

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) : Ruth Gianella Marquina Arce

### 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:

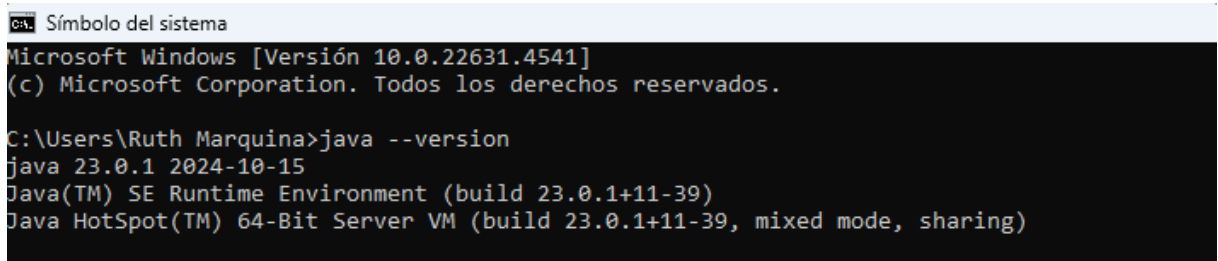
Al término de la evaluación, el alumno implementa las operaciones de mantenimiento sobre una entidad utilizando Java Persistence API.

#### 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



```
C:\> Símbolo del sistema
Microsoft Windows [Versión 10.0.22631.4541]
(c) Microsoft Corporation. Todos los derechos reservados.

C:\Users\Ruth Marquina>java --version
java 23.0.1 2024-10-15
Java(TM) SE Runtime Environment (build 23.0.1+11-39)
Java HotSpot(TM) 64-Bit Server VM (build 23.0.1+11-39, mixed mode, sharing)
```

- echo %JAVA\_HOME%



```
C:\Users\Ruth Marquina>echo %JAVA_HOME%
C:\Program Files\Java\jdk-23
```

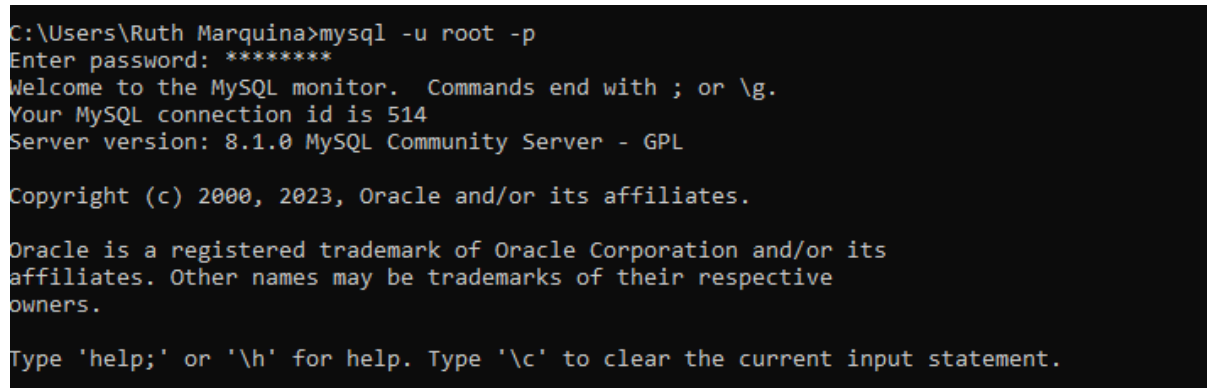
- echo %Path%

<imagen1>

<imagen2>

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

<imagen1>



```
C:\Users\Ruth Marquina>mysql -u root -p
Enter password: *****
Welcome to the MySQL monitor.  Commands end with ; or \g.
Your MySQL connection id is 514
Server version: 8.1.0 MySQL Community Server - GPL

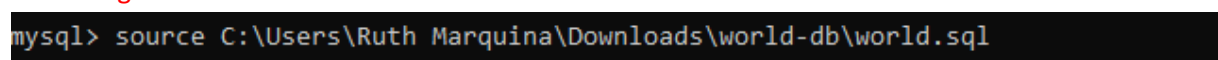
Copyright (c) 2000, 2023, Oracle and/or its affiliates.

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.
```

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

<imagen1>



```
mysql> source C:\Users\Ruth Marquina\Downloads\world-db\world.sql
```

<imagen2>

```
Simbolo del sistema - mysql -u root -p
Query OK, 1 row affected (0.00 sec)
Query OK, 1 row affected (0.00 sec)
Query OK, 1 row affected (0.00 sec)
Query OK, 1 row affected (0.00 sec)
Query OK, 0 rows affected (0.00 sec)
Query OK, 0 rows affected (0.00 sec)
Query OK, 0 rows affected (0.00 sec)
Query OK, 0 rows affected (0.00 sec)
Query OK, 0 rows affected (0.00 sec)
Query OK, 0 rows affected (0.00 sec)
Query OK, 0 rows affected (0.00 sec)
Query OK, 0 rows affected (0.00 sec)
Query OK, 0 rows affected (0.00 sec)
Query OK, 0 rows affected (0.00 sec)
Query OK, 0 rows affected (0.00 sec)
Query OK, 0 rows affected (0.00 sec)
Query OK, 0 rows affected (0.00 sec)
Query OK, 0 rows affected (0.00 sec)
Query OK, 0 rows affected (0.00 sec)
Query OK, 0 rows affected (0.00 sec)
Query OK, 0 rows affected (0.00 sec)
```

- Usar bd “world” y hacer un select de los primeros 20 registros de la tabla “city” (Use cmd).  
<imagen1>

```
mysql> use world
Database changed
mysql> select*from city LIMIT 20;
```

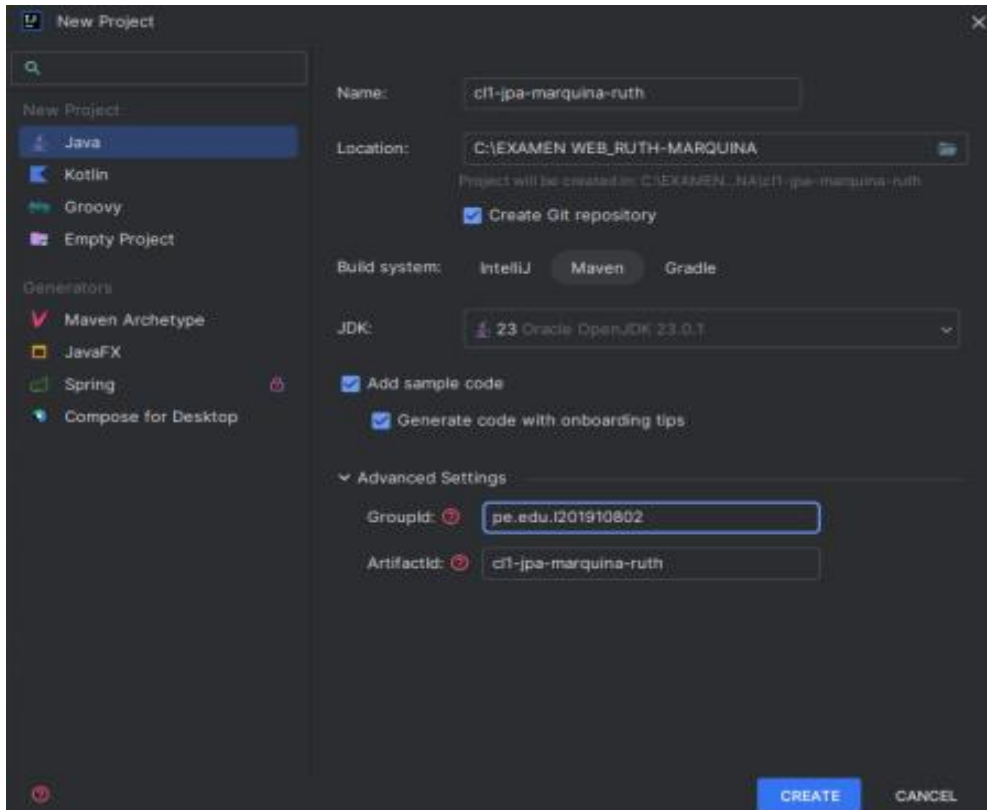
ID	Name	CountryCode	District	Population
1	Kabul	AFG	Kabul	1780000
2	Qandahar	AFG	Qandahar	237500
3	Herat	AFG	Herat	186800
4	Mazar-e-Sharif	AFG	Balkh	127800
5	Amsterdam	NLD	Noord-Holland	731200
6	Rotterdam	NLD	Zuid-Holland	593321
7	Haag	NLD	Zuid-Holland	440900
8	Utrecht	NLD	Utrecht	234323
9	Eindhoven	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
14	Nijmegen	NLD	Gelderland	152463
15	Enschede	NLD	Overijssel	149544
16	Haarlem	NLD	Noord-Holland	148772
17	Almere	NLD	Flevoland	142465
18	Arnhem	NLD	Gelderland	138020
19	Zaanstad	NLD	Noord-Holland	135621
20	's-Hertogenbosch	NLD	Noord-Brabant	129170

```
20 rows in set (0.00 sec)
```

## 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>

<imagen1>



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

<imagen1>

```

pom.xml (crl-jpa-marquina-ruth)
<?xml version="1.0" encoding="UTF-8"?>
<project xmlns="http://maven.apache.org/POM/4.0.0"
  xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
  xsi:schemaLocation="http://maven.apache.org/POM/4.0.0 http://maven.apache.org/xsd/maven-4.0.0.xsd">
  <modelVersion>4.0.0</modelVersion>

  <groupId>pe.edu.1201910802</groupId>
  <artifactId>crl-jpa-marquina-ruth</artifactId>
  <version>1.0-SNAPSHOT</version>

  <properties>
    <maven.compiler.source>25</maven.compiler.source>
    <maven.compiler.target>25</maven.compiler.target>
    <project.build.sourceEncoding>UTF-8</project.build.sourceEncoding>
  </properties>

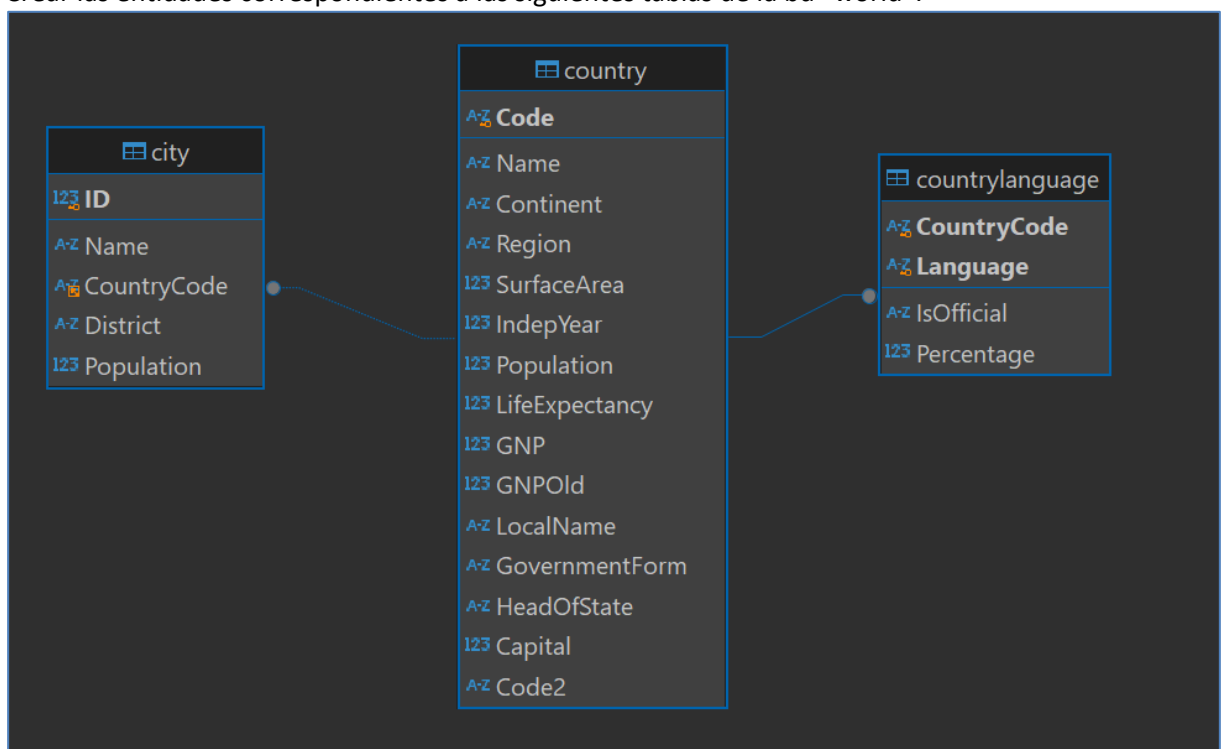
  <dependencies>
    <!-- Hibernate -->
    <dependency>
      <groupId>org.hibernate.orm</groupId>
      <artifactId>hibernate-core</artifactId>
      <version>6.6.2.Final</version>
    </dependency>

    <!-- MySQL Driver -->
    <dependency>
      <groupId>com.mysql</groupId>
      <artifactId>mysql-connector-j</artifactId>
      <version>9.1.0</version>
    </dependency>
  </dependencies>

</project>

```

- Configurar unidad de persistencia en archivo “persistence.xml”  
**<imagen1>**
- 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.

**<imagen1>**

**<imagen2>**

- 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>

The screenshot shows the Spring Initializr web application interface. The 'Project' section has 'Maven' selected. The 'Language' section has 'Java' selected. The 'Spring Boot' section has '3.3.6' selected. The 'Project Metadata' section shows the following fields: Group (pe.edu.1201910802), Artifact (cl1-jpa-data-marquina-ruth), Name (cl1-jpa-data-marquina-ruth), Description (Project for Spring Boot), Package name (pe.edu.1201910802:cl1-jpa-data-marquina-ruth), Packaging (Jar), and Java (23). The 'Dependencies' section shows 'Spring Data JPA', 'MySQL Driver', and 'Lombok' selected.

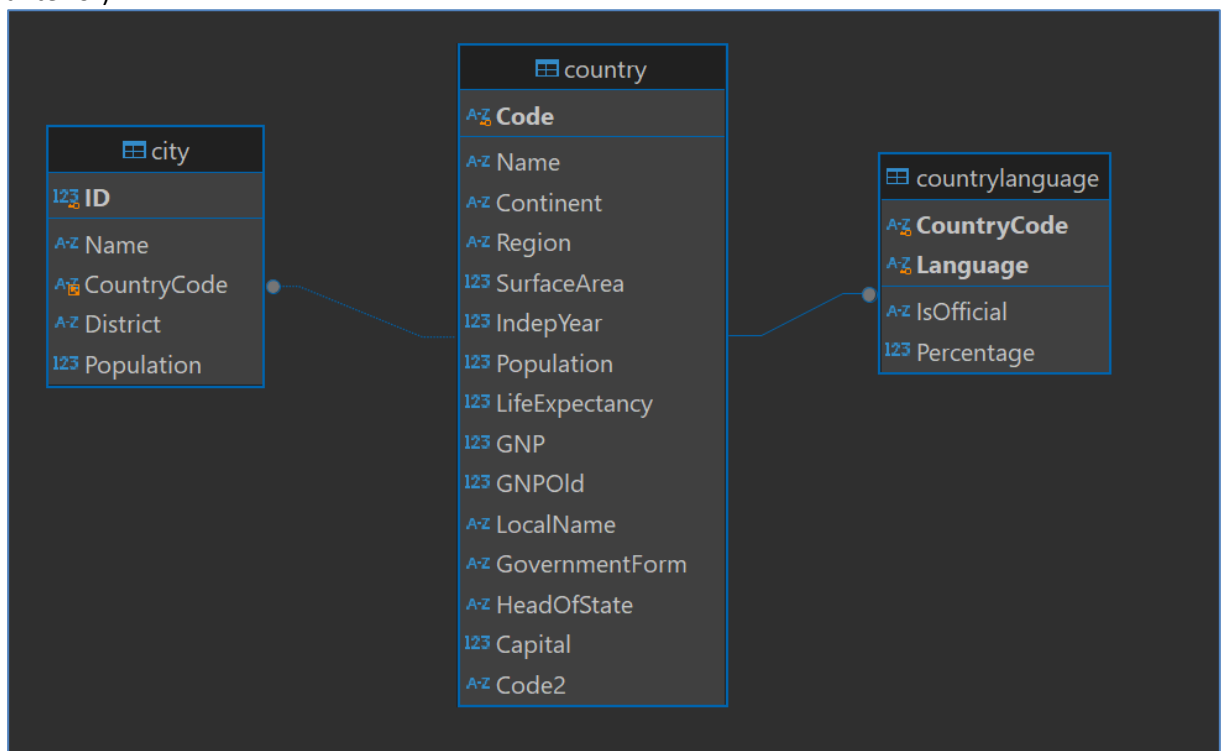
- Configurar el “application.properties” con los datos de conectividad a la bd “world”.  
<imagen1>

```

1  spring.application.name=cl1-jpa-data-marquina-ruth
2
3  spring.datasource.url=jdbc:mysql://localhost:3306/world?useSSL=false&serverTimezone=UTC
4
5  spring.datasource.username=root
6  spring.datasource.password=12345678
7
8  spring.datasource.driver-class-name=com.mysql.cj.jdbc.Driver
9
10 spring.jpa.hibernate.ddl-auto=update
11 spring.jpa.show-sql=true
12 spring.jpa.properties.hibernate.format_sql=true
13 spring.jpa.properties.hibernate.dialect=org.hibernate.dialect.MySQLDialect
14
15 spring.jpa.properties.hibernate.use_sql_comments=true
16

```

- Crear las entidades correspondientes a las siguientes tablas (Las mismas del proyecto anterior):



- Mapear las 3 entidades usando Lombok.

```

application.properties  City.java  Country.java
1 package pe.edu.I201910802.cl1_jpa_data_marquina_ruth.entity;
2
3 > import ...
4
5
6
7
8 @Entity
9 @Table(name = "city")
10 @AllArgsConstructor
11 @NoArgsConstructor
12 @Data
13 public class City {
14     @Id
15     @GeneratedValue(strategy = GenerationType.IDENTITY)
16     private Integer id;
17
18     @Column(length = 35, nullable = false)
19     private String name;
20
21     @Column(name = "CountryCode", columnDefinition = "CHAR(3)")
22     private String countryCode;
23
24     @Column(length = 20, nullable = false)
25     private String district;
26

```

```

application.properties  City.java  Country.java
1 package pe.edu.I201910802.cl1_jpa_data_marquina_ruth.entity;
2
3 > import ...
4
5
6
7
8 @Entity
9 @Table(name = "country")
10 @AllArgsConstructor
11 @NoArgsConstructor
12 @Data
13 public class Country {
14     @Id
15     @Column(name = "Code", columnDefinition = "CHAR(3)")
16     private String code;
17
18     @Column(length = 52, nullable = false)
19     private String name;
20
21     @Column(length = 13, nullable = false)
22     private String continent;
23
24     @Column(length = 26, nullable = false)
25     private String region;
26
27     @Column(name = "SurfaceArea", nullable = false)
28

```



```
application.properties  Country.java  CountryLanguage.java
1 package pe.edu.I201910802.cl1_jpa_data_marquina_ruth.entity;
2
3 > import ...
4
5
6
7
8
9
10 @Entity
11 @Table(name = "country")
12 @AllArgsConstructor
13 @NoArgsConstructor
14 @Data
15 public class Country {
16     @Id
17     @Column(name = "Code", columnDefinition = "CHAR(3)")
18     private String code;
19
20     @Column(length = 52, nullable = false)
21     private String name;
22
23     @Column(length = 13, nullable = false)
24     private String continent;
25
26     @Column(length = 26, nullable = false)
27     private String region;
```

- Definir la estrategia de generación correcta para los PKs.
- Considerar el mapeo de las relaciones de forma bidireccional.

<imagen1>

```
@OneToMany(mappedBy = "country")
private List<City> cities;

@OneToMany(mappedBy = "country")
private List<CountryLanguage> languages;
}
```

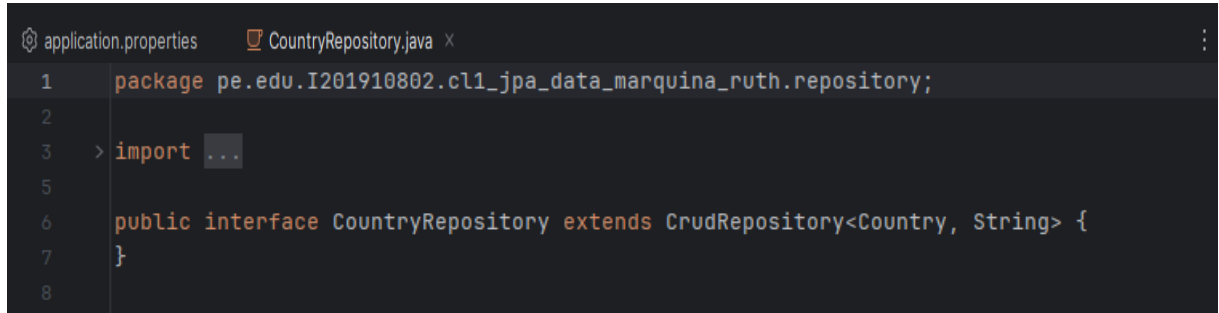
<imagen2>

```
@ManyToOne
@JoinColumn(name = "CountryCode", referencedColumnName = "Code", insertable = false, updatable = false)
private Country country;
}
```

```
@ManyToOne
@JoinColumn(name = "CountryCode", referencedColumnName = "Code", insertable = false, updatable = false)
private Country country;
}
```

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

<imagen1>



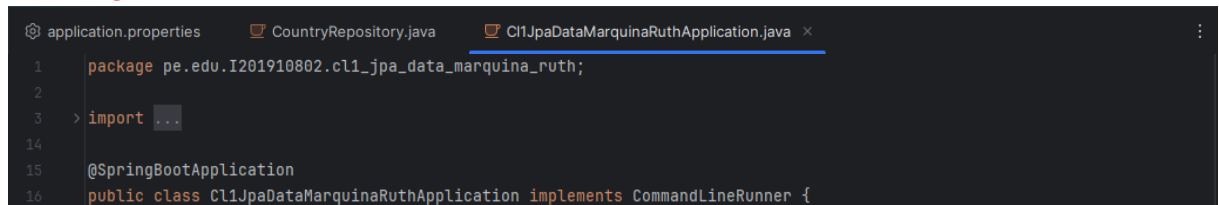
```

1 package pe.edu.I201910802.cl1_jpa_data_marquina_ruth.repository;
2
3 > import ...
4
5
6 public interface CountryRepository extends CrudRepository<Country, String> {
7 }
8

```

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

<imagen1>



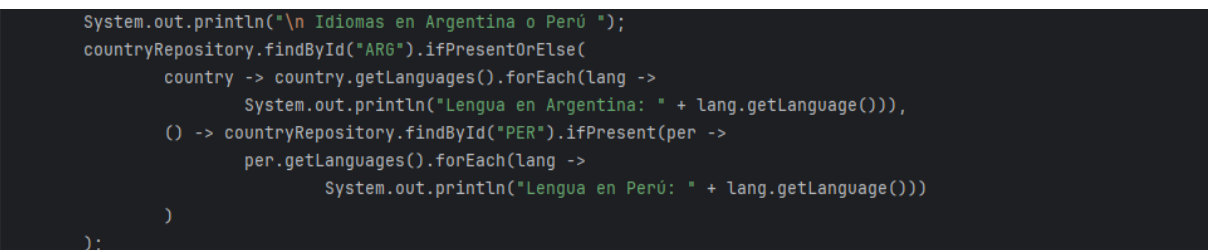
```

1 package pe.edu.I201910802.cl1_jpa_data_marquina_ruth;
2
3 > import ...
4
5
6 @SpringBootApplication
7 public class Cl1JpaDataMarquinaRuthApplication implements CommandLineRunner {
8
9
10
11
12
13
14
15
16
17
18
19
20
21
22
23
24
25
26
27
28
29
30
31
32
33
34
35
36
37
38
39
40
41
42
43
44
45
46
47
48
49
50
51
52
53
54
55
56
57
58
59
60
61
62
63
64
65
66
67
68
69
70
71
72
73
74
75
76
77
78
79
80
81
82
83
84
85
86
87
88
89
90
91
92
93
94
95
96
97
98
99
100

```

- 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>



```

System.out.println("\n Idiomas en Argentina o Perú ");
countryRepository.findById("ARG").ifPresentOrElse(
    country -> country.getLanguages().forEach(lang ->
        System.out.println("Lengua en Argentina: " + lang.getLanguage())),
    () -> countryRepository.findById("PER").ifPresent(per ->
        per.getLanguages().forEach(lang ->
            System.out.println("Lengua en Perú: " + lang.getLanguage()))
    )
);

```

- **deleteAllById()**

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

<imagen2>

```
System.out.println("\n Eliminamos Colombia y Argentina ");
List<String> codesList = List.of("COL", "ARG");
String formattedCodes = codesList.stream()
    .map(code -> "'" + code + "'")
    .collect(Collectors.joining(", "));

try {
    entityManager.getTransaction().begin();

    String[] queries = {
        "DELETE FROM city WHERE CountryCode IN (" + formattedCodes + ")",
        "DELETE FROM countrylanguage WHERE CountryCode IN (" + formattedCodes + ")",
        "DELETE FROM country WHERE Code IN (" + formattedCodes + ")"
    };

    for (String query : queries) {
        entityManager.createNativeQuery(query).executeUpdate();
    }

    entityManager.getTransaction().commit();
    System.out.println("Eliminación exitosa");
} catch (Exception e) {
    if (entityManager.getTransaction().isActive()) {
        entityManager.getTransaction().rollback();
    }
    System.err.println("Error: " + e.getMessage());
    e.printStackTrace();
}
}
```

- 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>

Variables de usuario para Ruth Marquina	
Variable	Valor
DB_WORLD_DRIVER	com.mysql.cj.jdbc.Driver
DB_WORLD_PASS	12345
DB_WORLD_URL	jdbc:mysql://localhost:3306/world
DB_WORLD_USER	root

```

application.properties  ClassConfig.java
package pe.edu.I201910802.cl1_jpa_data_marquina_ruth.config;

import ...

@Configuration
public class ClassConfig {
    @Value("${DB_WORLD_URL}")
    private String dbWorldUrl;

    @Value("${DB_WORLD_USER}")
    private String dbWorldUser;

    @Value("${DB_WORLD_PASS}")
    private String dbWorldPass;

    @Value("${DB_WORLD_DRIVER}")
    private String dbWorldDriver;

    @Bean
    public HikariDataSource hikariDataSource() {
        HikariConfig config = new HikariConfig();

        config.setJdbcUrl(dbWorldUrl);
        config.setUsername(dbWorldUser);
        config.setPassword(dbWorldPass);
        config.setDriverClassName(dbWorldDriver);

        config.setMaximumPoolSize(30);
        config.setMinimumIdle(4);
        config.setIdleTimeout(240000);
        config.setConnectionTimeout(45000);

        return new HikariDataSource(config);
    }
}

```

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

<Respuesta>

No, no es estrictamente necesario si ya se han configurado las credenciales como variables de entorno. Spring Boot permite acceder a estas variables directamente en tiempo de ejecución. Esto ofrece una mayor seguridad al evitar que las credenciales sensibles queden expuestas en el código fuente o en los archivos de configuración que podrían ser subidos accidentalmente a repositorios públicos. Configurar las variables de entorno también facilita la gestión y el

cambio de credenciales en diferentes entornos (desarrollo, pruebas, producción) sin modificar el archivo application.properties.

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

<Respuesta>

Configurar un JNDI es útil si trabajas con servidores de aplicaciones grandes que lo usan para gestionar recursos como bases de datos. Pero en proyectos simples con Spring Boot, suele ser innecesario, ya que configurar directamente el DataSource es más fácil y directo.