



API HIBERNATE

Endpoints Relacionales



29 DE FEBRERO DE 2024
JOSE MIGUEL GARCÍA NAVARRO
2ºDAM

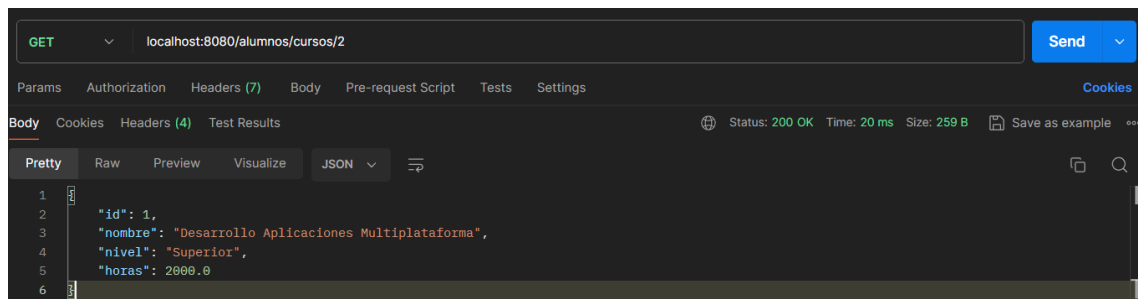
Contenido

1.	Obtener el curso que estudia un alumno:	2
2.	Obtener los alumnos que estudian un curso:	3
3.	Asignar un curso a un alumno:	4
4.	Obtener profesores que dan clase a un alumno:	5
5.	Obtener alumnos que son enseñados por un profesor:	6
6.	Asignar un profesor a un alumno:	7
7.	Obtener las asignaturas de un curso:	8
8.	Obtener los cursos a los que pertenece una asignatura:	9
9.	Asignar una asignatura a un curso:	10

1. Obtener el curso que estudia un alumno:

Este endpoint muestra el curso que estudia el alumno 2 el cual es Ana Martínez, es la alumna con id = 2 que está estudiando el curso con id = 1 que es Desarrollos de Aplicaciones Multiplataforma

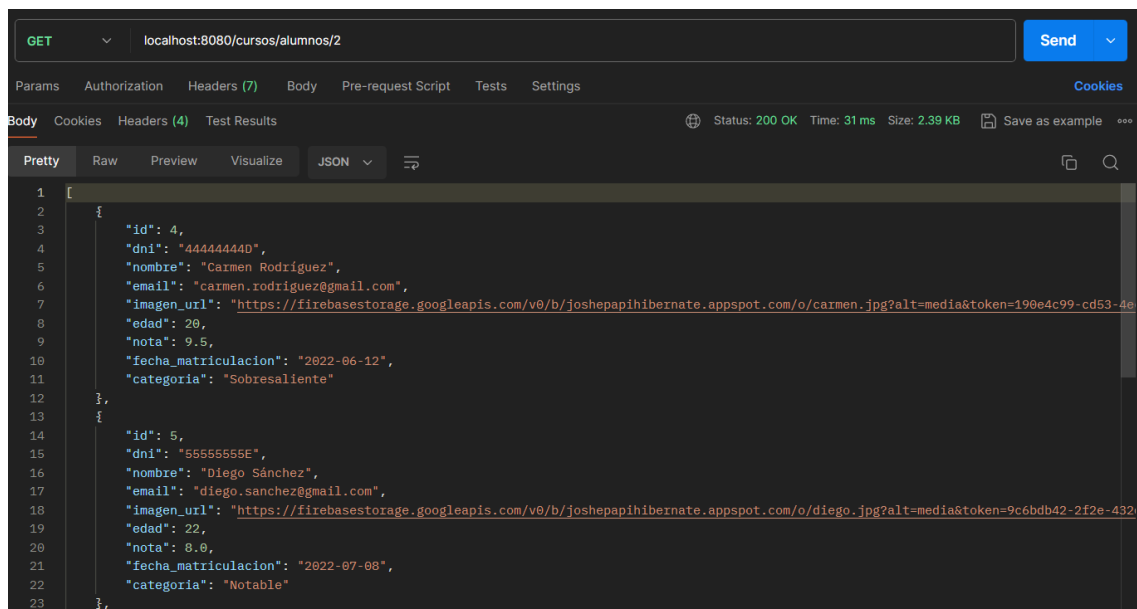
```
// Obtener el curso de un alumno
Spark.get( path: "/alumnos/cursos/:idal", (request, response) -> {
    Long idal = Long.parseLong(request.params(":idal"));
    Alumno a = dao.buscarById(idal);
    Curso c = dao_asoc.obtenercursoAlumno(a);
    response.type( contentType: "application/json");
    if (c!=null) {
        return gson.toJson(c);
    }
    else{
        return "No existe ese alumno";
    }
});
```



2. Obtener los alumnos que estudian un curso:

Este endpoint muestra una lista de alumnos que estudian el curso con el id indicado

```
// Obtener los alumnos de un curso
Spark.get( path: "/cursos/alumnos/:idcurs", (request, response) -> {
    Long id = Long.parseLong(request.params(":idcurs"));
    Curso p= dao_curs.buscarPorId(id);
    List<Alumno> a = dao_asoc.obtenerAlumnosCurso(p);
    response.type( contentType: "application/json");
    if (a!=null) {
        return gson.toJson(a);
    }
    else{
        return "No existe ese curso";
    }
});
```



The screenshot shows a REST client interface with the following details:

- Method:** GET
- URL:** localhost:8080/cursos/alumnos/2
- Status:** 200 OK
- Time:** 31 ms
- Size:** 2.39 KB
- Response Format:** JSON
- Response Body:** A JSON array containing two student objects.

```
[{"id": 4, "dni": "44444444D", "nombre": "Carmen Rodriguez", "email": "carmen.rodriguez@gmail.com", "imagen_url": "https://firebasestorage.googleapis.com/v0/b/joshepapihibernate.appspot.com/o/carmen.jpg?alt=media&token=190e4c99-cd53-4e", "edad": 20, "nota": 9.5, "fecha_matriculacion": "2022-06-12", "categoria": "Sobresaliente"}, {"id": 5, "dni": "55555555E", "nombre": "Diego S\u00e1nchez", "email": "diego.sanchez@gmail.com", "imagen_url": "https://firebasestorage.googleapis.com/v0/b/joshepapihibernate.appspot.com/o/diego.jpg?alt=media&token=9c6bdb42-2f2e-432", "edad": 22, "nota": 8.0, "fecha_matriculacion": "2022-07-08", "categoria": "Notable"}]
```

3. Asignar un curso a un alumno:

Este endpoint asocia al alumno (que ha sido creado con el endpoint de crear alumno simple `localhost:8080/alumnos/crear` y que por lo tanto aparece en la base de datos con la foreign key "curso_key" como nula) el curso.

```
// Asignar curso a alumno
Spark.post(path: "/cursos/alumnos/:idcurs/:idal", (request, response) -> {
    Long idal = Long.parseLong(request.params(":idal"));
    Long idcurs = Long.parseLong(request.params(":idcurs"));
    Alumno a = dao.buscarById(idal);
    Curso c = dao_curs.buscarPorId(idcurs);
    response.type(contentType: "application/json");
    return gson.toJson(dao_asoc.asignarCurso(a,c));
});
```

Después de crear

	id	categoria	dni	edad	email	fecha_matriculacion	imagen_url	nombre	nota	curso_key
	22	Suspense	751525260	42	antua.contreras@gmail.com	2021-10-02	https://firebasestorage.googleapis.com/v0/b/joshep...	Antua Contreras	3.2	NULL

Proceso de asignación

POST localhost:8080/cursos/alumnos/4/22

Status: 200 OK Time: 31 ms Size: 153 B

Body: true

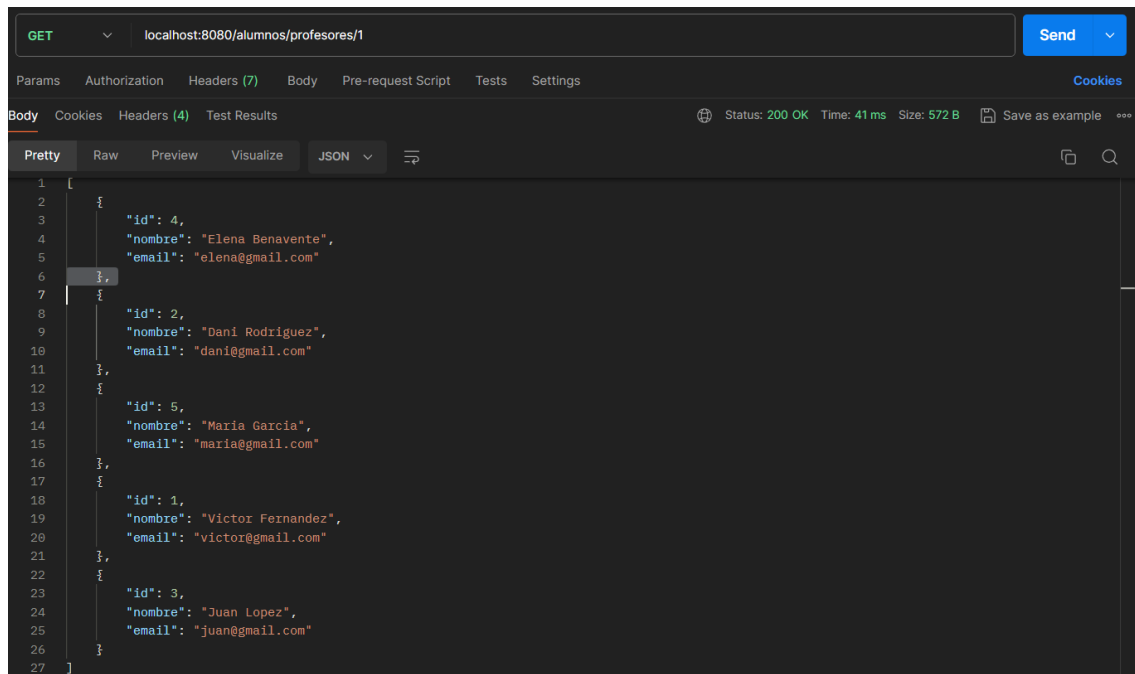
Después de asignar

	id	categoria	dni	edad	email	fecha_matriculacion	imagen_url	nombre	nota	curso_key
	22	Suspense	751525260	42	antua.contreras@gmail.com	2021-10-02	https://firebasestorage.googleapis.com/v0/b/joshep...	Antua Contreras	3.2	4

4. Obtener profesores que dan clase a un alumno:

Muestra los profesores que enseñan al alumno con la id indicada en la URL

```
// Obtener profesores de un alumno
Spark.get(path: "/alumnos/profesores/:idal", (request, response) -> {
    Long idal = Long.parseLong(request.params(":idal"));
    Alumno a = dao.buscarById(idal);
    List<Profesor> p = dao_asoc.alumnosConProfesor(a);
    response.type(contentType: "application/json");
    if (p!=null) {
        return gson.toJson(p);
    }
    else{
        return "No existe ese alumno";
    }
});
```



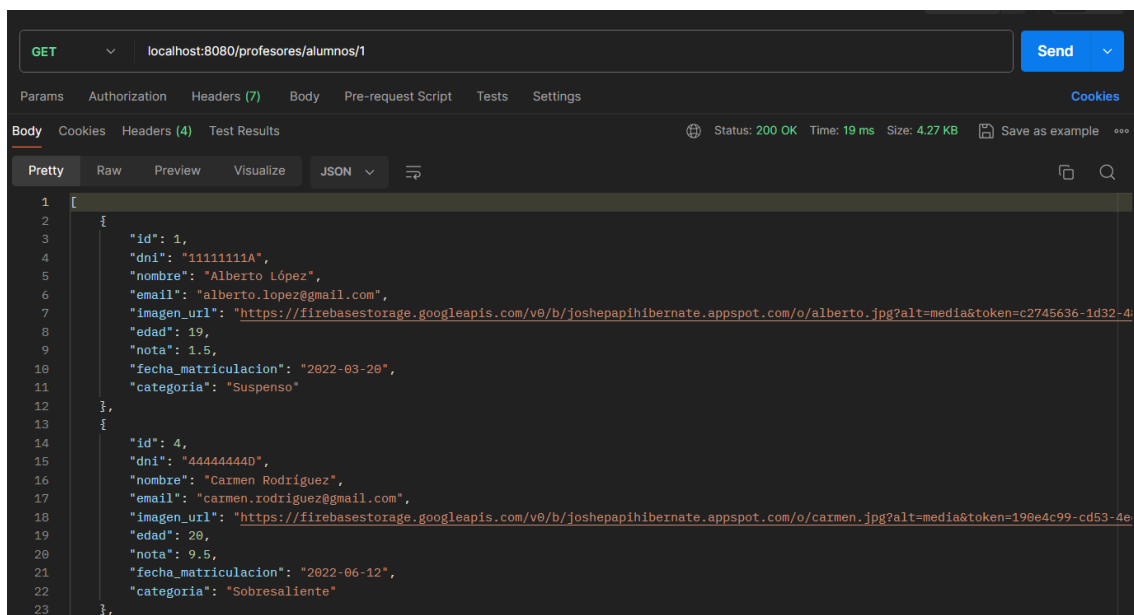
The screenshot shows a REST client interface with a GET request to `localhost:8080/alumnos/profesores/1`. The response is a JSON array of five teachers, each with an id, name, and email.

```
[
  {
    "id": 4,
    "nombre": "Elena Benavente",
    "email": "elena@gmail.com"
  },
  {
    "id": 2,
    "nombre": "Dani Rodriguez",
    "email": "dani@gmail.com"
  },
  {
    "id": 5,
    "nombre": "Maria Garcia",
    "email": "maria@gmail.com"
  },
  {
    "id": 1,
    "nombre": "Victor Fernandez",
    "email": "victor@gmail.com"
  },
  {
    "id": 3,
    "nombre": "Juan Lopez",
    "email": "juan@gmail.com"
  }
]
```

5. Obtener alumnos que son enseñados por un profesor:

Muestra los alumnos asociados al profesor con la id indicada en la URL

```
// Obtener alumnos de un profesor
Spark.get( path: "/profesores/alumnos/:idprof", (request, response) -> {
    Long id = Long.parseLong(request.params(":idprof"));
    Profesor p = dao_prof.buscarPorId(id);
    List<Alumno> a = dao_asoc.profesoresConAlumnos(p);
    response.type( contentType: "application/json");
    if (a!=null) {
        return gson.toJson(a);
    }
    else{
        return "No existe ese profesor";
    }
});
```



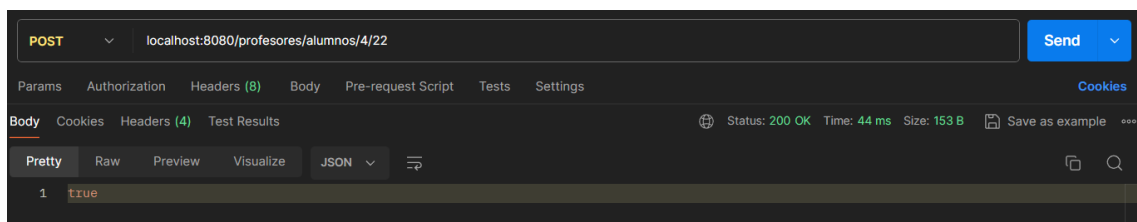
The screenshot shows a REST client interface with a GET request to `localhost:8080/profesores/alumnos/1`. The response is a JSON array of two student objects. The status is 200 OK, time is 19 ms, and size is 4.27 KB.

```
[
  {
    "id": 1,
    "dni": "11111111A",
    "nombre": "Alberto López",
    "email": "alberto.lopez@gmail.com",
    "imagen_url": "https://firebasestorage.googleapis.com/v0/b/joshepapihibernate.appspot.com/o/alberto.jpg?alt=media&token=c2745636-1d32-4e",
    "edad": 19,
    "nota": 1.5,
    "fecha_matriculacion": "2022-03-20",
    "categoria": "Suspendido"
  },
  {
    "id": 4,
    "dni": "44444444D",
    "nombre": "Carmen Rodríguez",
    "email": "carmen.rodriguez@gmail.com",
    "imagen_url": "https://firebasestorage.googleapis.com/v0/b/joshepapihibernate.appspot.com/o/carmen.jpg?alt=media&token=190e4c99-cd53-4e",
    "edad": 20,
    "nota": 9.5,
    "fecha_matriculacion": "2022-06-12",
    "categoria": "Sobresaliente"
  }
]
```

6. Asignar un profesor a un alumno:

Este endpoint asocia al alumno que ha sido creado con el endpoint de crear alumno simple `localhost:8080/alumnos/crear` y que no tiene relación aun con ningún profesor en la tabla relación “profesor_alumno”.

```
// Asignar profesor a alumno
Spark.post( path: "/profesores/alumnos/:idprof/:idal", (request, response) -> {
    Long idal = Long.parseLong(request.params(":idal"));
    Long idprof = Long.parseLong(request.params(":idprof"));
    Alumno a= dao.buscarById(idal);
    Profesor p= dao_prof.buscarPorId(idprof);
    response.type( contentType: "application/json");
    return gson.toJson(dao_asoc.asignarProfesor(a,p));
});
```

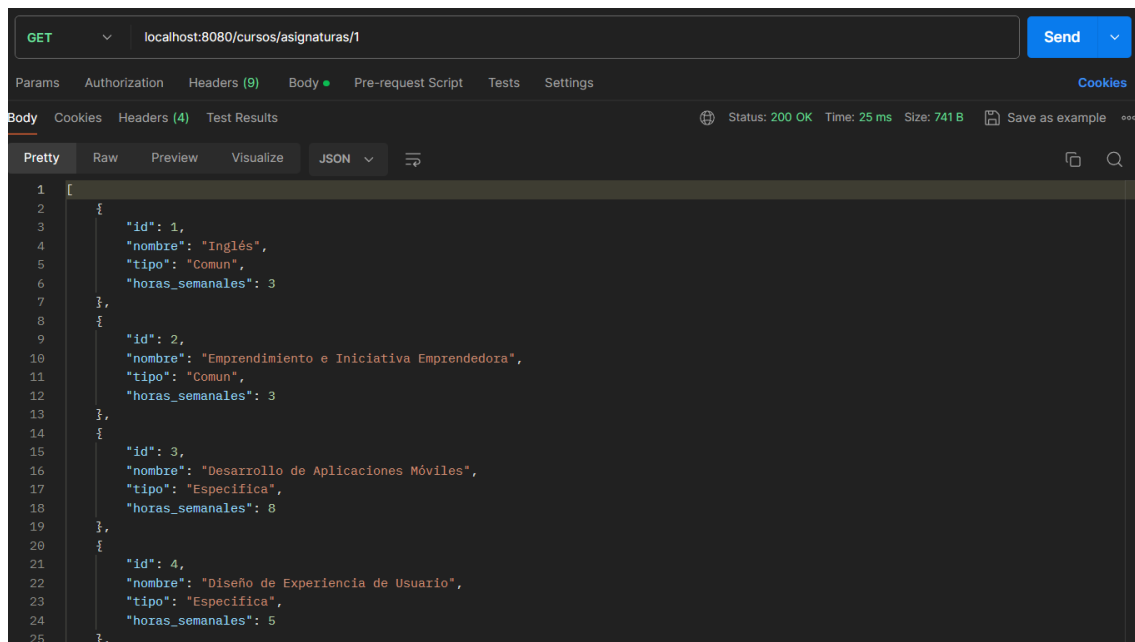


alumno_id	profesor_id
22	4

7. Obtener las asignaturas de un curso:

Muestra las asignaturas del curso cuya id es indicada en la URL

```
// Obtener asignaturas de un curso
Spark.get(path: "/cursos/asignaturas/:idcurs", (request, response) -> {
    Long idcurs = Long.parseLong(request.params(":idcurs"));
    Curso c= dao_curs.buscarPorId(idcurs);
    List<Asignatura> as = dao_asoc.cursosAsignaturas(c);
    response.type(contentType: "application/json");
    if (as!=null) {
        return gson.toJson(as);
    }
    else{
        return "No existe ese curso";
    }
});
```



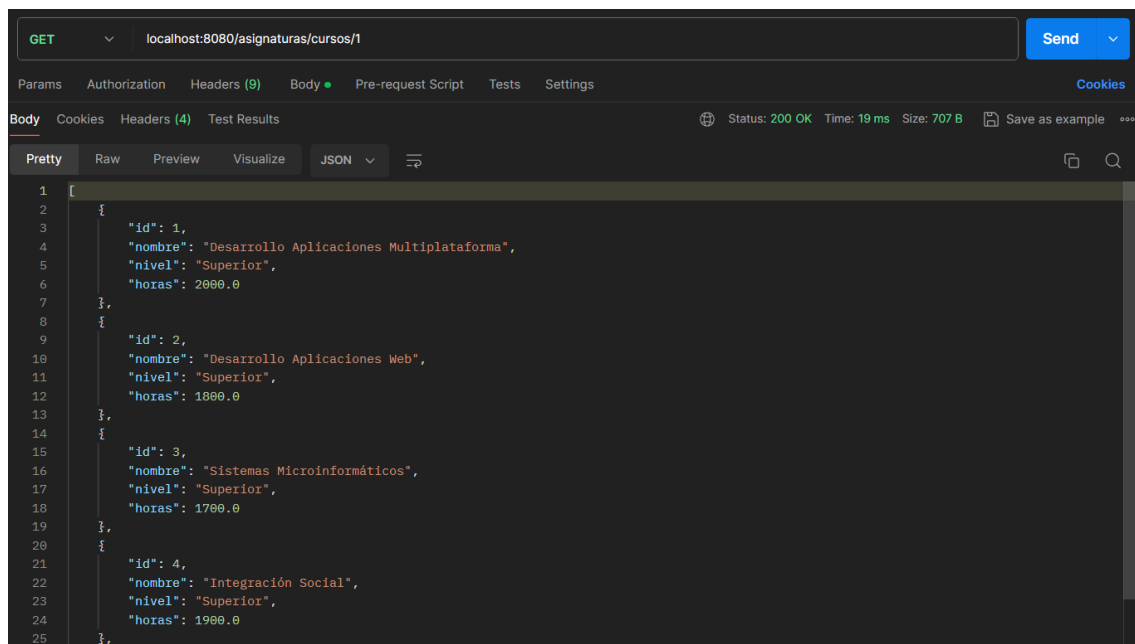
The screenshot shows a web browser interface with a GET request to `localhost:8080/cursos/asignaturas/1`. The response status is 200 OK, with a time of 25 ms and a size of 741 B. The response body is displayed in JSON format, showing a list of four subjects for the course with ID 1.

```
[{"id": 1, "nombre": "Ingl\u00e9s", "tipo": "Comun", "horas_semanales": 3}, {"id": 2, "nombre": "Emprendimiento e Iniciativa Emprendedora", "tipo": "Comun", "horas_semanales": 3}, {"id": 3, "nombre": "Desarrollo de Aplicaciones M\u00f3viles", "tipo": "Especifica", "horas_semanales": 8}, {"id": 4, "nombre": "Dise\u00f1o de Experiencia de Usuario", "tipo": "Especifica", "horas_semanales": 5}]
```

8. Obtener los cursos a los que pertenece una asignatura:

Muestra los cursos a los que pertenecen la asignatura con el id indicado en la URL

```
// Obtener cursos de una asignatura
Spark.get( path: "/asignaturas/cursos/:idasig", (request, response) -> {
    Long idasig = Long.parseLong(request.params(":idasig"));
    Asignatura as= dao_asig.buscarPorId(idasig);
    List<Curso> c = dao_asoc.asignaturasCursos(as);
    response.type( contentType: "application/json");
    if (c!=null) {
        return gson.toJson(c);
    }
    else{
        return "No existe esa asignatura";
    }
});
```



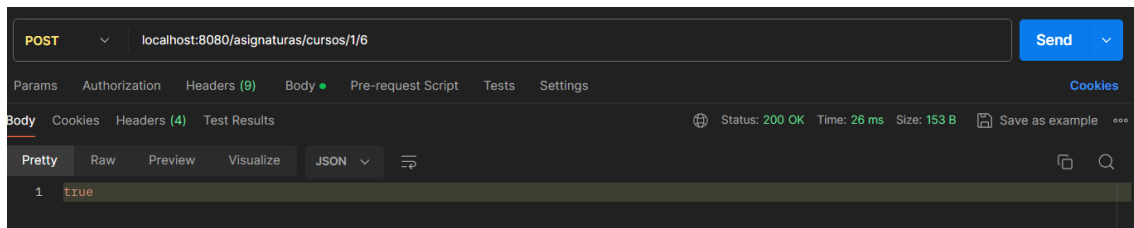
The screenshot shows a REST client interface with a GET request to `localhost:8080/asignaturas/cursos/1`. The response is a JSON array of four course objects. The status is 200 OK, time is 19 ms, and size is 707 B.

```
[{"id": 1, "nombre": "Desarrollo Aplicaciones Multiplataforma", "nivel": "Superior", "horas": 2000.0}, {"id": 2, "nombre": "Desarrollo Aplicaciones Web", "nivel": "Superior", "horas": 1800.0}, {"id": 3, "nombre": "Sistemas Microinformáticos", "nivel": "Superior", "horas": 1700.0}, {"id": 4, "nombre": "Integración Social", "nivel": "Superior", "horas": 1900.0}]
```

9. Asignar una asignatura a un curso:

Asigna una asignatura a un curso y añade el registro a la tabla relación “cursos_asignaturas”

```
// Asignar asignatura a curso
Spark.post(path: "/asignaturas/cursos/:idasig/:idcurs", (request, response) -> {
    Long idcurs = Long.parseLong(request.params(":idcurs"));
    Long idasig = Long.parseLong(request.params(":idasig"));
    Curso c= dao_curs.buscarPorId(idcurs);
    Asignatura as= dao_asig.buscarPorId(idasig);
    response.type(contentType: "application/json");
    return gson.toJson(dao_asoc.asignarAsignatura(c,as));
});
```



curso_id	asignatura_id
5	16
6	1