Guía Desarrollo Basado en Plataformas

Juan Felipe Tarazona Pita ID:832837

Ingeniería de sistemas, séptimo semestre, Corporación Universitaria Minuto de Dios

Arquitectura de software/Desarrollo Basado en plataforma

06/04/2025

Descargamos la imagen de “pgadmin”

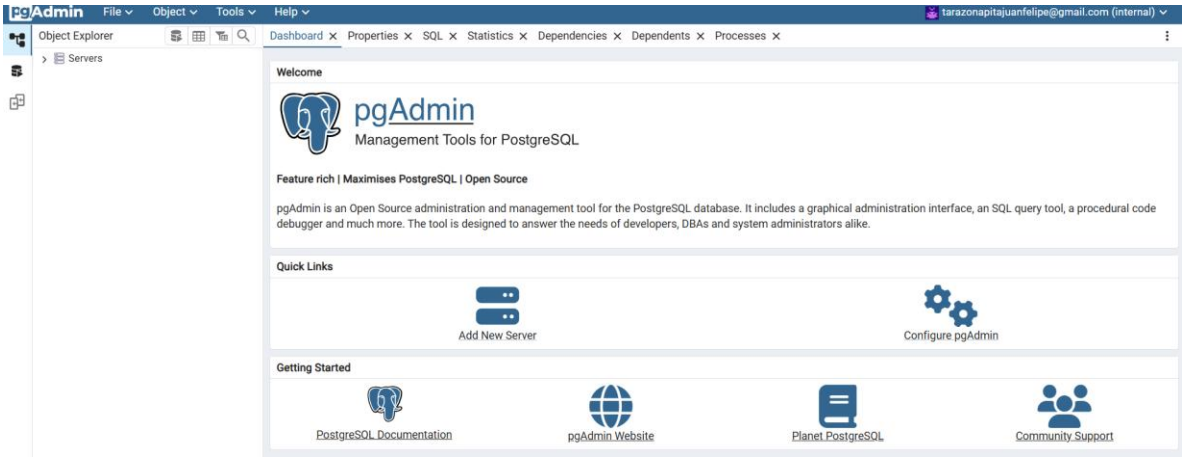
```
Terminal

Windows PowerShell
Copyright (C) Microsoft Corporation. Todos los derechos reservados.

Instale la versión más reciente de PowerShell para obtener nuevas características y mejoras. https://aka.ms/PSWindows

PS C:\Users\USUARIO> docker pull dpage/pgadmin4
Using default tag: latest
latest: Pulling from dpage/pgadmin4
6b5b0437a91d: Pulling fs layer
236726433656: Pulling fs layer
```

Levantamos un contenedor en el docker

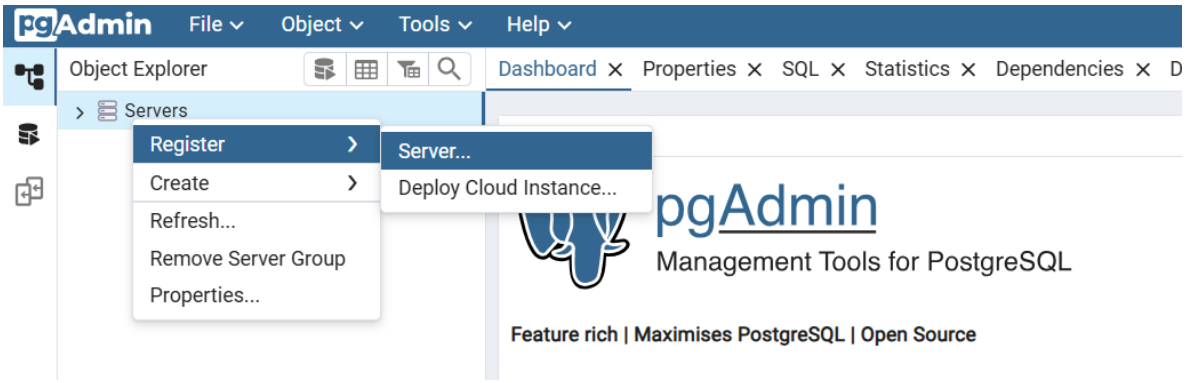
```
Terminal

PS C:\Users\USUARIO> docker images
REPOSITORY       TAG       IMAGE ID       CREATED        SIZE
dpage/pgadmin4   latest    bdebdfc4b165   5 weeks ago    760MB
mysql            9.2       9b9d0aab4860   2 months ago   1.09GB
mysql            latest    9b9d0aab4860   2 months ago   1.09GB
PS C:\Users\USUARIO> docker run -d --name pgadmin -p 5050:80 -e PGADMIN_DEFAULT_EMAIL=tarazonapitajuanfelipe@gmail.com -e PGADMIN_DEFAULT_PASSWORD
=12345 dpage/pgadmin4
>>
4e1358f7c3da926b55840e382517b04ec283a32845a1d5f0aa5f4cdfff834001
PS C:\Users\USUARIO>
```

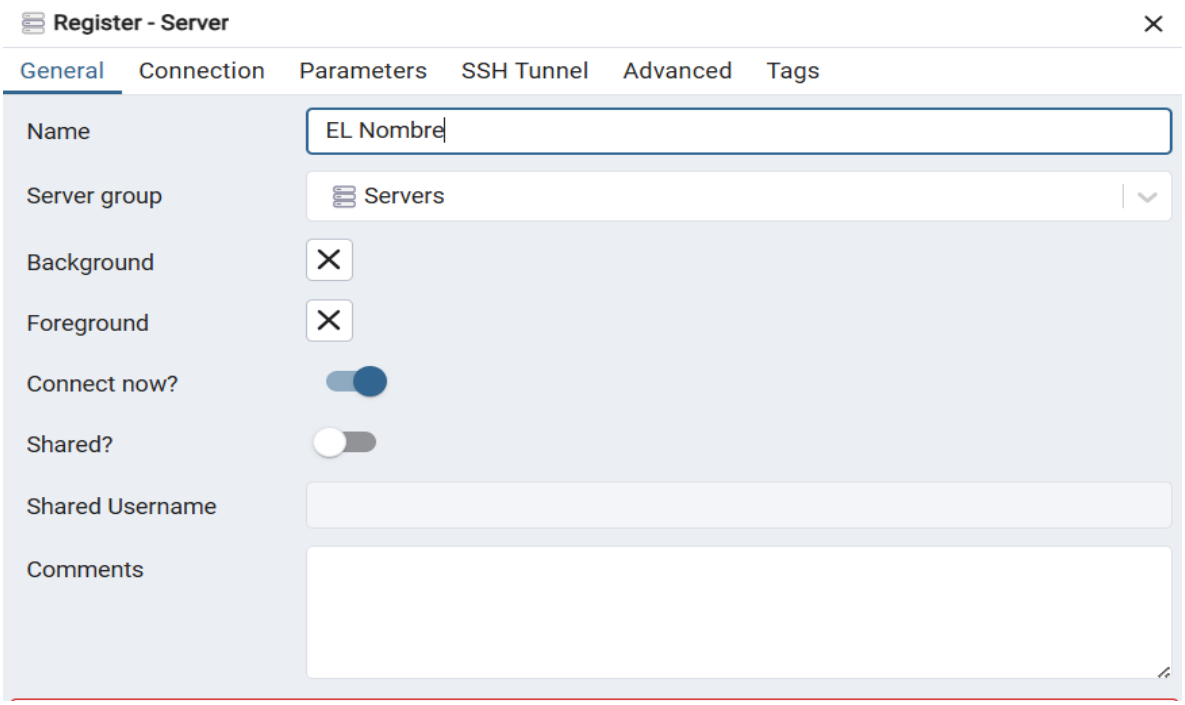
Seleccionamos el puerto y no lleva a una paginas donde ingresamos el correo y la contraseña

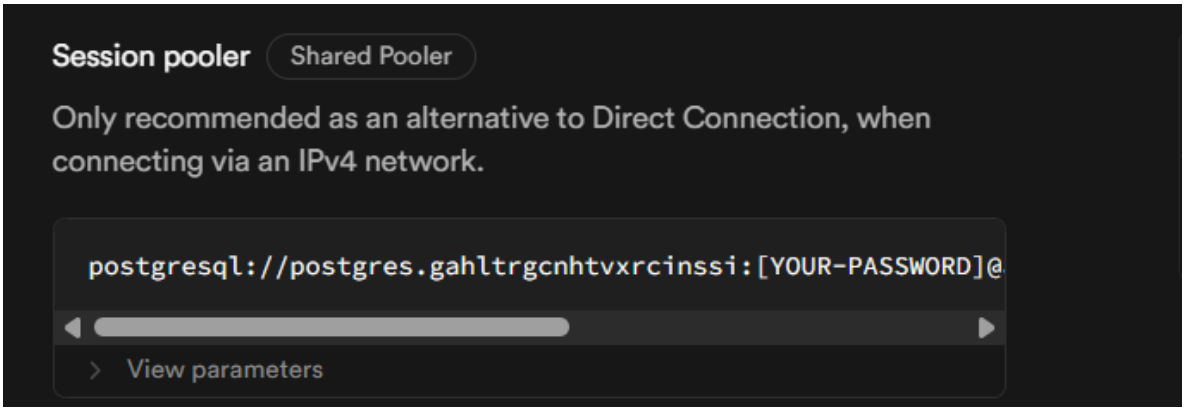
	Name	Container ID	Image	Port(s)	CPU (%)	Last started
☐	● Profesor	9463202ee742	mysql:latest	3307:3306	0.6%	53 minutes ago
☐	● pgadmin	4e1358f7c3da	dpage/pgadmin4	5050:80	0.03%	25 minutes ago

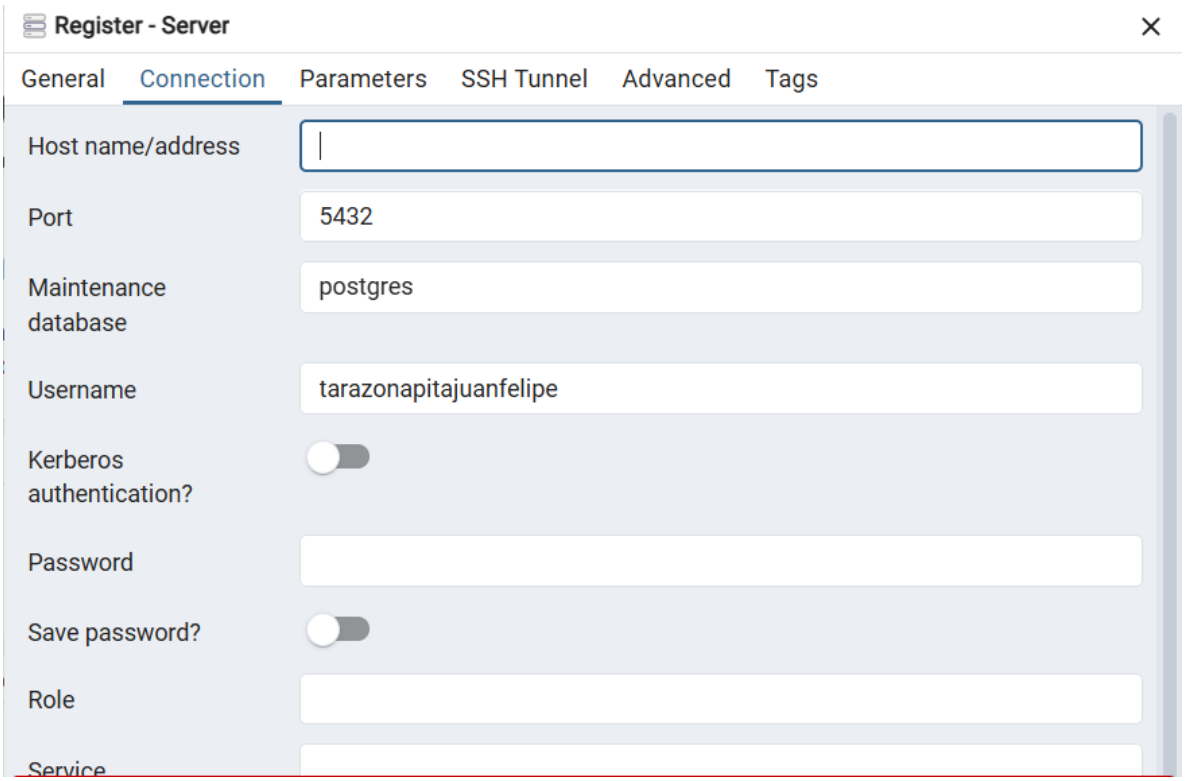


Generamos la conexión con la Base de Datos en base a los datos o la información proporcionada

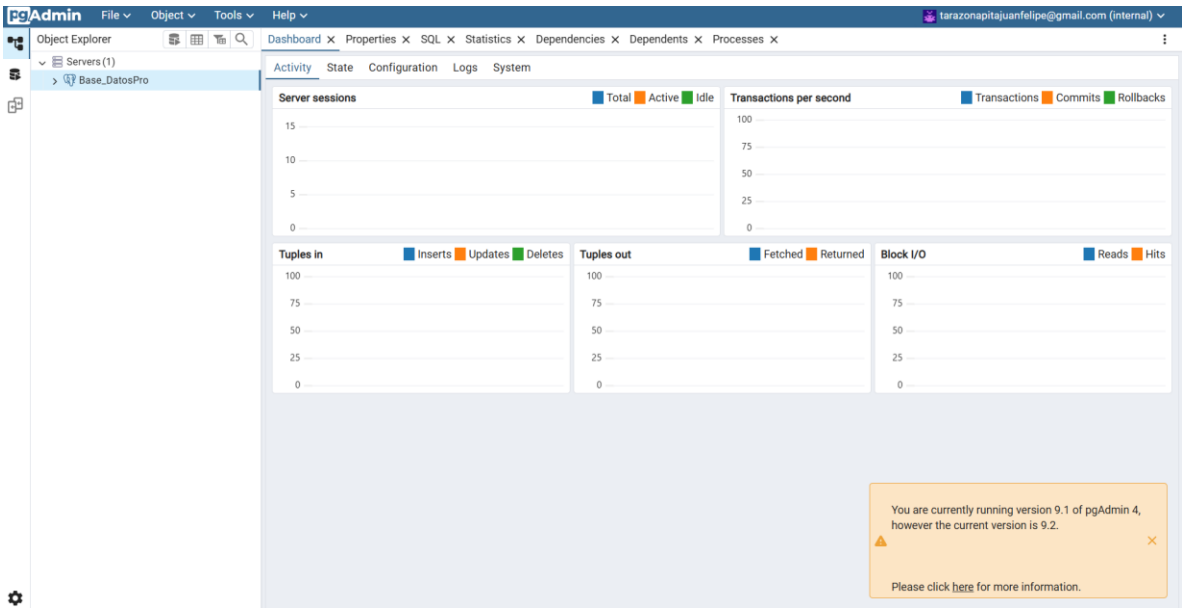

pgAdmin
File
Object
Tools
Help
Object Explorer
Dashboard
Properties
SQL
Statistics
Dependencies
Servers
Register
Create
Refresh...
Remove Server Group
Properties...
Server...
Deploy Cloud Instance...
pgAdmin
Management Tools for PostgreSQL
Feature rich | Maximises PostgreSQL | Open Source


Register - Server
General
Connection
Parameters
SSH Tunnel
Advanced
Tags
Name
EL Nombre
Server group
Servers
Background
Foreground
Connect now?
Shared?
Shared Username
Comments

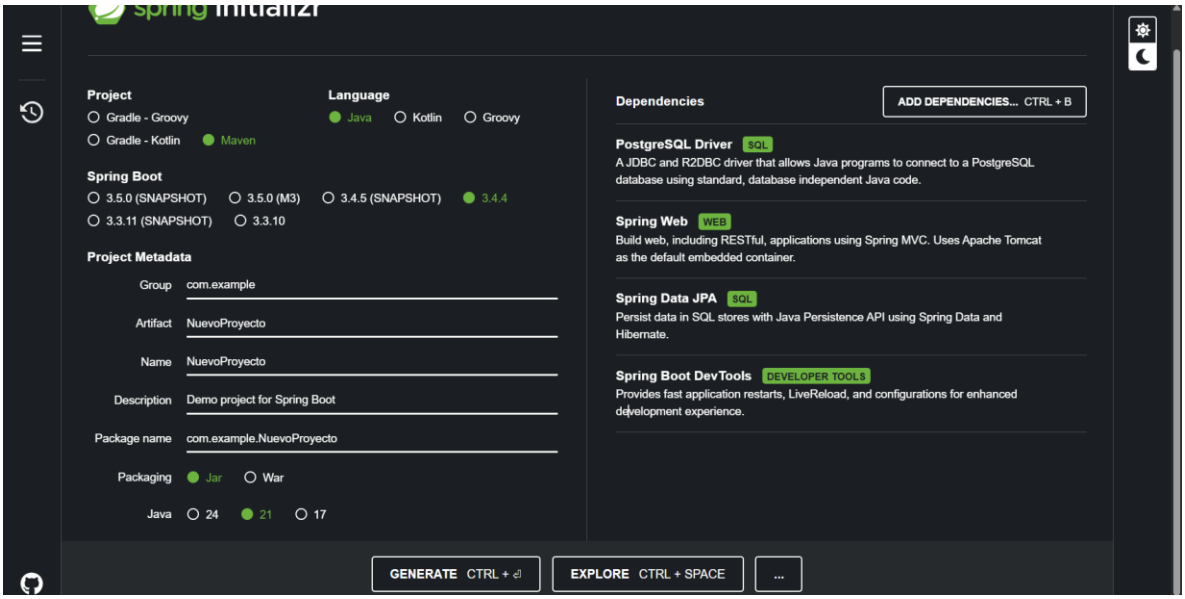

Session pooler
Shared Pooler
Only recommended as an alternative to Direct Connection, when connecting via an IPv4 network.
postgresql://postgres.gahltrgcnhtvxrcinssi:[YOUR-PASSWORD]@
View parameters



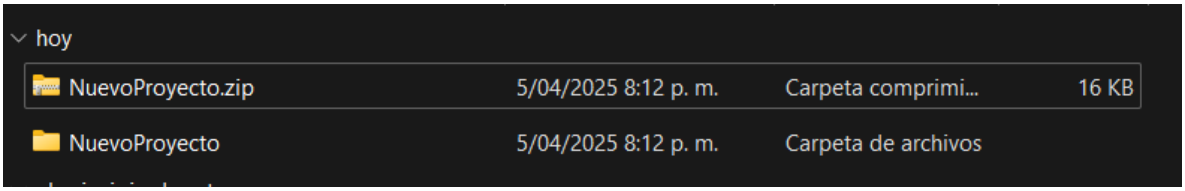
Ahora vamos a pegar todos los datos para establecer la conexión y una vez finalizado debe quedar así

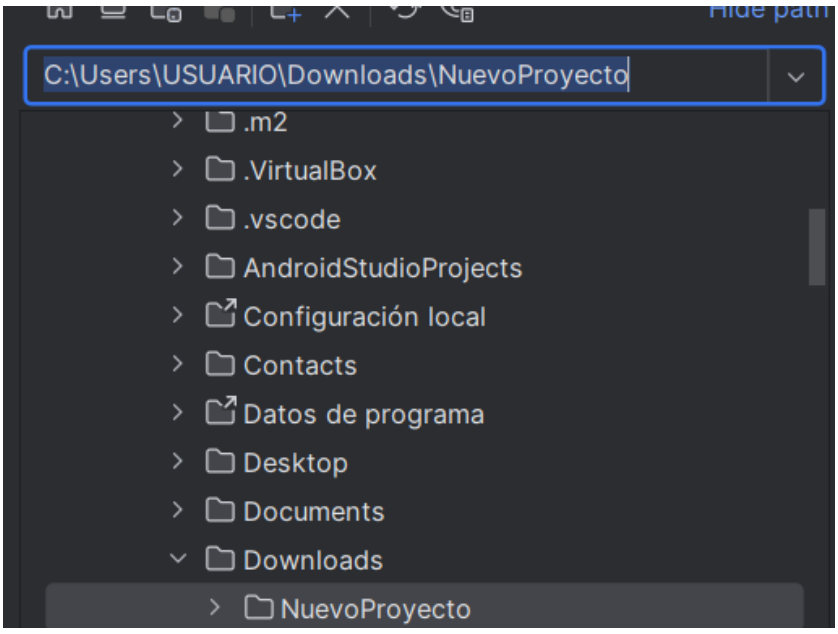


Ahora desde el Spring intallinz y generamos un archivo con las siguientes configuraciones

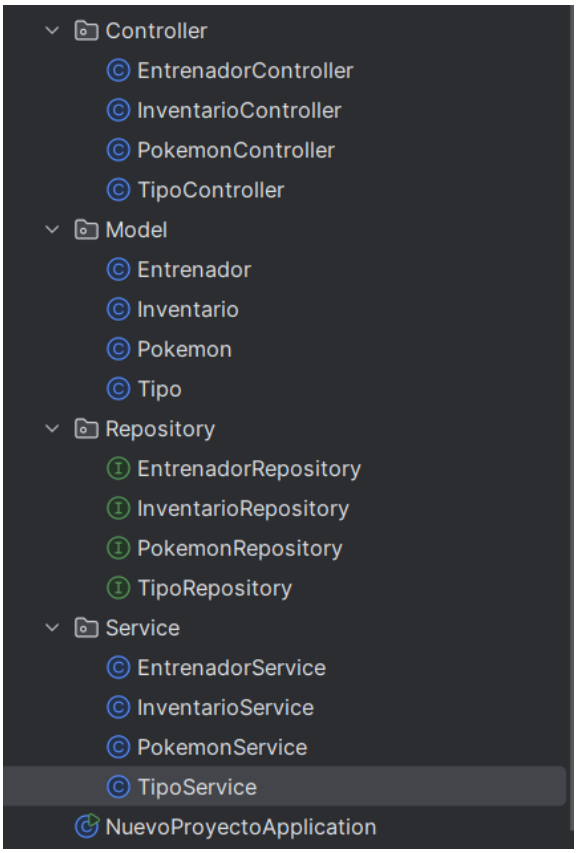


Una vez que descomprimimos y abrimos en el archivo en el el Intellij DAE lo que hacemos en configurar y crear nuestros paquetes y dentro de estos nuestras clases y demás.

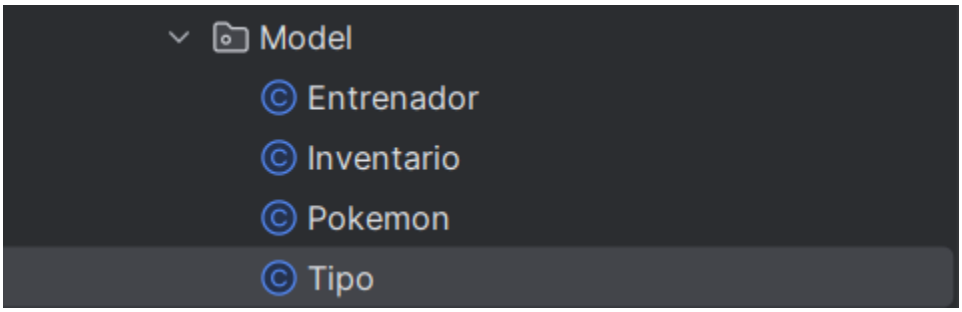




Generamos cuatro paquetes los cuales son, el controlador, los modelos, el repositorio y los servicios



Unas vez hemos creado todo no vamos a centrar en los modelos,



Dentro de los modelos tenemos varios atributos como vemos en la siguientes imagenes

```java
package com.example.NuevoProyecto.Model;

import jakarta.persistence.*;

@Entity  10 usages  JuanFTP2233
public class Tipo {
    @Id  2 usages
    @GeneratedValue(strategy = GenerationType.IDENTITY)
    private int id;
    private String nombre;  3 usages
    private String descripcion;  3 usages
```

```java
package com.example.NuevoProyecto.Model;

import jakarta.persistence.*;

@Entity  10 usages  JuanFTP2233
public class Pokemon {
    @Id  2 usages
    @GeneratedValue(strategy = GenerationType.IDENTITY)
    private int id;
    private String nombre;  3 usages
    private int nivel;  3 usages
    private String habilidad;  3 usages
```

```java
package com.example.NuevoProyecto.Model;

import jakarta.persistence.*;

@Entity  10 usages  JuanFTP2233
public class Inventario {
    @Id  2 usages
    @GeneratedValue(strategy = GenerationType.IDENTITY)
    private int id;
    private String objeto;  3 usages
    private int cantidad;  3 usages
    private String descripcion;  3 usages
```

```java
package com.example.NuevoProyecto.Model;

import jakarta.persistence.*;

@Entity  10 usages  JuanFTP2233
public class Entrenador {
    @Id  2 usages
    @GeneratedValue(strategy = GenerationType.IDENTITY)
    private int id;
    private String nombre;  3 usages
    private String region;  3 usages
    private int edad;  3 usages
    private String especialidad;  3 usages
```

Lo que debemos hacer ahora en relacionar la tablas entre si, para eso vamos a usar las etiquetas ManyToOne,OneToMany y jOINColum.

Las anotaciones @ManyToOne y @OneToMany en Java con JPA (Hibernate) se usan para definir **relaciones entre entidades** (tablas). @ManyToOne se pone en la clase que tiene **muchos registros relacionados con uno solo** en otra tabla; por ejemplo, muchos estudiantes pueden pertenecer a un solo curso. En cambio, @OneToMany se pone en la clase que representa **uno**

que se relaciona con muchos; por ejemplo, un curso tiene muchos estudiantes. La anotación @JoinColumn indica la columna en la base de datos que se usará como **clave foránea** para unir esas dos tablas. Es decir, le dice a JPA: "usa esta columna para enlazar con la otra tabla". Así, estas anotaciones permiten que las entidades se conecten entre sí sin que tengas que hacer las uniones a mano en SQL.

Después de realizar y relacionar las tablas, nos damos cuenta de que IntelliJ IDEA se llena de errores. Esto ocurre porque los métodos constructores, así como los *getters* y *setters*, estaban hechos con base en tablas que no tenían relaciones ni listas definidas. Por eso, es necesario regenerar todo nuevamente para que se ajusten a las nuevas relaciones entre las entidades.

```java
package com.example.NuevoProyecto.Model;                                  10 17

import jakarta.persistence.*;
import java.util.List;

@Entity  14 usages  JuanFTP2233 *
public class Tipo {
    @Id  3 usages
    @GeneratedValue(strategy = GenerationType.IDENTITY)
    private int id;
    private String nombre;  3 usages
    private String descripcion;  3 usages

    @OneToMany(mappedBy = "tipo")  3 usages
    private List<Pokemon> pokemones;

    public Tipo() {  no usages  JuanFTP2233
    }

    public Tipo(int id, String nombre, String descripcion, List<Pokemon> pokemones) {  no usages  JuanFTP2233 *
        this.id = id;
        this.nombre = nombre;
        this.descripcion = descripcion;
        this.pokemones = pokemones;
    }
```

```java
package com.example.NuevoProyecto.Model;

import jakarta.persistence.*;

@Entity 18 usages JuanFTP2233 *
public class Pokemon {
    @Id 3 usages
    @GeneratedValue(strategy = GenerationType.IDENTITY)
    private int id;
    private String nombre; 3 usages
    private int nivel; 3 usages
    private String habilidad; 3 usages

    @ManyToOne 3 usages
    @JoinColumn(name = "entrenador_id")
    private Entrenador entrenador;

    @ManyToOne 3 usages
    @JoinColumn(name = "tipo_id")
    private Tipo tipo;

    public Pokemon() { no usages JuanFTP2233
    }

    public Pokemon(int id, String nombre, int nivel, String habilidad, Entrenador entrenador, Tipo tipo) { no usage
        this.id = id;
        this.nombre = nombre;
        this.nivel = nivel;
        this.habilidad = habilidad;
        this.entrenador = entrenador;
        this.tipo = tipo;
    }
```

```java
package com.example.NuevoProyecto.Model;

import jakarta.persistence.*;
import java.util.List;

@Entity 14 usages JuanFTP2233 *
public class Inventario {
    @Id 3 usages
    @GeneratedValue(strategy = GenerationType.IDENTITY)
    private int id;
    private String objeto; 3 usages
    private int cantidad; 3 usages
    private String descripcion; 3 usages

    @ManyToOne 3 usages
    @JoinColumn(name = "entrenador_id")
    private Entrenador entrenador;

    public Inventario() { no usages JuanFTP2233
    }

    public Inventario(int id, String objeto, int cantidad, String descripcion, Entrenador entrenador) { no usages
        this.id = id;
        this.objeto = objeto;
        this.cantidad = cantidad;
        this.descripcion = descripcion;
        this.entrenador = entrenador;
    }
```

```java
package com.example.NuevoProyecto.Model;

import jakarta.persistence.*;
import java.util.List;

@Entity  18 usages  JuanFTP2233 *
public class Entrenador {
    @Id  3 usages
    @GeneratedValue(strategy = GenerationType.IDENTITY)
    private int id;
    private String nombre;  3 usages
    private String region;  3 usages
    private int edad;  3 usages
    private String especialidad;  3 usages

    @OneToMany(mappedBy = "entrenador")  3 usages
    private List<Pokemon> pokemones;

    @OneToMany(mappedBy = "entrenador")  3 usages
    private List<Inventario> inventario;

    public Entrenador() {  no usages  JuanFTP2233
    }

    public Entrenador(int id, String nombre, String region, int edad, String especialidad, List<Pokemon> pokemones
        this.id = id;
        this.nombre = nombre;
        this.region = region;
        this.edad = edad;
        this.especialidad = especialidad;
        this.pokemones = pokemones;
        this.inventario = inventario;
    }
```

Una vez tengamos cotos, corremos nuestra base de datos y verificamos que todo haya salido bien

```
"C:\Program Files\Java\jdk-21\bin\java.exe" ...

  .   ____          _            __ _ _
 /\\ / ___'_ __ _ _(_)_ __  __ _ \ \ \ \
( ( )\___ | '_ | '_| | '_ \/ _` | \ \ \ \
 \\/  ___)| |_)| | | | | || (_| |  ) ) ) )
  '  |____| .__|_| |_|_| |_\__, | / / / /
 =========|_|==============|___/=/_/_/_/

 :: Spring Boot ::                (v3.4.4)

2025-04-08T20:14:28.335-05:00  INFO 4088 --- [NuevoProyecto] [  restartedMain] c.e.N.NuevoProyectoApplication          : Starting NuevoProyectoApplication using Java 21.0.6 wi
2025-04-08T20:14:28.335-05:00  INFO 4088 --- [NuevoProyecto] [  restartedMain] c.e.N.NuevoProyectoApplication          : No active profile set, falling back to 1 default profi
2025-04-08T20:14:28.393-05:00  INFO 4088 --- [NuevoProyecto] [  restartedMain] o.s.b.devtools.restart.ChangeableUrls    : The Class-Path manifest attribute in C:\Users\USUARIO\
2025-04-08T20:14:28.402-05:00  INFO 4088 --- [NuevoProyecto] [  restartedMain] .e.DevToolsPropertyDefaultsPostProcessor : Devtools property defaults active! Set 'spring.devtool
2025-04-08T20:14:28.402-05:00  INFO 4088 --- [NuevoProyecto] [  restartedMain] .e.DevToolsPropertyDefaultsPostProcessor : For additional web related logging consider setting th
2025-04-08T20:14:29.105-05:00  INFO 4088 --- [NuevoProyecto] [  restartedMain] .s.d.r.c.RepositoryConfigurationDelegate : Bootstrapping Spring Data JPA repositories in DEFAULT
2025-04-08T20:14:29.184-05:00  INFO 4088 --- [NuevoProyecto] [  restartedMain] .s.d.r.c.RepositoryConfigurationDelegate : Finished Spring Data repository scanning in 59 ms. Fou
2025-04-08T20:14:29.752-05:00  INFO 4088 --- [NuevoProyecto] [  restartedMain] o.s.b.w.embedded.tomcat.TomcatWebServer  : Tomcat initialized with port 8080 (http)
2025-04-08T20:14:29.768-05:00  INFO 4088 --- [NuevoProyecto] [  restartedMain] o.apache.catalina.core.StandardService   : Starting service [Tomcat]
2025-04-08T20:14:29.768-05:00  INFO 4088 --- [NuevoProyecto] [  restartedMain] o.apache.catalina.core.StandardEngine    : Starting Servlet engine: [Apache Tomcat/10.1.39]
2025-04-08T20:14:29.815-05:00  INFO 4088 --- [NuevoProyecto] [  restartedMain] o.a.c.c.C.[Tomcat].[localhost].[/]       : Initializing Spring embedded WebApplicationContext
2025-04-08T20:14:29.815-05:00  INFO 4088 --- [NuevoProyecto] [  restartedMain] w.s.c.ServletWebServerApplicationContext : Root WebApplicationContext: initialization completed i
2025-04-08T20:14:30.015-05:00  INFO 4088 --- [NuevoProyecto] [  restartedMain] o.hibernate.jpa.internal.util.LogHelper  : HHH000204: Processing PersistenceUnitInfo [name: defau
2025-04-08T20:14:30.078-05:00  INFO 4088 --- [NuevoProyecto] [  restartedMain] org.hibernate.Version                    : HHH000412: Hibernate ORM core version 6.6.11.Final
2025-04-08T20:14:30.100-05:00  INFO 4088 --- [NuevoProyecto] [  restartedMain] o.h.c.internal.RegionFactoryInitiator    : HHH000026: Second-level cache disabled
2025-04-08T20:14:30.354-05:00  INFO 4088 --- [NuevoProyecto] [  restartedMain] o.s.o.j.p.SpringPersistenceUnitInfo      : No LoadTimeWeaver setup: ignoring JPA class transforme
2025-04-08T20:14:30.370-05:00  INFO 4088 --- [NuevoProyecto] [  restartedMain] com.zaxxer.hikari.HikariDataSource       : HikariPool-1 - Starting...
2025-04-08T20:14:31.939-05:00  INFO 4088 --- [NuevoProyecto] [  restartedMain] com.zaxxer.hikari.pool.HikariPool        : HikariPool-1 - Added connection org.postgresql.jdbc.Pg
2025-04-08T20:14:31.939-05:00  INFO 4088 --- [NuevoProyecto] [  restartedMain] com.zaxxer.hikari.HikariDataSource       : HikariPool-1 - Start completed.
2025-04-08T20:14:32.064-05:00  WARN 4088 --- [NuevoProyecto] [  restartedMain] org.hibernate.orm.deprecation            : HHH90000025: PostgreSQLDialect does not need to be spe
2025-04-08T20:14:32.276-05:00  INFO 4088 --- [NuevoProyecto] [  restartedMain] org.hibernate.orm.connections.pooling    : HHH10001005: Database info:
    Database JDBC URL [Connecting through datasource 'HikariDataSource (HikariPool-1)']
```

Por último Visualizamos en el Supabase si quedaron relacionadas nuestras tablas.