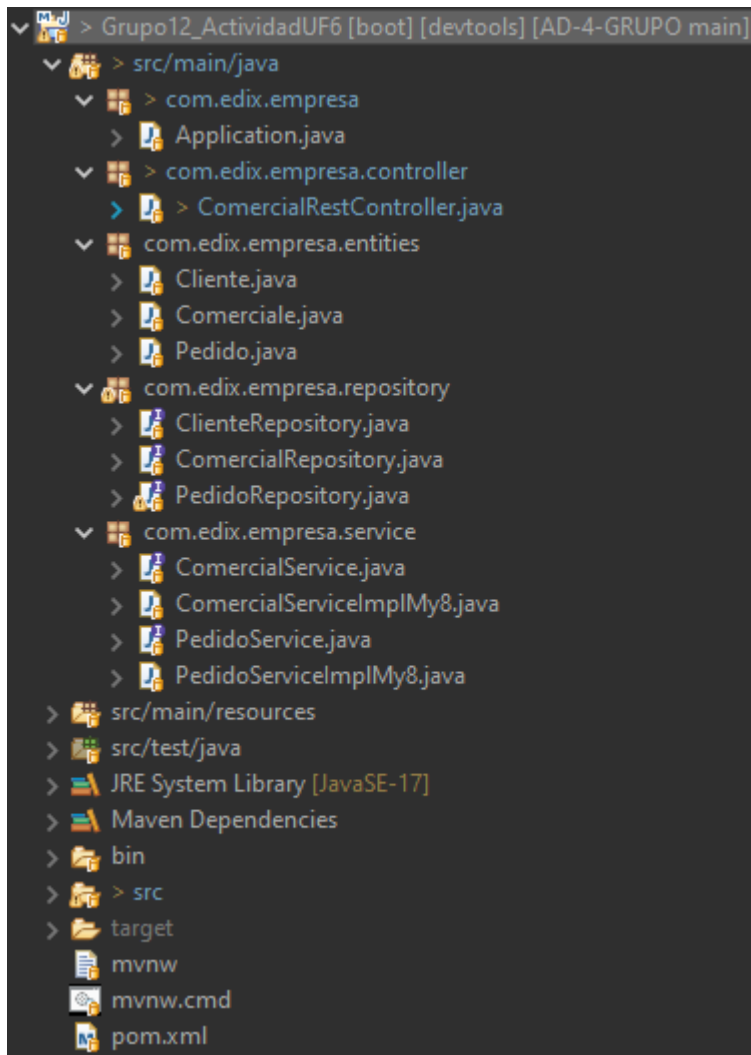
Crear un proyecto Spring web, con Spring data Jpa y MySQL 8 para los siguientes servicios web restfull, y probar con postman:

url	Descripción
/comercial	@RequestMapping("/comercial")
/alta	Dar de alta el comercial
/eliminar/{id}	Eliminar de la bbdd el comercial cuyo id coincida
/uno/{id}	Devolver los datos del comercial cuyo id coincida
/bycliente/{id}	Devolver la lista de los comerciales que han atendido pedidos del cliente que coincida con ese id.
/conpedidos	Devolver la lista de comerciales que han atendido algún pedido
/pedidos/{id}	Devolver la lista de pedidos gestionados por el comercial que coincida con ese id.

RESULTADOS

I. Crear @RequestMapping("/comercial")

Estructura del proyecto:



La aplicación se ejecuta en puerto 8088: <http://localhost:8088>

Se crea dentro de ComercialRestController.java el correspondiente RequestMapping

```
@RestController
@RequestMapping("/comercial")
public class ComercialRestController {
```

2. /alta para Dar de alta el comercial

2.1. Código realizado

2.1.1. En el Interface ComercialService (no volvemos a poner este código en este pdf ya que se usa para todos los métodos de comercial)

```
public interface ComercialService {
    Comerciale altaComercial(Comerciale comercial);
    Comerciale buscarUno(int idComercial);
    List<Comerciale> buscarTodos();
    boolean eliminarComercial(int idComercial);
    List<Comerciale> porCliente(int idCliente);
    List<Comerciale> comercialesConPedidos();
}
```

2.1.2. En la clase ComercialServiceImplMy8

```
@Autowired
private ComercialRepository cRepo;
/**
 * Damos de alta el comercial, buscamos el comercial y si no existe lo añadimos a la lista y si existe, damos valor null
 */
@Override
public Comerciale altaComercial(Comerciale comercial) {
    if (buscarUno(comercial.getIdComercial()) == null)
        return cRepo.save(comercial);
    return null;
}
```

2.1.3. En la clase controlador ComercialRestController

```
@RestController
@RequestMapping("/comercial")
public class ComercialRestController {
    @Autowired
    private ComercialService cServ;
    @Autowired
    private PedidoService pServ;

    /**
     * Dar de alta el comercial mediante la @Query en la clase Comercial Service
     * @param comercial
     * @return
     */
    @PostMapping("/alta")
    public Comerciale alta(@RequestBody Comerciale comercial) {
        return cServ.altaComercial(comercial);
    }
}
```

2.2. Resultado /alta

3. /eliminar Eliminar de la bbdd el comercial cuyo id coincida.

3.1. En la clase ComercialServiceImplMy8

```
/**
 * Buscamos un comercial y lo eliminamos mediante su id si existe
 */
@Override
public boolean eliminarComercial(int idComercial) {
    try{
        if (buscarUno(idComercial)!=null) {
            cRepo.deleteById(idComercial);
            return true;
        }catch (Exception e) {
            return false;
        }
    }
}
```

3.2. En la clase controlador ComercialRestController

```
/**
 * Mediante el Id de un comercial, eliminarlo de la Base de Datos
 * @param idComercial
 * @return Devuelve un String con el texto de si el comercial ha sido o no eliminado.
 */
@DeleteMapping("/eliminar/{id}")
public String eliminarUno(@PathVariable("id") int idComercial) {
    return (cServ.eliminarComercial(idComercial)) == true?"Comercial ELIMINADO":"Error eliminar el comercial";
}
```


3.3. Resultado /eliminar

Resultado postman

DELETE

⌵

http://localhost:8088/comercial/eliminar/15

Params

Auth

Headers (8)

Body ●

Pre-req.

Tests

Settin

Query Params

KEY	VALUE
Key	Value

body

⌵

Pretty

Raw

Preview

Visualize

Text ⌵

☰

1 Comercial ELIMINADO

Resultado Mysql

	id_comercial	nombre	apellido1	apellido2	comision
▶	1	Daniel	Sáez	Vega	0.15
	2	Juan	Gómez	López	0.13
	3	Diego	Flores	Salas	0.11
	4	Marta	Herrera	Gil	0.14
	5	Antonio	Carretero	Ortega	0.12
	6	Manuel	Domínguez	Hernández	0.13
	7	Antonio	Vega	Hernández	0.11
	8	Alfredo	Ruiz	Flores	0.05
✱	NULL	NULL	NULL	NULL	NULL

4. /uno/{id} Devolver los datos del comercial cuyo id coincida

4.1. En la clase ComercialServiceImplMy8

```
/**
 * Buscamos un comercial, mediante su id
 */
@Override
public Comerciale buscarUno(int idComercial) {
    // TODO Auto-generated method stub
    return cRepo.findById(idComercial).orElse(null);
}
```

4.2. En la clase controlador ComercialRestController

```
/**
 * Mediante el Id de un comercial, obtener sus datos
 * @param idComercial
 * @return
 */
@GetMapping("/uno/{id}")
public Comerciale buscarUno(@PathVariable("id") int idComercial) {
    return cServ.buscarUno(idComercial);
}
```

4.3. Resultado /uno/{id}

Resultado Postman

GET ⌵ http://localhost:8088/comercial/uno/1

Params Auth Headers (6) Body Pre-req. Tests ⌵

Params

Query Params

	KEY	VALUE
	Key	Value

Body ⌵

Pretty Raw Preview Visualize JSON ⌵

```
1  {
2    "idComercial": 1,
3    "apellido1": "Sáez",
4    "apellido2": "Vega",
5    "comision": 0.15,
6    "nombre": "Daniel"
7  }
```

5. /bycliente/{id} Devolver la lista de los comerciales que han atendido pedidos del cliente que coincida con ese id.

5.1. En la interface ComercialRespository hemos creado una Query con nativeQuery.

```
/**
 * Devolvemos la lista de los comerciales que han atendido pedidos del cliente que coincida con ese id.
 * @param idCliente
 * @return
 */
@Query("select distinct c from Comercial c, Pedido p where p.comercial.idComercial= c and p.cliente.idCliente =?1")
List<Comercial> porCliente(int idCliente);
```

5.2. En la clase ComercialServiceImplMy8

```
@Override
public List<Comercial> porCliente(int idCliente) {
    // TODO Auto-generated method stub
    return cRepo.porCliente(idCliente);
}
```

5.3. En la clase ComercialRestController

```
/**
 * Mediante el id de un cliente, sacar la lista de comerciales que han atendido algun pedido que haya hecho dicho cliente.
 * @param idCliente
 * @return
 */
@GetMapping("/bycliente/{id}")
public List<Comercial> byCliente(@PathVariable("id") int idCliente) {
    return cServ.porCliente(idCliente);
}
```

5.4. Resultado /bycliente/{id}

Resultado Postman

GET

Params Auth Headers (6) Body Pre-req. Tests Settings

Query Params

KEY	VALUE
Key	Value

Body

Pretty Raw Preview Visualize JSON

```
1 [
2   {
3     "idComercial": 3,
4     "apellido1": "Flores",
5     "apellido2": "Salas",
6     "comision": 0.11,
7     "nombre": "Diego"
8   },
9   {
10    "idComercial": 2,
11    "apellido1": "Gómez",
12    "apellido2": "López",
13    "comision": 0.13,
14    "nombre": "Juan"
15  }
16 ]
```

6. /conpedidos Devolver la lista de los comerciales que han atendido algún pedido

6.1. En la interface ComercialRespository hemos creado una Query con nativeQuery.

```
/**
 * Query desde la base de datos, en el cual sacamos la lista de comerciales que hayan realizado algun pedido
 * @return
 */
@Query(value = "select distinct * from comerciales CO, pedidos P where CO.id_comercial=P.id_comercial and P.id_comercial is not null group by CO.id_comercial",
      nativeQuery = true)
List<Comerciale> comercialesConPedidos();
```

6.2. En la clase ComercialServiceImplMy8

```
@Override
public List<Comerciale> comercialesConPedidos() {
    // TODO Auto-generated method stub
    return cRepo.comercialesConPedidos();
}
```

6.3. En la clase ComercialRestController

```
/**
 * Sacar la lista de comerciales que hayan atendido algun pedido
 * @return
 */
@GetMapping("/conpedidos")
public List<Comerciale> comercialPedidos() {
    return cServ.comercialesConPedidos();
}
```

6.4. Resultado /conpedidos

Resultado Postman

GET ⌵ http://localhost:8088/comercial/conpedidos

Params Auth Headers (6) Body Pre-req. Tests Settings

Query Params

KEY	VALUE
Key	Value

Body ⌵

Pretty Raw Preview Visualize JSON ⌵

```

1  [
2    {
3      "idComercial": 2,
4      "apellido1": "Gómez",
5      "apellido2": "López",
6      "comision": 0.13,
7      "nombre": "Juan"
8    },
9    {
10     "idComercial": 5,
11     "apellido1": "Carretero",
12     "apellido2": "Ortega",
13     "comision": 0.12,
14     "nombre": "Antonio"
15   },
16   {
17     "idComercial": 1,
```

Resultado Texto Json

```

[
  {
    "idComercial": 2,
    "apellido1": "Gómez",
    "apellido2": "López",
    "comision": 0.13,
    "nombre": "Juan"
  },
  {
    "idComercial": 5,
    "apellido1": "Carretero",
    "apellido2": "Ortega",
    "comision": 0.12,
    "nombre": "Antonio"
  },
  {
    "idComercial": 1,
    "apellido1": "Sáez",
    "apellido2": "Vega",
    "comision": 0.15,
    "nombre": "Daniel"
  },
  {
    "idComercial": 3,
    "apellido1": "Flores",
    "apellido2": "Salas",
    "comision": 0.11,
    "nombre": "Diego"
  },
  {
    "idComercial": 6,
    "apellido1": "Domínguez",
    "apellido2": "Hernández",
    "comision": 0.13,
    "nombre": "Manuel"
  },
  {
    "idComercial": 7,
    "apellido1": "Vega",
    "apellido2": "Hernández",
    "comision": 0.11,
    "nombre": "Antonio"
  }
]
```

7. /pedidos/{id} Devolver la lista de pedidos gestionados por el comercial que coincida con ese id.

Para este método teníamos dudas de si crear un Restcontroller de pedidos, ya que lo que realmente nos devuelve es una lista de pedidos y no una lista de comerciales ,por lo que que requestmapping de comerciales quizá no sería el sitio.

Como la actividad tiene claramente el requestmapping de comercial lo hemos dejado ahí aunque si hemos creado los servicios y repositorios de pedidos para que se haga la consulta y métodos y luego sea el controlador del comercial quien lo llame.

7.1. En en interface PedidoRepository hemos creado una Query con nativeQuery.

```
/**
 * Devolvemos la lista de pedidos gestionados por el comercial que coincida con ese id.
 * @param idComercial
 * @return
 */
@Query(value = "select* from pedidos where id_comercial = ?", nativeQuery = true)
List<Pedido> pedidosPorComercial(int idComercial);
```

7.2. En el interface PedidoService

```
public interface PedidoService {

    List<Pedido> pedidosPorComercial (int idComercial);
    List <Pedido> buscarTodosPedidos ();
}
```

7.3. En la clase PedidoServiceImplMy8

```
@Service
public class PedidoServiceImplMy8 implements PedidoService {
    @Autowired
    private PedidoRepository pRepo;

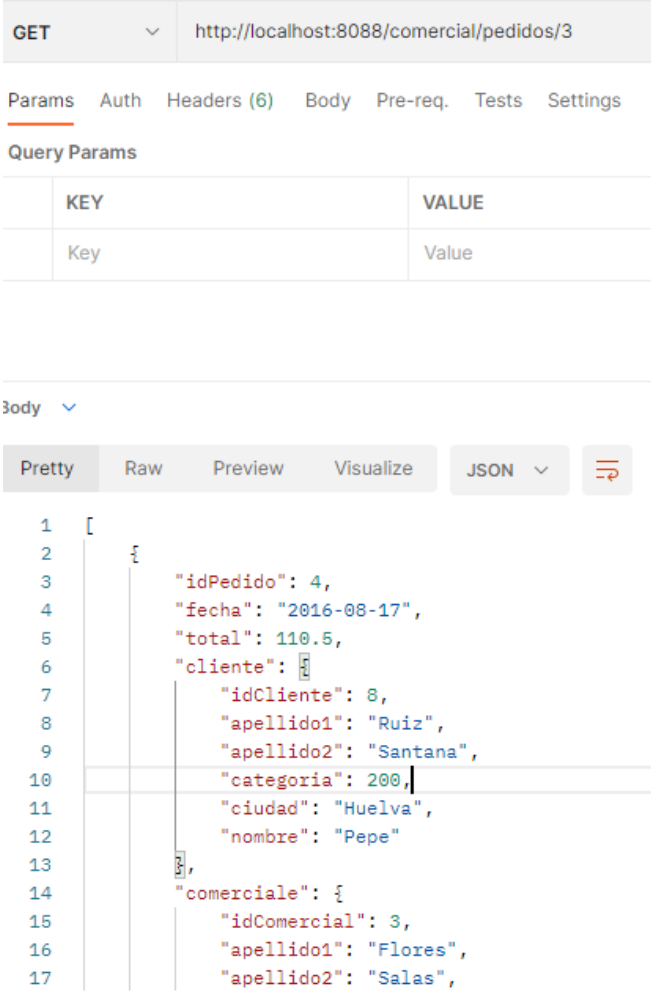
    @Override
    public List<Pedido> pedidosPorComercial(int idComercial) {
        return pRepo.pedidosPorComercial(idComercial);
    }
}
```


7.4. En la clase ComercialRestController

```
/**
 * Devolver todos los pedidos que haya atendido el comercial con ese id
 * @param idComercial
 * @return
 */
@GetMapping("/pedidos/{id}")
public List<Pedido> porPedidos(@PathVariable("id") int idComercial) {
    return pServ.pedidosPorComercial(idComercial);
}
```

7.5. Resultado /pedidos/{id}

Resultado Postman



GET ▼ http://localhost:8088/comercial/pedidos/3

Params Auth Headers (6) Body Pre-req. Tests Settings

Query Params

KEY	VALUE
Key	Value

Body ▼

Pretty Raw Preview Visualize JSON ▼ ≡

```

1  [
2    {
3      "idPedido": 4,
4      "fecha": "2016-08-17",
5      "total": 110.5,
6      "cliente": {
7        "idCliente": 8,
8        "apellido1": "Ruiz",
9        "apellido2": "Santana",
10       "categoria": 200,
11       "ciudad": "Huelva",
12       "nombre": "Pepe"
13     },
14     "comerciale": {
15       "idComercial": 3,
16       "apellido1": "Flores",
17       "apellido2": "Salas",

```

Resultado Texto Json

```

[
  {
    "idPedido": 4,
    "fecha": "2016-08-17",
    "total": 110.5,
    "cliente": {
      "idCliente": 8,
      "apellido1": "Ruiz",
      "apellido2": "Santana",
      "categoria": 200,
      "ciudad": "Huelva",
      "nombre": "Pepe"
    },
    "comerciale": {
      "idComercial": 3,
      "apellido1": "Flores",
      "apellido2": "Salas",
      "comision": 0.11,
      "nombre": "Diego"
    }
  },
  {
    "idPedido": 9,
    "fecha": "2016-10-10",
    "total": 2480.4,
    "cliente": {
      "idCliente": 8,
      "apellido1": "Ruiz",
      "apellido2": "Santana",
      "categoria": 200,
      "ciudad": "Huelva",
      "nombre": "Pepe"
    },
    "comerciale": {
      "idComercial": 3,
      "apellido1": "Flores",
      "apellido2": "Salas",
      "comision": 0.11,
      "nombre": "Diego"
    }
  }
]

```