1. CLASE INTERFAZ

En la **interfaz de Reservation** para acceder al recurso que se necesita.

**EJEMPLOS**

/\*SELECT \* FROM Reservation WHERE id=10

public List<Reservation> findAllByIdReservation(int id){ }

/\*SELECT \* FROM Reservation WHERE status=created AND id = 10;

public List<Reservation> findAllByStatusAndIdReservation(String status, int id){ }

**Reporte 1 – Reservas en un periodo de fechas**

/\* SELECT \* FROM Reservation WHERE startDate AFTER fechaUsuarioStart AND devolutionDate BEFORE fechaUsuarioFin \*/

Public List <Reservation> findAllByStartDateAfterAndDevolutionDateBefore(Date fechaUsuarioStart, Date fechaUsuarioFin){

} /\*Solo se incluye la firma del método a utilizar por ser interfaz

**Count para saber el número total**

/\*SELECT COUNT(\*) FROM Reservation WHERE status=”completada” OR status=”canceladas”

**Reporte 2 – contar según estado.**

/\*SELECT \* FROM Reservation WHERE status=”lo que necesite” /\*lo que necesite se reemplaza

por “completed” y “canceled”)

public List<Reservation> findAllByStatus(String status){

}

**Reporte 3**

SELECT COUNT(client), clientFROM Reservation GROUP BY client ORDER BY COUNT(client) DESC;

[total1, Cliente1]

[total2, Cliente2]

[total3, Cliente3]

La clase que se obtiene se pide List<Object[]> (arreglo de objetos) para traer la información sin importar que tipo sea.

@Query(“SELECT COUNT(c.client), c.clientFROM Reservation AS c GROUP BY client ORDER BY COUNT(c.client) DESC;”) /\*debe ser la de JPA, me permite hacer consultas SQL\*/

public List<Object[]> getTotalReservationByClient();

1. **Clase repositorio**

**Reporte 1**

public List<Reservation> getReservationsBetweenDates(Date fechaUsuarioStart, Date fechaUsuarioFin ){

return reservationInterfaz.findAllByStartDateAfterAndDevolutionDateBefore(Date fechaUsuarioStart, Date fechaUsuarioFin)

}

**Reporte 2**

public List<Reservation> getResevationsByStatus(String status){

return reservationInterfaz. findAllByStatus(status);

}

**Reporte 3 – Necesita implementar un DTO (Data Transfer Object) un objeto cuya única función es la de transmitir datos. (no es entidad, tabla, controlador, etc)**

**Este DTO se crea en Modelo: En el paquete Modelo creamos un nuevo paquete llamado DTO y adentro creamos una nueva clase llamada TotalAndClient**

Public Class TotalAndClient{

Private Long total;

Private Client client;

# CREAR EL MÉTODO COSNTRUCTOR (Clic derecho insert code constructor)

# CREAR GETTERS Y SETTERS (Clic derecho, insert code, Gettes and Setter, seleccionar los atributos)

}

EN el repositorio creamos el llamado para reporte 3

Public List<TotalAndClient> getTopClients{

List<TotalAndClient> respuesta =new ArrayList<>();

List<Object[]> reporte = reservationInterfaz.getTotalReservationsByClient();

for(int i=0; i<reporte.size(); i++){

respuesta.add(new TotalAndClient( (Long) reporte.get(i)[1], (Client) reporte.get(i)[0]));

}

return reserva;

1. **Servicios – Se implmentan las validaciones**

**Reporte 1**

Public List<Reservation> getReservationsBetweenDatesReport(String fechaA, String fechaB){

SimpleDateFormat parser = new SimpleDateFormat( pattern: “yyyy-MM-dd”);

Date a = new Date();

Date b = new Date();

try{

a = parser.parse(fechaA);

b = parser.parse(fechaB);

}catch (ParseException exception){

Exception.printStackTrace();

}

If(a.before(b)){

return reservationRepository.getReservationsBetweenDates (a, b);

}else{

return new ArrayList<>();

}

}

**Reporte 2**

/\*Debo armar el DTO para enviar el JSON formateado

Public class CompletedAndCancelled{

Private Long completed;

Private Long canceled;

/\* CREAR CONSTRUCTOR

/\*CREAR GETTER AND SETTER

}

Public CompletedAndCanceled getReservationStatusReport(){

List<Reservation> completed = reservationRepository.getReservationsByStatus(“completed”);

List<Reservation> canceled = reservationRepository.getReservationsByStatus(“canceled”);

Int cantidadCompleted = completed.size();

Int cantidadCanceled = canceled.size();

return new CompletedAndCancelled((long) cantidadCompleted, (long) cantidadCanceled);

}

**Reporte 3**

Public List<TotalAndClient> getTopClientsReport(){

return reservationRepository.getTopClients();

}

1. **Controladores (Manejadores) @PathVariable toma el dato de la ruta.**

**@GetMappin(“/report-dates/{fecha1}/{fecha2}”)**

**Public List<Reservation> getReservationsBetweenDatesReports(@PathVariable(“fecha1”) String fecha1, (@PathVariable(“fecha2”) String fecha2){**

**Retorna el servicio**

**}**

**@GetMapping(/report-status”)**

**Public CompletedAndCancelled getReservationStatusReport(){**

**return reservationService.getReservation**

**}**

**@GetMapping(/report-clients”)**

**Public METODO TOTALCLIENTES{**

**Return SERVICIO}**