

Parcial

Cristian Esteban Cano Vega ID:824426

Corporación Universitaria Minuto de Dios

Ingeniería de Sistemas, 7° Semestre

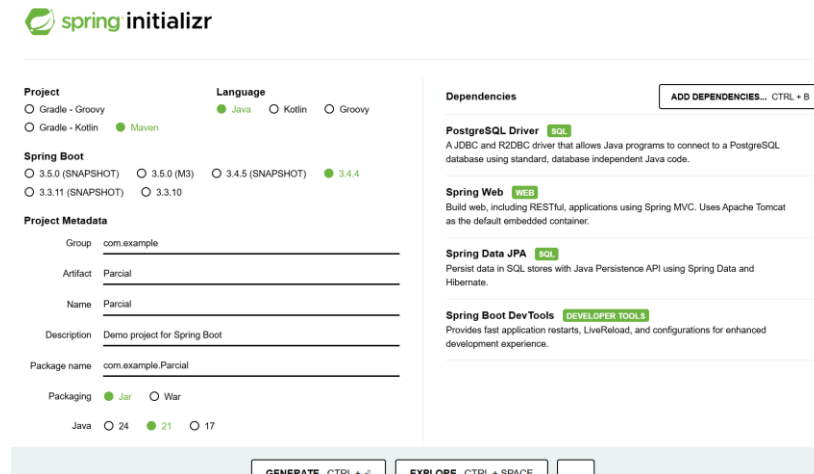
Arquitectura de Software NRC:66167

Pr. William Alexander Matallana Porras

22 de abril de 2025

Paso a Paso

1. Ingresamos a <https://start.spring.io/> ingresamos los siguientes ajustes y agregamos dependencias para empezar con el proyecto.



The screenshot shows the Spring Initializr web form with the following configuration:

- Project:**
 - ☐ Gradle - Groovy
 - ☐ Gradle - Kotlin
 - ☒ **Maven**
- Language:**
 - ☒ **Java**
 - ☐ Kotlin
 - ☐ Groovy
- Spring Boot:**
 - ☐ 3.5.0 (SNAPSHOT)
 - ☐ 3.5.0 (M3)
 - ☐ 3.4.5 (SNAPSHOT)
 - ☒ **3.4.4**
 - ☐ 3.3.11 (SNAPSHOT)
 - ☐ 3.3.10
- Project Metadata:**
 - Group:
 - Artifact:
 - Name:
 - Description:
 - Package name:
 - Packaging: ☒ **Jar** ☐ War
 - Java: ☐ 24 ☒ **21** ☐ 17
- Dependencies:**
 - PostgreSQL Driver** (JDBC): A JDBC and R2DBC driver that allows Java programs to connect to a PostgreSQL database using standard, database independent Java code.
 - Spring Web** (WEB): Build web, including RESTful, applications using Spring MVC. Uses Apache Tomcat as the default embedded container.
 - Spring Data JPA** (SQL): Persist data in SQL stores with Java Persistence API using Spring Data and Hibernate.
 - Spring Boot DevTools** (DEVELOPER TOOLS): Provides fast application restarts, LiveReload, and configurations for enhanced development experience.

Buttons at the bottom: **GENERATE** (CTRL + G), **EXPLORE** (CTRL + SPACE), and a menu icon.

2. Descomprimos el paquete y abrimos con IntelliJ, empezamos creando cada uno de los modelos con respectivas relaciones.

Entrenador

```

1 package com.example.Partial.Model;
2
3 import jakarta.persistence.*;
4
5 @Entity
6 public class Entrenador {
7     @Id
8     @GeneratedValue(strategy = GenerationType.IDENTITY)
9     private int id_entrenador;
10    private String nombre;
11    private String especialidad;
12
13    @ManyToOne
14    @JoinColumn(name = "id_equipo")
15    private Equipo equipo;
16
17    public Entrenador() {}
18
19    public Entrenador(int id_entrenador, String nombre, String especialidad, Equipo equipo) {
20        this.id_entrenador = id_entrenador;
21        this.nombre = nombre;
22        this.especialidad = especialidad;
23        this.equipo = equipo;
24    }
25
26    public int getId_entrenador() { return id_entrenador; }
27    public void setId_entrenador(int id_entrenador) { this.id_entrenador = id_entrenador; }
28    public String getNombre() { return nombre; }
29    public void setNombre(String nombre) { this.nombre = nombre; }
30    public String getEspecialidad() { return especialidad; }
31    public void setEspecialidad(String especialidad) { this.especialidad = especialidad; }
32    public Equipo getEquipo() { return equipo; }
33    public void setEquipo(Equipo equipo) { this.equipo = equipo; }
34

```

Equipo

```

1 package com.example.Parcial.Model;
2
3 import jakarta.persistence.*;
4 import java.util.Date;
5 import java.util.List;
6
7 @Entity 16 usages
8 public class Equipo {
9     @Id 4 usages
10     @GeneratedValue(strategy = GenerationType.IDENTITY)
11     private int id_equipo;
12     private String nombre; 4 usages
13     private String ciudad; 4 usages
14     private Date fundacion; 4 usages
15
16     @OneToMany(mappedBy = "equipo", cascade = CascadeType.ALL) no usages
17     private List<Jugador> jugadores;
18
19     @OneToMany(mappedBy = "equipo", cascade = CascadeType.ALL) no usages
20     private List<Entrenador> entrenadores;
21
22     public Equipo() {} no usages
23
24     public Equipo(int id_equipo, String nombre, String ciudad, Date fundacion) { no usages
25         this.id_equipo = id_equipo;
26         this.nombre = nombre;
27         this.ciudad = ciudad;
28         this.fundacion = fundacion;
29     }
30
31     public int getId_equipo() { return id_equipo; } no usages
32     public void setId_equipo(int id_equipo) { this.id_equipo = id_equipo; } no usages
33     public String getNombre() { return nombre; } no usages

```

EstadisticaJugador

```

1 package com.example.Parcial.Model;
2
3 import jakarta.persistence.*;
4
5 @Entity 2 usages
6 public class EstadisticaJugador {
7     @Id 4 usages
8     @GeneratedValue(strategy = GenerationType.IDENTITY)
9     private int id_estadistica;
10
11     private int minutos_jugados; 4 usages
12     private int goles; 4 usages
13     private int asistencias; 4 usages
14     private int tarjetas_amarillas; 4 usages
15     private int tarjetas_rojas; 4 usages
16
17     @ManyToOne 3 usages
18     @JoinColumn(name = "id_jugador")
19     private Jugador jugador;
20
21     @ManyToOne 3 usages
22     @JoinColumn(name = "id_partido")
23     private Partido partido;
24
25     public EstadisticaJugador() {} no usages
26
27     public EstadisticaJugador(int id_estadistica, int minutos_jugados, int goles, int asistencias,
28         this.id_estadistica = id_estadistica;
29         this.minutos_jugados = minutos_jugados;
30         this.goles = goles;
31         this.asistencias = asistencias;

```

Jugador

```

1 package com.example.Parcial.Model;
2
3 import jakarta.persistence.*;
4 import java.util.Date;
5 import java.util.List;
6
7 @Entity 5 usages
8 public class Jugador {
9     @Id 4 usages
10    @GeneratedValue(strategy = GenerationType.IDENTITY)
11    private int id_jugador;
12    private String nombre; 4 usages
13    private String posicion; 4 usages
14    private int dorsal; 4 usages
15    private Date fecha_nac; 4 usages
16    private String nacionalidad; 4 usages
17
18    @ManyToOne 3 usages
19    @JoinColumn(name = "id_equipo")
20    private Equipo equipo;
21
22    @OneToMany(mappedBy = "jugador", cascade = CascadeType.ALL) no usages
23    private List<EstadisticaJugador> estadisticas;
24
25    public Jugador() {} no usages
26
27    public Jugador(int id_jugador, String nombre, String posicion, int dorsal, Date fecha_nac, String
28        this.id_jugador = id_jugador;
29        this.nombre = nombre;
30        this.posicion = posicion;
31        this.dorsal = dorsal;

```

Partido

```

1 package com.example.Parcial.Model;
2
3 import jakarta.persistence.*;
4 import java.util.Date;
5 import java.util.List;
6
7 @Entity 4 usages
8 public class Partido {
9     @Id 4 usages
10    @GeneratedValue(strategy = GenerationType.IDENTITY)
11    private int id_partido;
12    private Date fecha; 4 usages
13    private String estadio; 4 usages
14    private int goles_local; 4 usages
15    private int goles_visita; 4 usages
16
17    @ManyToOne 3 usages
18    @JoinColumn(name = "equipo_local")
19    private Equipo equipoLocal;
20
21    @ManyToOne 3 usages
22    @JoinColumn(name = "equipo_visita")
23    private Equipo equipoVisita;
24
25    @OneToMany(mappedBy = "partido", cascade = CascadeType.ALL) no usages
26    private List<EstadisticaJugador> estadisticas;
27
28    public Partido() {} no usages
29

```

3. Creamos cada uno de los repositorios

Entrenador

```
EntrenadorRepository.java ×  
  
package com.example.Parcial.Repository;  
  
import com.example.Parcial.Model.Entrenador;  
import org.springframework.data.jpa.repository.JpaRepository;  
import org.springframework.stereotype.Repository;  
  
@Repository 3 usages new *  
public interface EntrenadorRepository extends JpaRepository<Entrenador, Integer> {  
}  
}
```

Equipo

```
EquipoRepository.java ×  
  
1 package com.example.Parcial.Repository;  
2  
3 import com.example.Parcial.Model.Equipo;  
4 import org.springframework.data.jpa.repository.JpaRepository;  
5 import org.springframework.stereotype.Repository;  
6  
7 @Repository 3 usages new *  
8 public interface EquipoRepository extends JpaRepository<Equipo, Integer> {  
9 }  
}
```

EstadisticaJugador

```
EstadisticaJugadorRepository.java ×  
  
1 package com.example.Parcial.Repository;  
2  
3 import com.example.Parcial.Model.EstadisticaJugador;  
4 import org.springframework.data.jpa.repository.JpaRepository;  
5 import org.springframework.stereotype.Repository;  
6  
7 @Repository 3 usages new *  
8 public interface EstadisticaJugadorRepository extends JpaRepository<EstadisticaJugador, Integer> {  
9 }  
}
```

Jugador

```
JugadorRepository.java x
package com.example.Parcial.Repository;

> import ...

@Repository 3 usages new *
public interface JugadorRepository extends JpaRepository<Jugador, Integer> {

    @Query(value = "SELECT * FROM jugador WHERE id_equipo = :id_equipo", nativeQuery = true) 1 usage
    List<Jugador> findByEquipoId(@Param("id_equipo") int id_equipo);

    @Query(value = "" 1 usage new *
    SELECT_
        j.id_jugador,
        j.nombre,
        j.posicion,
        j.dorsal,
        j.fecha_nac,
        j.nacionalidad,
        j.id_equipo,
        SUM(e.goles) AS total_goles
    FROM jugador j
    JOIN estadistica_jugador e ON j.id_jugador = e.id_jugador
    GROUP BY j.id_jugador, j.nombre, j.posicion, j.dorsal, j.fecha_nac, j.nacionalidad, j.id_equipo
    HAVING SUM(e.goles) > :goles
    ORDER BY total_goles DESC
    "", nativeQuery = true)
    List<Map<String, Object>> findJugadoresConMasGoles(@Param("goles") int goles);
}
```

Partido

```
PartidoRepository.java x
package com.example.Parcial.Repository;

> import ...

@Repository 3 usages new *
public interface PartidoRepository extends JpaRepository<Partido, Integer> {

    @Query(value = "" 1 usage new *
    SELECT_
        e.nombre AS equipo,
        SUM(CASE WHEN p.equipo_local = e.id_equipo THEN p.goles_local
            WHEN p.equipo_visita = e.id_equipo THEN p.goles_visita
            ELSE 0 END) AS total_goles
    FROM equipo e
    JOIN partido p ON e.id_equipo = p.equipo_local OR e.id_equipo = p.equipo_visita
    WHERE e.id_equipo = :id_equipo
    GROUP BY e.nombre
    "", nativeQuery = true)
    Map<String, Object> getTotalGolesEquipo(@Param("id_equipo") int id_equipo);

    @Query(value = "" 1 usage new *
    SELECT p.id_partido, p.fecha, p.estadio,
        el.nombre AS equipo_local,
        ev.nombre AS equipo_visita,
        p.goles_local, p.goles_visita
    FROM partido p
    JOIN equipo el ON p.equipo_local = el.id_equipo
    JOIN equipo ev ON p.equipo_visita = ev.id_equipo
    ORDER BY p.fecha DESC
    "", nativeQuery = true)
    List<Map<String, Object>> getResultadosPartidos();
}
```

4. Creamos los servicios

Entrenador

```
EntrenadorService.java x
1 package com.example.Parcial.Service;
2
3 > import ...
4
5 @Service 3 usages new *
6 public class EntrenadorService {
7     private final EntrenadorRepository entrenadorRepository; 5 usages
8
9     public EntrenadorService(EntrenadorRepository entrenadorRepository) { no usages
10         this.entrenadorRepository = entrenadorRepository;
11     }
12
13     public List<Entrenador> getAllEntrenadores() { 1 usage new *
14         return entrenadorRepository.findAll();
15     }
16
17     public Optional<Entrenador> getEntrenadorById(int id) { 1 usage new *
18         return entrenadorRepository.findById(id);
19     }
20
21     public Entrenador saveEntrenador(Entrenador entrenador) { 2 usages new *
22         return entrenadorRepository.save(entrenador);
23     }
24
25     public void deleteEntrenador(int id) { 1 usage new *
26         entrenadorRepository.deleteById(id);
27     }
28 }
```

Equipo

```
EquipoService.java x
1 package com.example.Parcial.Service;
2
3 > import ...
4
5 @Service 3 usages new *
6 public class EquipoService {
7     private final EquipoRepository equipoRepository; 5 usages
8
9     public EquipoService(EquipoRepository equipoRepository) { no usages
10         this.equipoRepository = equipoRepository;
11     }
12
13     public List<Equipo> getAllEquipos() { 1 usage new *
14         return equipoRepository.findAll();
15     }
16
17     public Optional<Equipo> getEquipoById(int id) { 1 usage new *
18         return equipoRepository.findById(id);
19     }
20
21     public Equipo saveEquipo(Equipo equipo) { 2 usages new *
22         return equipoRepository.save(equipo);
23     }
24
25     public void deleteEquipo(int id) { 1 usage new *
26         equipoRepository.deleteById(id);
27     }
28 }
```

EstadisticaJugador

```

1 package com.example.Parcial.Service;
2
3 > import ...
4
5
6
7
8
9
10 @Service 3 usages new *
11 public class EstadisticaJugadorService {
12     private final EstadisticaJugadorRepository estadisticaJugadorRepository; 5 usages
13
14     public EstadisticaJugadorService(EstadisticaJugadorRepository estadisticaJugadorRepository) {
15         this.estadisticaJugadorRepository = estadisticaJugadorRepository;
16     }
17
18     public List<EstadisticaJugador> getAllEstadisticas() { 1 usage new *
19         return estadisticaJugadorRepository.findAll();
20     }
21
22     public Optional<EstadisticaJugador> getEstadisticaById(int id) { 1 usage new *
23         return estadisticaJugadorRepository.findById(id);
24     }
25
26     public EstadisticaJugador saveEstadistica(EstadisticaJugador estadistica) { 2 usages new *
27         return estadisticaJugadorRepository.save(estadistica);
28     }
29
30     public void deleteEstadistica(int id) { 1 usage new *
31         estadisticaJugadorRepository.deleteById(id);
32     }
33 }

```

Jugador

```

1 package com.example.Parcial.Service;
2
3 > import ...
4
5
6
7
8
9
10
11 @Service 3 usages new *
12 public class JugadorService {
13     private final JugadorRepository jugadorRepository; 7 usages
14
15     public JugadorService(JugadorRepository jugadorRepository) { no usages new *
16         this.jugadorRepository = jugadorRepository;
17     }
18
19     public List<Jugador> getAllJugadores() { 1 usage new *
20         return jugadorRepository.findAll();
21     }
22
23     public Optional<Jugador> getJugadorById(int id) { 1 usage new *
24         return jugadorRepository.findById(id);
25     }
26
27     public Jugador saveJugador(Jugador jugador) { 2 usages new *
28         return jugadorRepository.save(jugador);
29     }
30
31     public void deleteJugador(int id) { 1 usage new *
32         jugadorRepository.deleteById(id);
33     }
34
35     public List<Jugador> getJugadoresByEquipo(int id_equipo) { 1 usage new *
36         return jugadorRepository.findByEquipoId(id_equipo);
37     }
38
39     public List<Map<String, Object>> getJugadoresConMasDeXGoles(int goles) { 1 usage new *
40         return jugadorRepository.findJugadoresConMasGoles(goles);
41     }
42 }

```


Partido

```

PartidoService.java
1 package com.example.Parcial.Service;
2
3 > import ...
4
5 @Service 3 usages new *
6 public class PartidoService {
7     private final PartidoRepository partidoRepository; 7 usages
8
9     public PartidoService(PartidoRepository partidoRepository) { no usages new *
10         this.partidoRepository = partidoRepository;
11     }
12
13     public List<Partido> getAllPartidos() { 1 usage new *
14         return partidoRepository.findAll();
15     }
16
17     public Optional<Partido> getPartidoById(int id) { 1 usage new *
18         return partidoRepository.findById(id);
19     }
20
21     public Partido savePartido(Partido partido) { 2 usages new *
22         return partidoRepository.save(partido);
23     }
24
25     public void deletePartido(int id) { 1 usage new *
26         partidoRepository.deleteById(id);
27     }
28
29     public Map<String, Object> getTotalGolesPorEquipo(int id_equipo) { 1 usage new *
30         return partidoRepository.getTotalGolesEquipo(id_equipo);
31     }
32
33     public List<Map<String, Object>> getResultadosDePartidos() { 1 usage new *
34         return partidoRepository.getResultadosPartidos();
35     }
36 }

```

5. Creamos los controladores para métodos básicos y consultas

Entrenador

```

EntrenadorController.java
29 @RestController no usages new *
30 @RequestMapping("/entrenadores")
31 public class EntrenadorController {
32
33     private final EntrenadorService entrenadorService; 6 usages
34
35     public EntrenadorController(EntrenadorService entrenadorService) { no usages new *
36         this.entrenadorService = entrenadorService;
37     }
38
39     @GetMapping no usages new *
40     public List<Entrenador> getAllEntrenadores() {
41         return entrenadorService.getAllEntrenadores();
42     }
43
44     @GetMapping("/{id}") no usages new *
45     public Optional<Entrenador> getEntrenadorById(@PathVariable int id) {
46         return entrenadorService.getEntrenadorById(id);
47     }
48
49     @PostMapping no usages new *
50     public Entrenador createEntrenador(@RequestBody Entrenador entrenador) {
51         return entrenadorService.saveEntrenador(entrenador);
52     }
53
54     @PutMapping("/{id}") no usages new *
55     public Entrenador updateEntrenador(@PathVariable int id, @RequestBody Entrenador entrenador) {
56         entrenador.setId_entrenador(id);
57         return entrenadorService.saveEntrenador(entrenador);
58     }
59
60     @DeleteMapping("/{id}") no usages new *
61     public void deleteEntrenador(@PathVariable int id) {
62         entrenadorService.deleteEntrenador(id);
63     }
64 }

```

Equipo

```

EquipoController.java
@RestController no usages new *
@RequestMapping("/equipos")
public class EquipoController {

    private final EquipoService equipoService; 6 usages

    public EquipoController(EquipoService equipoService) { no usages new *
        this.equipoService = equipoService;
    }

    @GetMapping no usages new *
    public List<Equipo> getAllEquipos() {
        return equipoService.getAllEquipos();
    }

    @GetMapping("/{id}") no usages new *
    public Optional<Equipo> getEquipoById(@PathVariable int id) {
        return equipoService.getEquipoById(id);
    }

    @PostMapping no usages new *
    public Equipo createEquipo(@RequestBody Equipo equipo) {
        return equipoService.saveEquipo(equipo);
    }

    @PutMapping("/{id}") no usages new *
    public Equipo updateEquipo(@PathVariable int id, @RequestBody Equipo equipo) {
        equipo.setId_equipo(id);
        return equipoService.saveEquipo(equipo);
    }

    @DeleteMapping("/{id}") no usages new *
    public void deleteEquipo(@PathVariable int id) {
        equipoService.deleteEquipo(id);
    }
}

```

EstadisticaJugador

```

EstadisticaJugadorController.java
@RestController no usages new *
@RequestMapping("/estadisticas")
public class EstadisticaJugadorController {

    private final EstadisticaJugadorService estadisticaJugadorService; 6 usages

    public EstadisticaJugadorController(EstadisticaJugadorService estadisticaJugadorService) { no usages new *
        this.estadisticaJugadorService = estadisticaJugadorService;
    }

    @GetMapping no usages new *
    public List<EstadisticaJugador> getAllEstadisticas() {
        return estadisticaJugadorService.getAllEstadisticas();
    }

    @GetMapping("/{id}") no usages new *
    public Optional<EstadisticaJugador> getEstadisticaById(@PathVariable int id) {
        return estadisticaJugadorService.getEstadisticaById(id);
    }

    @PostMapping no usages new *
    public EstadisticaJugador createEstadistica(@RequestBody EstadisticaJugador estadistica) {
        return estadisticaJugadorService.saveEstadistica(estadistica);
    }

    @PutMapping("/{id}") no usages new *
    public EstadisticaJugador updateEstadistica(@PathVariable int id, @RequestBody EstadisticaJugador estadistica) {
        estadistica.setId_estadistica(id);
        return estadisticaJugadorService.saveEstadistica(estadistica);
    }

    @DeleteMapping("/{id}") no usages new *
    public void deleteEstadistica(@PathVariable int id) {
        estadisticaJugadorService.deleteEstadistica(id);
    }
}

```

Jugador

```

JugadorController.java X
1 package com.example.Parcial.Controller;
2
3 > import ...
4
5
6
7
8
9
10
11 @RestController no usages new *
12 @RequestMapping("/jugadores")
13 public class JugadorController {
14
15     private final JugadorService jugadorService; 8 usages
16
17     public JugadorController(JugadorService jugadorService) { no usages new *
18         this.jugadorService = jugadorService;
19     }
20
21     @GetMapping no usages new *
22     public List<Jugador> getAllJugadores() {
23         return jugadorService.getAllJugadores();
24     }
25
26     @GetMapping("/{id}") no usages new *
27     public Optional<Jugador> getJugadorById(@PathVariable int id) {
28         return jugadorService.getJugadorById(id);
29     }
30
31     @PostMapping no usages new *
32     public Jugador createJugador(@RequestBody Jugador jugador) {
33         return jugadorService.saveJugador(jugador);
34     }
35
36     @PutMapping("/{id}") no usages new *
37     @
38     public Jugador updateJugador(@PathVariable int id, @RequestBody Jugador jugador) {
39         jugador.setId_jugador(id);
40         return jugadorService.saveJugador(jugador);
41     }
42
43     @DeleteMapping("/{id}") no usages new *
44     public void deleteJugador(@PathVariable int id) {
45         jugadorService.deleteJugador(id);
46     }
47
48     @GetMapping("/equipo/{id_equipo}") no usages new *
49     public List<Jugador> getJugadoresByEquipo(@PathVariable int id_equipo) {
50         return jugadorService.getJugadoresByEquipo(id_equipo);
51     }
52
53     @GetMapping("/goles-mayor/{goles}") no usages new *
54     public List<Map<String, Object>> getJugadoresConMasDeXGoles(@PathVariable int goles) {
55         return jugadorService.getJugadoresConMasDeXGoles(goles);
56     }
57 }

```

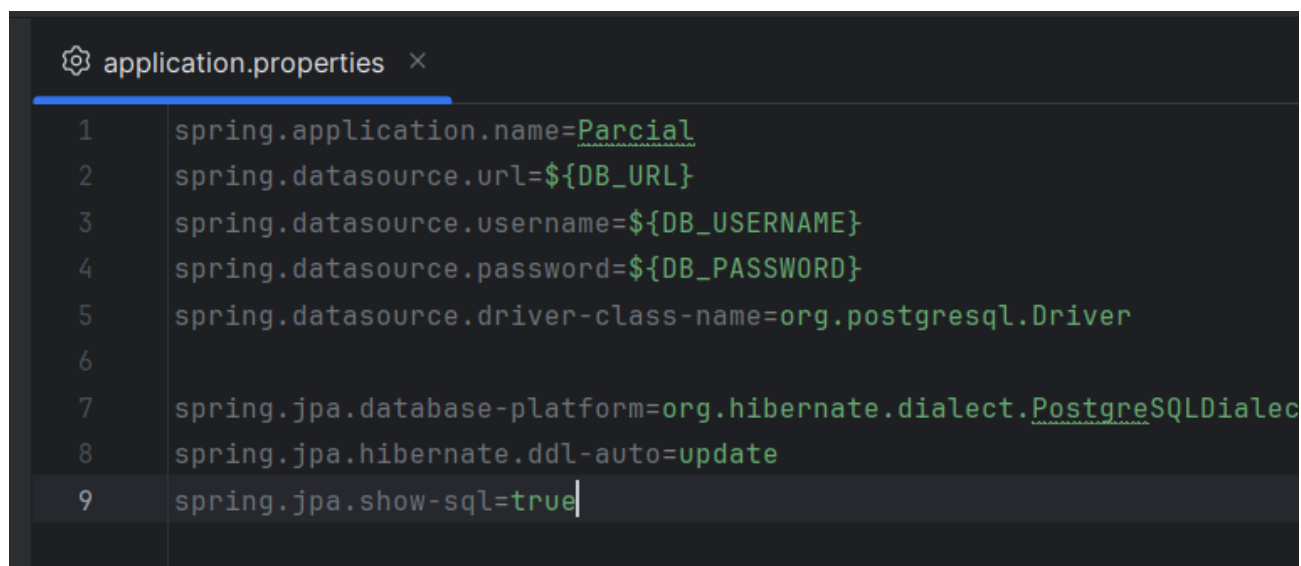
Partido

```

1 package com.example.Parcial.Controller;
2
3 > import ...
4
5
6
7
8
9
10
11 @RestController no usages new *
12 @RequestMapping("/partidos")
13 public class PartidoController {
14
15     private final PartidoService partidoService; 8 usages
16
17     public PartidoController(PartidoService partidoService) { no usages new *
18         this.partidoService = partidoService;
19     }
20
21     @GetMapping no usages new *
22     public List<Partido> getAllPartidos() {
23         return partidoService.getAllPartidos();
24     }
25
26     @GetMapping("/{id}") no usages new *
27     public Optional<Partido> getPartidoById(@PathVariable int id) {
28         return partidoService.getPartidoById(id);
29     }
30
31     @PostMapping no usages new *
32     public Partido createPartido(@RequestBody Partido partido) {
33         return partidoService.savePartido(partido);
34     }
35
36     @PutMapping("/{id}") no usages new *
37     public Partido updatePartido(@PathVariable int id, @RequestBody Partido partido) {
38         partido.setId_partido(id);
39         return partidoService.savePartido(partido);
40     }
41
42     @DeleteMapping("/{id}") no usages new *
43     public void deletePartido(@PathVariable int id) {
44         partidoService.deletePartido(id);
45     }
46
47     @GetMapping("/total-goles-equipo/{id_equipo}") no usages new *
48     public Map<String, Object> getTotalGolesEquipo(@PathVariable int id_equipo) {
49         return partidoService.getTotalGolesPorEquipo(id_equipo);
50     }
51
52     @GetMapping("/resultados") no usages new *
53     public List<Map<String, Object>> getResultadosPartidos() {
54         return partidoService.getResultadosDePartidos();
55     }
56 }
57

```

6. Vamos a agregar la conexión con el supabase con PostgreSQL

A screenshot of an IDE showing the 'application.properties' file. The file contains configuration for a Spring application connected to a PostgreSQL database. The properties are: spring.application.name=Parcial, spring.datasource.url=\${DB_URL}, spring.datasource.username=\${DB_USERNAME}, spring.datasource.password=\${DB_PASSWORD}, spring.datasource.driver-class-name=org.postgresql.Driver, spring.jpa.database-platform=org.hibernate.dialect.PostgreSQLDialect, spring.jpa.hibernate.ddl-auto=update, and spring.jpa.show-sql=true. The text is in a dark-themed editor with syntax highlighting.

```
1 spring.application.name=Parcial
2 spring.datasource.url=${DB_URL}
3 spring.datasource.username=${DB_USERNAME}
4 spring.datasource.password=${DB_PASSWORD}
5 spring.datasource.driver-class-name=org.postgresql.Driver
6
7 spring.jpa.database-platform=org.hibernate.dialect.PostgreSQLDialect
8 spring.jpa.hibernate.ddl-auto=update
9 spring.jpa.show-sql=true
```

7. Vamos a la página de <https://supabase.com> donde vamos a crear un nuevo proyecto y copiar las credenciales para la conexión

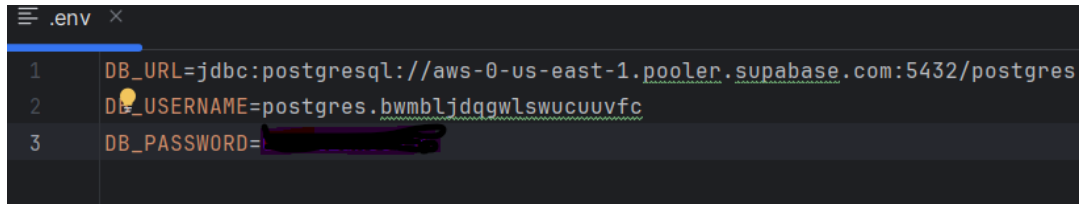
Session pooler ☒ Shared Pooler

Only recommended as an alternative to Direct Connection, when connecting via an IPv4 network.

postgresql://postgres.tuadltpbsjusofsmvikp:[YOUR-PASSWORD]@

> View parameters

8. Creamos un archivo .env y ahí ingresamos las credenciales



```
.env
1 DB_URL=jdbc:postgresql://aws-0-us-east-1.pooler.supabase.com:5432/postgres
2 DB_USERNAME=postgres.bwmbldqgqlswucuvfc
3 DB_PASSWORD=[REDACTED]
```

9. Cargamos las credenciales



```
@SpringBootApplication
public class PartialApplication {

    public static void main(String[] args) {
        loadEnv();
        SpringApplication.run(PartialApplication.class, args);
    }

    private static void loadEnv(){ 1 usage
        Dotenv dotenv = Dotenv.load();
        System.setProperty("DB_URL",dotenv.get("DB_URL"));
        System.setProperty("DB_USERNAME",dotenv.get("DB_USERNAME"));
        System.setProperty("DB_PASSWORD",dotenv.get("DB_PASSWORD"));
    }
}
```

10. En Docker teniendo descargada la imagen de pgAdmin vamos a crear un contenedor nuevo para hacer la conexión a este.

```
PS C:\Users\canox> docker run -d --name parcial2 -p 5000:80 -e PGADMIN_DEFAULT_EMAIL=canox09@gmail.com -e PGADMIN_DEFAULT_PASSWORD=Cristian8503#
dpage/pgadmin4
e852a8298a1f8b671edee12863aed0e302038e107ed6075d5baa8ab9eb148dc1
PS C:\Users\canox> █
```

11. Ingresamos a la ruta donde está corriendo el pgAdmin y ingresamos con los datos



12. Creamos un nuevo server con las mismas configuraciones

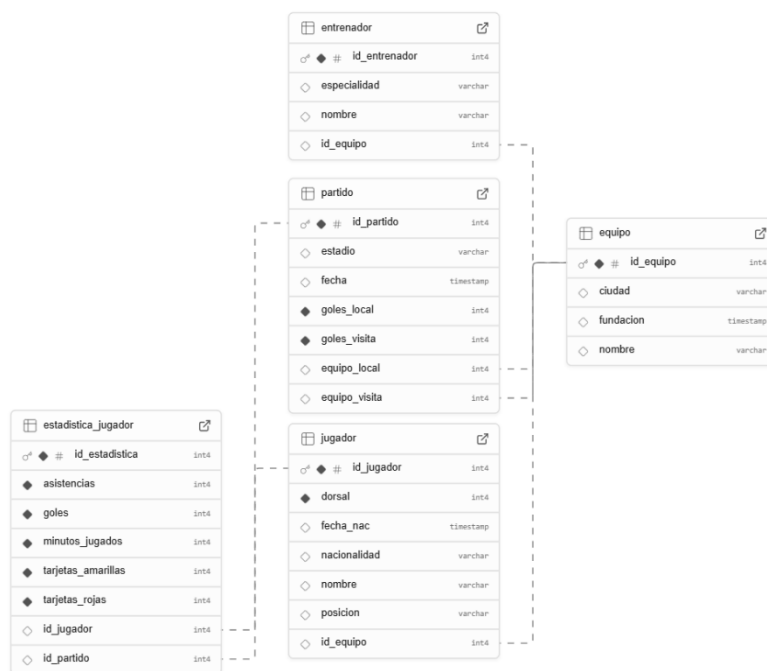
Register - Server [X]

General Connection Parameters SSH Tunnel Advanced Tags

Host name/address	aws-0-us-east-1.pooler.supabase.com
Port	5432
Maintenance database	postgres
Username	postgres.tuadltpbsjusofsmvikp
Kerberos authentication?	<input type="checkbox"/>
Password
Save password?	<input type="checkbox"/>
Role	
Service	

[i] [?] [X Close] [↺ Reset] [Save]

13. Ejecutamos el proyecto y queda terminada la estructura de la base de datos



14. Insertamos registros a cada tabla por medio del SQL Editor de supabase

SQL Editor

Search queries...

+

Templates

Quickstarts

> SHARED

> FAVORITES

> PRIVATE (1)

Insertar Equipos, Jugadores, E...

```

1 INSERT INTO estadistica_jugador (id_estadistica, asistencias, goles, minutos_jugados, tarjetas_amarillas, tarjetas_rojas, id_jugador, id_partido)
2 -- Jornada 1: Barcelona vs Real Madrid
3 (1, 2, 1, 90, 0, 0, 1, 1), -- Lewandowski (Barcelona)
4 (2, 1, 1, 90, 1, 0, 2, 1), -- Lamine Yamal (Barcelona)
5 (3, 0, 1, 90, 0, 0, 5, 1), -- Pedri (Barcelona)
6 (4, 0, 0, 90, 0, 0, 7, 1), -- Bellingham (Real Madrid)
7 (5, 0, 0, 90, 1, 0, 6, 1), -- Vinicius (Real Madrid)
8
9 -- Jornada 1: Atlético Madrid vs Sevilla
10 (6, 1, 1, 90, 0, 0, 11, 2), -- Griezmann (Atlético)
11 (7, 0, 1, 90, 0, 0, 12, 2), -- Koke (Atlético)
12 (8, 0, 0, 90, 1, 0, 16, 2), -- Navas (Sevilla)
13 (9, 0, 0, 78, 0, 0, 18, 2), -- En-Nesyri (Sevilla)
14 (10, 0, 0, 90, 2, 0, 17, 2), -- Sergio Ramos (Sevilla)
15
16 -- Jornada 2: Celta vs Osasuna
17 (11, 0, 1, 90, 0, 0, 51, 6), -- Iago Aspas (Celta)
18 (12, 1, 0, 85, 1, 0, 54, 6), -- Fran Beltrán (Celta)
19 (13, 0, 0, 90, 0, 0, 56, 6), -- Budimir (Osasuna)
20 (14, 0, 0, 70, 1, 0, 59, 6), -- Moncayola (Osasuna)
21 (15, 0, 0, 90, 0, 0, 58, 6), -- Sergio Herrera (Osasuna)
22
23 -- Jornada 2: Getafe vs Rayo Vallecano
24 (16, 0, 0, 90, 1, 0, 61, 7), -- Mayoral (Getafe)
25 (17, 0, 0, 75, 0, 0, 62, 7), -- Greenwood (Getafe)
26 (18, 0, 0, 90, 0, 0, 66, 7), -- Raúl de Tomás (Rayo)

```

Results

Chart

Click Run to execute your query.

Consultas

1. Obtener todos los jugadores de un equipo específico

Método

```
@Query(value = "SELECT * FROM jugador WHERE id_equipo = :id_equipo", nativeQuery = true)
List<Jugador> findByEquipoId(@Param("id_equipo") int id_equipo);
```

```
@GetMapping("/equipo/{id_equipo}") no usages new *
public List<Jugador> getJugadoresByEquipo(@PathVariable int id_equipo) {
    return jugadorService.getJugadoresByEquipo(id_equipo);
}
```

Parametros y Respuesta

GET http://localhost:8080/jugadores/equipo/1

Params Authorization Headers (6) Body Scripts Settings

Body Cookies Headers (5) Test Results ↺

{ } JSON Preview Visualize

```

1  [
2    {
3      "id_jugador": 1,
4      "nombre": "Robert Lewandowski",
5      "posicion": "Delantero",
6      "dorsal": 9,
7      "fecha_nac": "1988-08-21T05:00:00.000+00:00",
8      "nacionalidad": "Polonia",
9      "equipo": {
10         "id_equipo": 1,
11         "nombre": "FC Barcelona",
12         "ciudad": "Barcelona",
13         "fundacion": "1899-11-29T05:00:00.000+00:00"
14       }
15     },
16     {
17       "id_jugador": 2,
18       "nombre": "Lamine Yamal",
19       "posicion": "Extremo derecho",
20       "dorsal": 19,
21       "fecha_nac": "2007-07-13T05:00:00.000+00:00",
22       "nacionalidad": "España",
23       "equipo": {
24         "id_equipo": 1,
25         "nombre": "FC Barcelona",
26         "ciudad": "Barcelona",
27         "fundacion": "1899-11-29T05:00:00.000+00:00"
28       }
29     },
30     {
31       "id_jugador": 3,
32       "nombre": "Marc-André ter Stegen",

```

2. Obtener los jugadores que han marcado más de X goles

Método

```
@Query(value = "" 1 usage new *
SELECT
    j.id_jugador,
    j.nombre,
    j.posicion,
    j.dorsal,
    j.fecha_nac,
    j.nacionalidad,
    j.id_equipo,
    SUM(e.goles) AS total_goles
FROM jugador j
JOIN estadistica_jugador e ON j.id_jugador = e.id_jugador
GROUP BY j.id_jugador, j.nombre, j.posicion, j.dorsal, j.fecha_nac, j.nacionalidad, j.id_equipo
HAVING SUM(e.goles) > :goles
ORDER BY total_goles DESC
, nativeQuery = true)
List<Map<String, Object>> findJugadoresConMasGoles(@Param("goles") int goles);
```

```
@GetMapping("/goles-mayor/{goles}") no usages new *
public List<Map<String, Object>> getJugadoresConMasDeXGoles(@PathVariable int goles) {
    return jugadorService.getJugadoresConMasDeXGoles(goles);
}
```

Parámetros y Respuesta

GET http://localhost:8080/jugadores/goles-mayor/3

Params Authorization Headers (6) Body Scripts Settings

Body Cookies Headers (5) Test Results ↺

{ } JSON Preview Visualize

```
1 [
2   {
3     "id_jugador": 1,
4     "fecha_nac": "1988-08-21T05:00:00.000+00:00",
5     "total_goles": 9,
6     "nombre": "Robert Lewandowski",
7     "dorsal": 9,
8     "posicion": "Delantero",
9     "nacionalidad": "Polonia",
10    "id_equipo": 1
11  },
12  {
13    "fecha_nac": "2007-07-13T05:00:00.000+00:00",
14    "dorsal": 19,
15    "id_jugador": 2,
16    "nombre": "Lamine Yamal",
17    "nacionalidad": "España",
18    "posicion": "Extremo derecho",
19    "total_goles": 5,
20    "id_equipo": 1
21  },
22  {
23    "dorsal": 7,
24    "fecha_nac": "1986-10-12T05:00:00.000+00:00",
25    "posicion": "Delantero",
26    "total_goles": 4,
27    "nacionalidad": "Uruguay",
28    "id_equipo": 16,
29    "id_jugador": 76,
```

3. Obtener el número total de goles marcados por un equipo en todos sus partidos

Método

```
@Query(value = "" 1usage new *
SELECT
  e.nombre AS equipo,
  SUM(CASE WHEN p.equipo_local = e.id_equipo THEN p.goles_local
           WHEN p.equipo_visita = e.id_equipo THEN p.goles_visita
           ELSE 0 END) AS total_goles
FROM equipo e
JOIN partido p ON e.id_equipo = p.equipo_local OR e.id_equipo = p.equipo_visita
WHERE e.id_equipo = :id_equipo
GROUP BY e.nombre
nativeQuery = true)
Map<String, Object> getTotalGolesEquipo(@Param("id_equipo") int id_equipo);
```

```
@GetMapping("/total-goles-equipo/{id_equipo}") no usages new *
public Map<String, Object> getTotalGolesEquipo(@PathVariable int id_equipo) {
    return partidoService.getTotalGolesPorEquipo(id_equipo);
}
```

Parametro y Respuesta

The screenshot shows a REST client interface. At the top, a GET request is defined for the URL `http://localhost:8080/partidos/total-goles-equipo/2`. Below this, the 'Body' tab is selected, showing a JSON response. The response is a map with two entries: 'equipo' with the value 'Real Madrid' and 'total_goles' with the value 24. The interface includes tabs for Params, Authorization, Headers (6), Body, Scripts, and Settings. Below the Body tab, there are options for JSON, Preview, and Visualize.

```
GET http://localhost:8080/partidos/total-goles-equipo/2
```

Params Authorization Headers (6) Body Scripts Settings

Body Cookies Headers (5) Test Results ↺

{ } JSON Preview Visualize

```
1 {
2   "equipo": "Real Madrid",
3   "total_goles": 24
4 }
```

4. Obtener los resultados de todos los partidos indicando los nombres de los equipos

Método

```
@Query(value = "" 1 usage new *
SELECT p.id_partido, p.fecha, p.estadío,
       el.nombre AS equipo_local,
       ev.nombre AS equipo_visita,
       p.goles_local, p.goles_visita
FROM partido p
JOIN equipo el ON p.equipo_local = el.id_equipo
JOIN equipo ev ON p.equipo_visita = ev.id_equipo
ORDER BY p.fecha DESC
, nativeQuery = true)
List<Map<String, Object>> getResultadosPartidos();
```

```
@GetMapping("/resultados") no usages new *
public List<Map<String, Object>> getResultadosPartidos() {
    return partidoService.getResultadosDePartidos();
}
```

Parámetro y Respuesta

The screenshot shows a REST client interface with a GET request to `http://localhost:8080/partidos/resultados`. The response is displayed in JSON format, showing a list of match results with details like team names, dates, stadiums, and scores.

```
{
  [
    {
      "equipo_local": "Celta de Vigo",
      "fecha": "2025-01-28T02:00:00.000+00:00",
      "estadío": "Balaidos",
      "id_partido": 100,
      "equipo_visita": "Villarreal CF",
      "goles_local": 0,
      "goles_visita": 2
    },
    {
      "estadío": "Son Moix",
      "equipo_visita": "Real Sociedad",
      "equipo_local": "RCD Mallorca",
      "id_partido": 99,
      "goles_visita": 1,
      "fecha": "2025-01-27T00:30:00.000+00:00",
      "goles_local": 1
    },
    {
      "estadío": "Nuevo Los Cármenes",
      "id_partido": 98,
      "fecha": "2025-01-26T22:30:00.000+00:00",
      "equipo_local": "Granada CF",
      "equipo_visita": "Athletic Club",
      "goles_local": 2,
      "goles_visita": 3
    },
    {
      "equipo_local": "Deportivo Alavés",
      "estadío": "Mendizorroza",
      "equipo_visita": "Valencia CF",
    }
  ]
}
```