

CURSO : Desarrollo de Aplicaciones Web I (0265)  
PROFESOR : César Enrique Santos Torres  
CICLO : Quinto  
SECCIÓN : 28257  
GRUPO : 2024331933  
FECHA : 23/11/2024 10:00am  
DURACIÓN : 2 horas

NOTA

ALUMNO (A) : Velasquez Pérez Hans Kevin

### CASO DE LABORATORIO 1 (CL1)

#### Consideraciones generales:

- El laboratorio consta de 4 partes, cada parte tiene una secuencia de pasos las cuales deberá ir acompañada (De forma obligatoria) de capturas de pantalla de lo implementado.
- Sólo debe subir este documento, con sus evidencias y respuestas en él. El código fuente de ambos proyectos debe ser subido a Github (Adjuntar links del repositorio). No se aceptará código zipeado.
- El nombre del presente archivo deberá tener la siguiente estructura: "DAWI-APELLIDOS-NOMBRES.pdf".

#### LOGRO DE LA EVALUACIÓN:

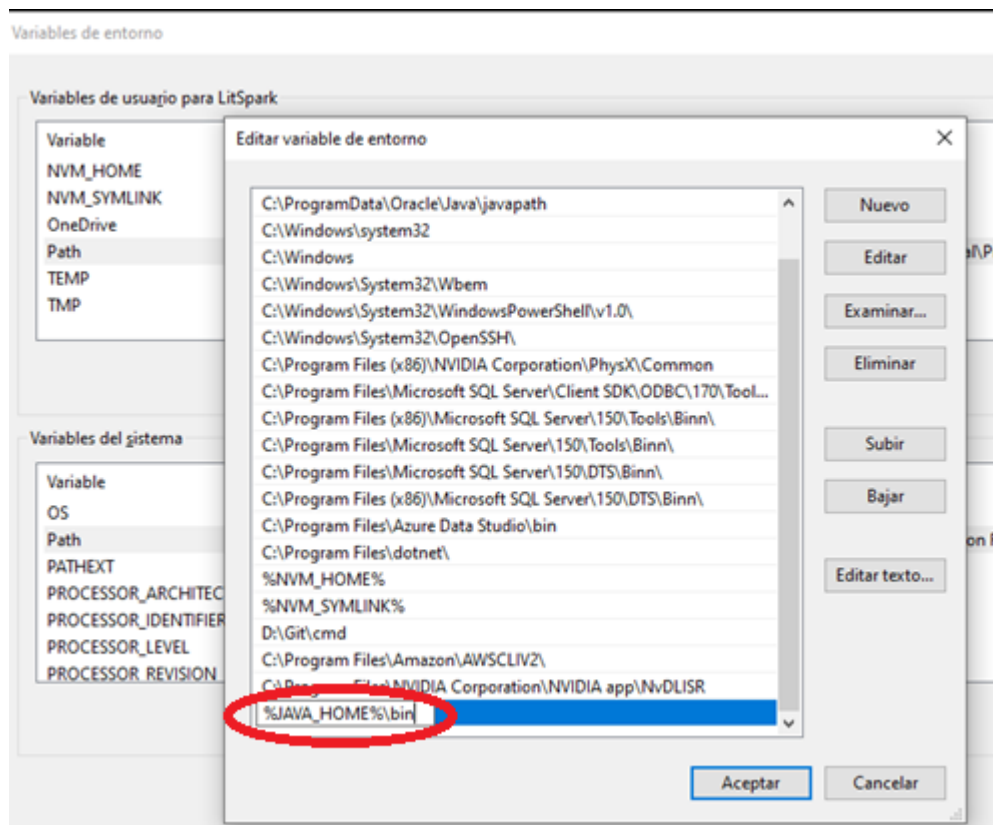
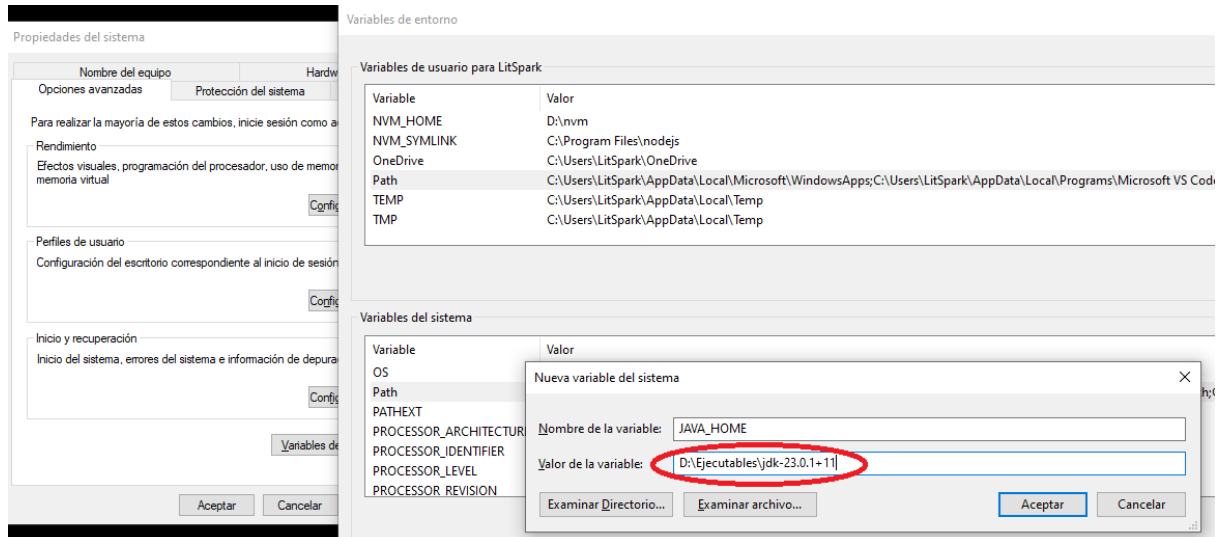
Al término de la evaluación, el alumno implementa las operaciones de mantenimiento sobre una entidad utilizando Java Persistence API.

#### CONSOLIDADO

Pregunta	Puntaje		Llenar solo en caso de Recalificación justificada	
	Máximo	Obtenido	Sustento	Puntaje
1	5			
2	5			
3	5			
4	5			
Total	20			
Nota Recalificada				

## Parte 01 Configuración básica (25%)

- Descargar **JDK versión 23** de <https://adoptium.net/es/temurin/releases/>
- Configurar variable de entorno **JAVA\_HOME** y Path
- Validar configuración Java con los siguientes comandos (Use cmd):
  - java -version
  - echo %JAVA\_HOME%
  - echo %Path%



```

C:\> Símbolo del sistema

Microsoft Windows [Versión 10.0.19045.5131]
(c) Microsoft Corporation. Todos los derechos reservados.

C:\Users\LitSpark>java -version
openjdk version "23.0.1" 2024-10-15
OpenJDK Runtime Environment Temurin-23.0.1+11 (build 23.0.1+11)
OpenJDK 64-Bit Server VM Temurin-23.0.1+11 (build 23.0.1+11, mixed mode, sharing)

C:\Users\LitSpark>echo %JAVA_HOME%
D:\Ejecutables\jdk-23.0.1+11

C:\Users\LitSpark>echo %Path%
D:\Ejecutables\jdk-23.0.1+11\bin;C:\Program Files (x86)\Common Files\Oracle\Java\java8path;C:\Program
Files\Oracle\Java\javapath;C:\ProgramData\Oracle\Java\javapath;C:\Windows\system32;C:\Windows;C:\Wind
C:\Windows\System32\WindowsPowerShell\v1.0\;C:\Windows\System32\OpenSSH\;C:\Program Files (x86)\NVIDIA
Common;C:\Program Files\Microsoft SQL Server\Client SDK\ODBC\170\Tools\Binn\;C:\Program Files (x86)\M
r\150\Tools\Binn\;C:\Program Files\Microsoft SQL Server\150\Tools\Binn\;C:\Program Files\Microsoft SQL
nn\;C:\Program Files (x86)\Microsoft SQL Server\150\DTs\Binn\;C:\Program Files\Azure Data Studio\bin;C
tnet\;D:\nvm;C:\Program Files\nodejs;D:\Git\cmd;C:\Program Files\Amazon\AWSCLIV2\;C:\Program Files\NVI

```

- Conectar al servidor MySQL usando la terminal (Use cmd):
  - Use el comando: `mysql -u root -p`

```
MySQL 9.1 Command Line Client
Enter password: **** root
Welcome to the MySQL monitor.  Commands end with ; or \g.
Your MySQL connection id is 8
Server version: 9.1.0 MySQL Community Server - GPL

Copyright (c) 2000, 2024, Oracle and/or its affiliates.

Oracle is a registered trademark of Oracle Corporation and/or its
affiliates. Other names may be trademarks of their respective
owners.

Type 'help;' or '\h' for help. Type '\c' to clear the current input statement.
mysql> La razón por la que lo hago a través del cmd de mysql es  
porque no lo reconoce a través del cmd de windows :(
```

- Restaurar bd “world” de <https://downloads.mysql.com/docs/world-db.zip> (Use cmd):
  - Use el comando: `source <ruta-archivo-world.sql>;`

A screenshot of the MySQL Command Line Client interface. The title bar reads "Selecciónar MySQL 9.1 Command Line Client". The main window displays a series of identical status messages stacked vertically: "Query OK, 1 row affected (0.00 sec)". A red arrow points from the right side towards the top-right corner of the application window.

Ya no se puede subir más, pero si se restauró la base de datos a partir de la función "source" y la ubicación de la base de datos.  
**En la siguiente imagen adjunto evidencia del uso de la base de datos.**

- Usar bd “world” y hacer un select de los primeros 20 registros de la tabla “city” (Use cmd).

```

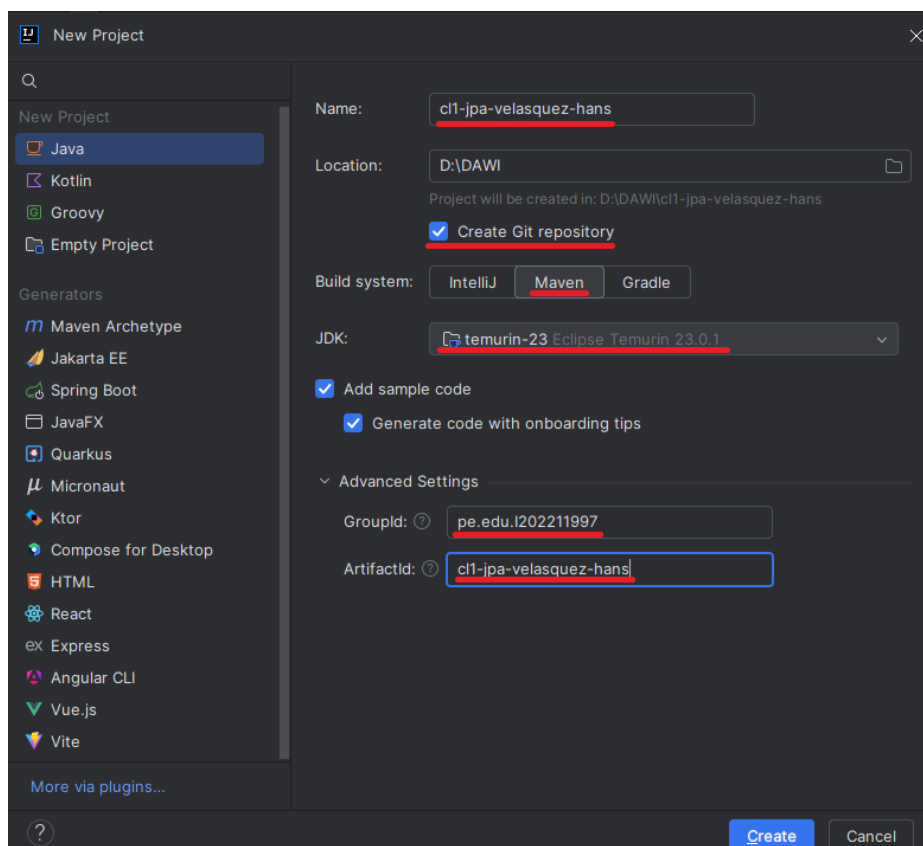
MySQL 9.1 Command Line Client
mysql> use world;
Database changed
mysql> show tables;
+-----+
| Tables_in_world |
+-----+
| city             |
| country          |
| countrylanguage  |
+-----+
3 rows in set (0.00 sec)

mysql> select * from city;
+----+-----+-----+-----+-----+
| ID | Name                | CountryCode | District      | Population |
+----+-----+-----+-----+-----+
| 1  | Kabul               | AFG         | Kabul         | 1780000    |
| 2  | Qandahar            | AFG         | Qandahar      | 237500     |
| 3  | Herat               | AFG         | Herat         | 186800     |
| 4  | Mazar-e-Sharif      | AFG         | Balkh         | 127800     |
| 5  | Amsterdam           | NLD         | Noord-Holland | 731200     |
| 6  | Rotterdam           | NLD         | Zuid-Holland  | 593321     |
| 7  | Haag                | NLD         | Zuid-Holland  | 440900     |
| 8  | Utrecht             | NLD         | Utrecht       | 234323     |
| 9  | Eindhoven           | NLD         | Noord-Brabant | 201843     |
| 10 | Tilburg             | NLD         | Noord-Brabant | 193238     |
| 11 | Groningen           | NLD         | Groningen     | 172701     |
| 12 | Breda               | NLD         | Noord-Brabant | 160398     |
| 13 | Apeldoorn           | NLD         | Gelderland    | 153491     |

```

## Parte 02 Proyecto JPA-Hibernate (25%)

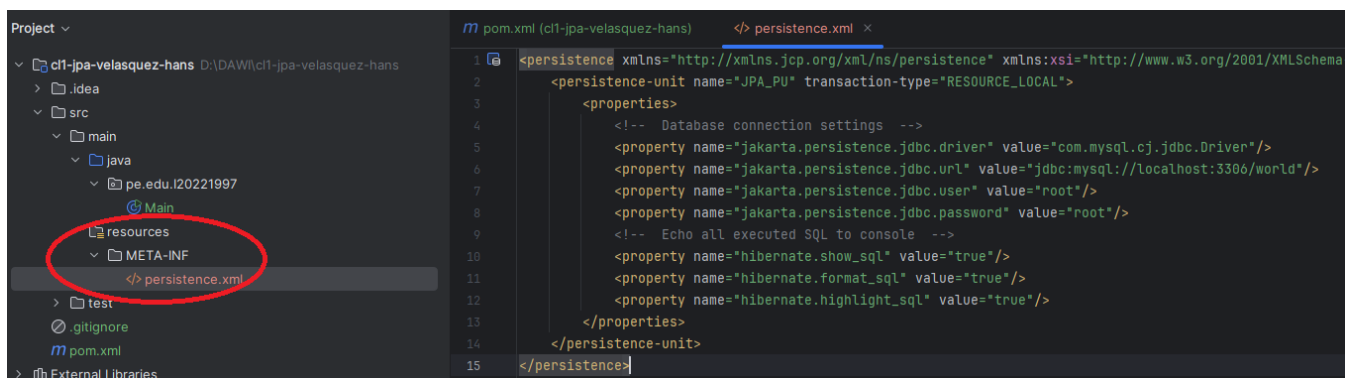
- Crear un proyecto JPA-Hibernate desde IntelliJ Idea con las siguientes características:
  - **Name:** cl1-jpa-<apellidoPaterno-primerNombre> (Use minúsculas y guión “-”)
  - **Location:** Seleccione un directorio con permisos de lectura y escritura
  - **Create Git repository:** Check
  - **Build system:** Maven
  - **JDK:** Eclipse Temurin-23
  - **Advanced Settings:**
    - **GroupId:** pe.edu.<codigoEstudiante>
    - **Artifact:** cl1-jpa-<apellidoPaterno-primerNombre>



- Configurar dependencias en el pom.xml
  - Hibernate (6.6.2.Final)
  - Driver de MySQL (9.1.0)

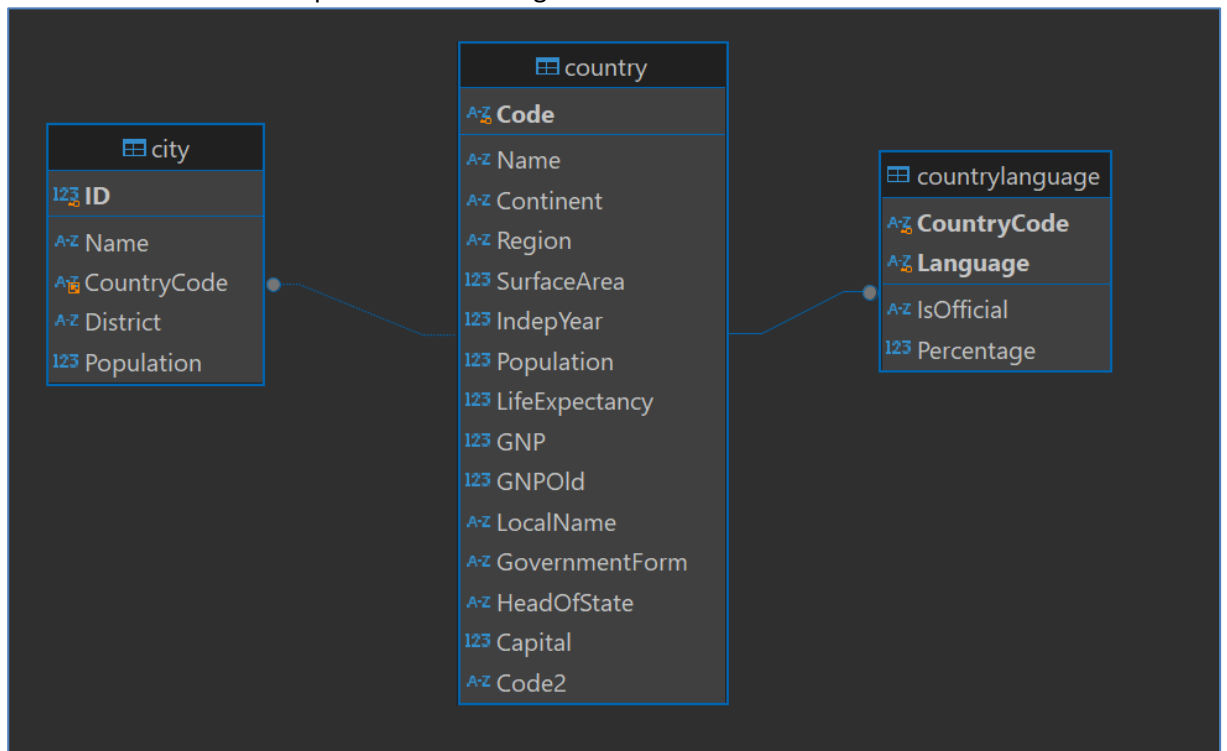
```
m pom.xml (cl1-jpa-velasquez-hans) ×
1  <?xml version="1.0" encoding="UTF-8"?>
2  <project xmlns="http://maven.apache.org/POM/4.0.0"
3      xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
4      xsi:schemaLocation="http://maven.apache.org/POM/4.0.0 http://maven.apache
5      <modelVersion>4.0.0</modelVersion>
6
7      <groupId>pe.edu.I20221997</groupId>
8      <artifactId>cl1-jpa-velasquez-hans</artifactId>
9      <version>1.0-SNAPSHOT</version>
10
11     <properties>
12         <maven.compiler.source>23</maven.compiler.source>
13         <maven.compiler.target>23</maven.compiler.target>
14         <project.build.sourceEncoding>UTF-8</project.build.sourceEncoding>
15     </properties>
16
17     <dependencies>
18         <dependency>
19             <groupId>org.hibernate.orm</groupId>
20             <artifactId>hibernate-core</artifactId>
21             <version>6.6.2.Final</version>
22         </dependency>
23
24         <dependency>
25             <groupId>com.mysql</groupId>
26             <artifactId>mysql-connector-j</artifactId>
27             <version>9.1.0</version>
28         </dependency>
29     </dependencies>
30
31 </project>
```

- Configurar unidad de persistencia en archivo “persistence.xml”



```
m pom.xml (cl1-jpa-velasquez-hans)  <> persistence.xml ×
1  <persistence xmlns="http://xmlns.jcp.org/xml/ns/persistence" xmlns:xsi="http://www.w3.org/2001/XMLSchema
2  <persistence-unit name="JPA_PU" transaction-type="RESOURCE_LOCAL">
3      <properties>
4          <!-- Database connection settings -->
5          <property name="jakarta.persistence.jdbc.driver" value="com.mysql.cj.jdbc.Driver"/>
6          <property name="jakarta.persistence.jdbc.url" value="jdbc:mysql://localhost:3306/world"/>
7          <property name="jakarta.persistence.jdbc.user" value="root"/>
8          <property name="jakarta.persistence.jdbc.password" value="root"/>
9          <!-- Echo all executed SQL to console -->
10         <property name="hibernate.show_sql" value="true"/>
11         <property name="hibernate.format_sql" value="true"/>
12         <property name="hibernate.highlight_sql" value="true"/>
13     </properties>
14 </persistence-unit>
15 </persistence>
```

- Crear las entidades correspondientes a las siguientes tablas de la bd “world”:



- Mapear las 3 entidades de forma tradicional (Sin Lombok).
- Definir la estrategia de generación correcta para los PKs.
- Considerar el mapeo de las relaciones de forma bidireccional.

```

pom.xml (cl1-jpa-velasquez-hans)  </> persistence.xml  City.java x
3  import jakarta.persistence.Entity;
4  import jakarta.persistence.Id;
5
6  @Entity no usages
7  public class City {
8
9      @Id
10     private Integer ID;
11     private String Name; 4 usages
12     private String CountryCode; 4 usages
13     private String District; 4 usages
14     private String Population; 4 usages
15
16     public City() {
17     }
18
19     public City(Integer ID, String name, String countryCode, String district, String population) {
20         this.ID = ID;
21         Name = name;
22         CountryCode = countryCode;
23         District = district;
24         Population = population;
25     }
26
27     @Override
28     public String toString() {
29         return "City{" +
30             "ID=" + ID +
31             ", Name=" + Name + '\'' +
32             ", CountryCode=" + CountryCode + '\'' +
33             ", District=" + District + '\'' +
34             ", Population=" + Population + '\'' +
35             '}';
36     }
37
38     public Integer getID() { no usages
39         return ID;
40     }
41
42     public void setID(Integer ID) { no usages
43         this.ID = ID;
44     }
  
```

```

pom.xml (cl1-jpa-velasquez-hans)  </> persistence.xml  City.java  Country.java x
9      public class Country {
31          public Country(Integer code, String name, String continent, String region,
41              this.GNPold = GNPold;
42              LocalName = localName;
43              GovernmentForm = governmentForm;
44              HeadOfState = headOfState;
45              Capital = capital;
46              Code2 = code2;
47          }
48
49          @Override
50          public String toString() {
51              return "Country{" +
52                  "Code=" + Code +
53                  ", Name=" + Name + '\'' +
54                  ", Continent=" + Continent + '\'' +
55                  ", Region=" + Region + '\'' +
56                  ", SurfaceArea=" + SurfaceArea +
57                  ", IndepYear=" + IndepYear +
58                  ", Population=" + Population +
59                  ", LifeExpectancy=" + LifeExpectancy +
60                  ", GNP=" + GNP +
61                  ", GNPold=" + GNPold +
62                  ", LocalName=" + LocalName + '\'' +
63                  ", GovernmentForm=" + GovernmentForm + '\'' +
64                  ", HeadOfState=" + HeadOfState + '\'' +
65                  ", Capital=" + Capital +
66                  ", Code2=" + Code2 + '\'' +
67                  '}';
68          }
69
70          public Integer getCode() { no usages
71              return Code;
72          }
73
74          public void setCode(Integer code) { no usages
75              Code = code;
76          }
77
78          public String getName() { no usages
79              return Name;

```

```

pom.xml (cl1-jpa-velasquez-hans)  </> persistence.xml  City.java  Country.java  Countrylanguage.java x
1      package pe.edu.I20221997.entity;
2
3      import jakarta.persistence.Entity;
4      import jakarta.persistence.Id;
5
6      @Entity no usages
7      public class Countrylanguage {
8
9          private String CountryCode; 4 usages
10         @Id
11         private String Language; 4 usages
12         private String IsOfficial; 4 usages
13         private Integer Percentage; 4 usages
14
15         public Countrylanguage() {
16         }
17
18         public Countrylanguage(String countryCode, String language, String isOfficial, Integer percentage)
19             CountryCode = countryCode;
20             Language = language;
21             IsOfficial = isOfficial;
22             Percentage = percentage;
23         }
24
25         @Override
26         public String toString() {
27             return "Countrylanguage{" +
28                 "CountryCode=" + CountryCode + '\'' +
29                 ", Language=" + Language + '\'' +
30                 ", IsOfficial=" + IsOfficial + '\'' +
31                 ", Percentage=" + Percentage +
32                 '}';
33         }
34
35         public String getCountryCode() { no usages
36             return CountryCode;
37         }
38
39         public void setCountryCode(String countryCode) { no usages
40             CountryCode = countryCode;
41         }

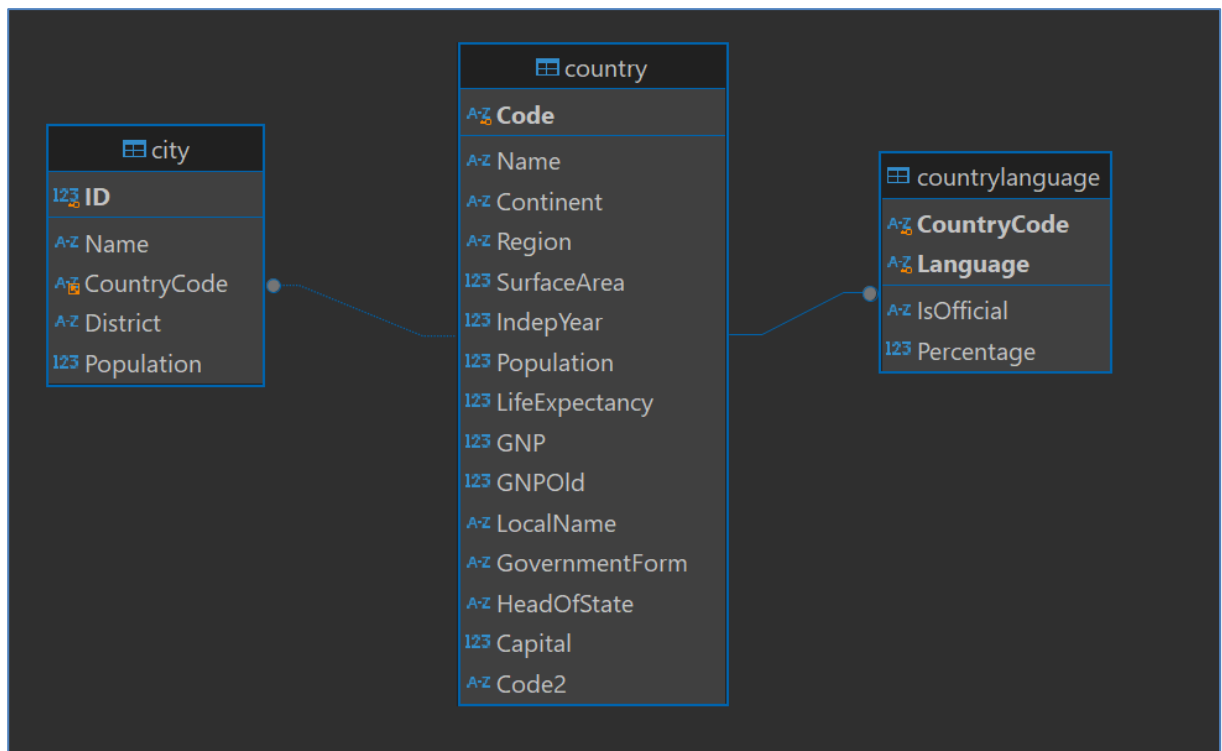
```

- Crear una clase “JPAPersist” y en ella registre un país imaginario, que tenga 3 ciudades y 2 lenguajes nativos. Sólo debe realizar un llamado al método “persist”.  
<imagen1>
- Crear una clase “JPARemove” y en ella elimine el país imaginario (Previamente creado). La eliminación debe eliminar el rastro de sus 3 ciudades y 2 lenguajes nativos. Sólo debe realizar un llamado al método “remove”. Considere, no afectar la funcionalidad de la clase “JPAPersist”.  
<imagen1>
- Crear una clase “JPAFind” y en ella realice una sola consulta a la entidad “Country” (Busque el código “PER” usando find) y en base al resultado imprima el nombre de las ciudades peruanas con población > 700k. **Deberá usar una función lambda** para discriminar el resultado.  
<imagen1>

### Parte 03 Proyecto Spring Data JPA (25%)

- Generar un proyecto con Spring Data JPA desde <https://start.spring.io/> con las siguientes características:
  - **Project:** Maven
  - **Language:** Java
  - **Spring Boot:** 3.3.5
  - **Group:** pe.edu.<codigoEstudiante>
  - **Artifact:** cl1-jpa-data-<apellidoPaterno-primerNombre>
  - **Packaging:** Jar
  - **Java:** 23
  - **Dependencies:**
    - Spring Data JPA
    - MySQL Driver
    - Lombok  
<imagen1>
- Configurar el “application.properties” con los datos de conectividad a la bd “world”.  
<imagen1>
- Crear las entidades correspondientes a las siguientes tablas (Las mismas del proyecto anterior):





- Mapear las 3 entidades usando Lombok.
- Definir la estrategia de generación correcta para los PKs.
- Considerar el mapeo de las relaciones de forma bidireccional.

<imagen1>

<imagen2>

- Crear una interfaz “CountryRepository” que extienda de “CrudRepository”.

<imagen1>

<imagen2>

- Implementar el método “run” definido en la interfaz “CommandLineRunner” desde la clase principal del proyecto de Spring Boot.

<imagen1>

<imagen2>

- Deberá implementar las siguientes 3 consultas:

- **ifPresentOrElse()**

Imprimir en la terminal los nombres de los lenguajes que se hablan en el país “ARG” (Argentina). En caso de no obtener resultado, deberá imprimir los nombres de los lenguajes del país “PER” (Perú).

<imagen2>

- **deleteAllById()**

Eliminar 2 países: “COL” y “ARG”. La eliminación deberá ser cascada y borrará sus ciudades y lenguajes correspondientes.

<imagen2>

- Volver a ejecutar la primera consulta, pues al eliminar “ARG”, deberá ejecutarse el flujo alterno. (Deberá restaurar la BD desde la terminal en cada prueba)

<imagen2>

#### Parte 04 Gestión de conexiones (25%)

- Crear una clase de configuración denominada “ConexionesConfig.java”, en ella deberá implementar una personalización del DataSource de HikariCP. Considere los siguientes valores para el pool de conexiones:
  - MaximunPoolSize: 30
  - MinimumIdle: 4
  - IdleTimeout: 4 minutos
  - ConnectionTimeout: 45 segundos

<imagen1>

<imagen2>

- Las credenciales del DataSource (Parámetros de conexión a la BD) no deben ser visibles en el código fuente. Para ello deberá crear las siguientes variables de entorno:
  - DB\_WORLD\_URL
  - DB\_WORLD\_USER
  - DB\_WORLD\_PASS
  - DB\_WORLD\_DRIVER

<imagen1>

<imagen2>

- Luego de configurar el DataSource, ¿Es necesario proporcionar las credenciales desde el archivo “application.properties”? ¿Por qué? Explique con sus propias palabras.

<Respuesta>

- ¿Por qué debo o no, configurar un JNDI en Spring Boot?

<Respuesta>