

CURSO : Desarrollo de Aplicaciones Web I (0265)  
PROFESOR : César Enrique Santos Torres  
CICLO : Quinto  
SECCIÓN : 26750  
GRUPO : 2024336362  
FECHA : 19/11/2024  
DURACIÓN : 3 horas

NOTA

ALUMNO (A) : BLAS GALICIA RAMIRO

### 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:

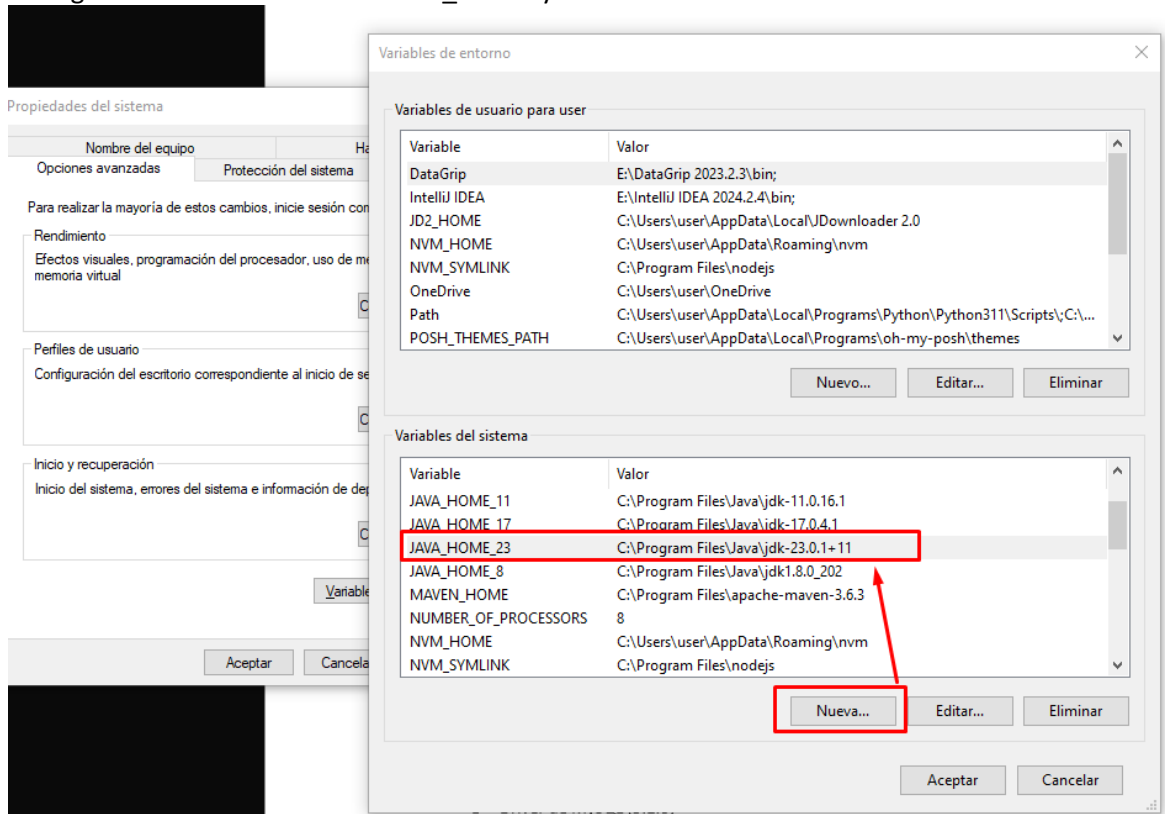
Al término de la evaluación, el alumno implementa las operaciones de mantenimiento sobre una entidad utilizando Java Persistence API.

#### 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





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

```
Program Files\modejs;E:\intellic3\IDEA 2024.2.4\bin;;E:\data\git\2023.2.3\bin;;E:\webstorm\2023.2.4\bin;;E:\pycharm\2024.2.4\bin
```

```
C:\Users\user>javac -version
javac 23.0.1

C:\Users\user>mysql -u root -p
Enter password: *****
Welcome to the MySQL monitor.  Commands end with ; or \g.
Your MySQL connection id is 9
Server version: 8.0.30 MySQL Community Server - GPL

Copyright (c) 2000, 2022, 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> |
```

- Restaurar bd “world” de <https://downloads.mysql.com/docs/world-db.zip> (Use cmd):
  - Use el comando: `source <ruta-archivo-world.sql>;`
- Usar bd “world” y hacer un select de los primeros 20 registros de la tabla “city” (Use cmd).

```
Windows Cmd - mysql -u root
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, 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, 1 warning (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)
mysql> use world;
Database changed
mysql>
```

```
Windows Cmd - mysql -u root x + v
Database changed
mysql> show tables;
+-----+
| Tables_in_world |
+-----+
| city             |
| country          |
| countrylanguage  |
+-----+
3 rows in set (0.00 sec)

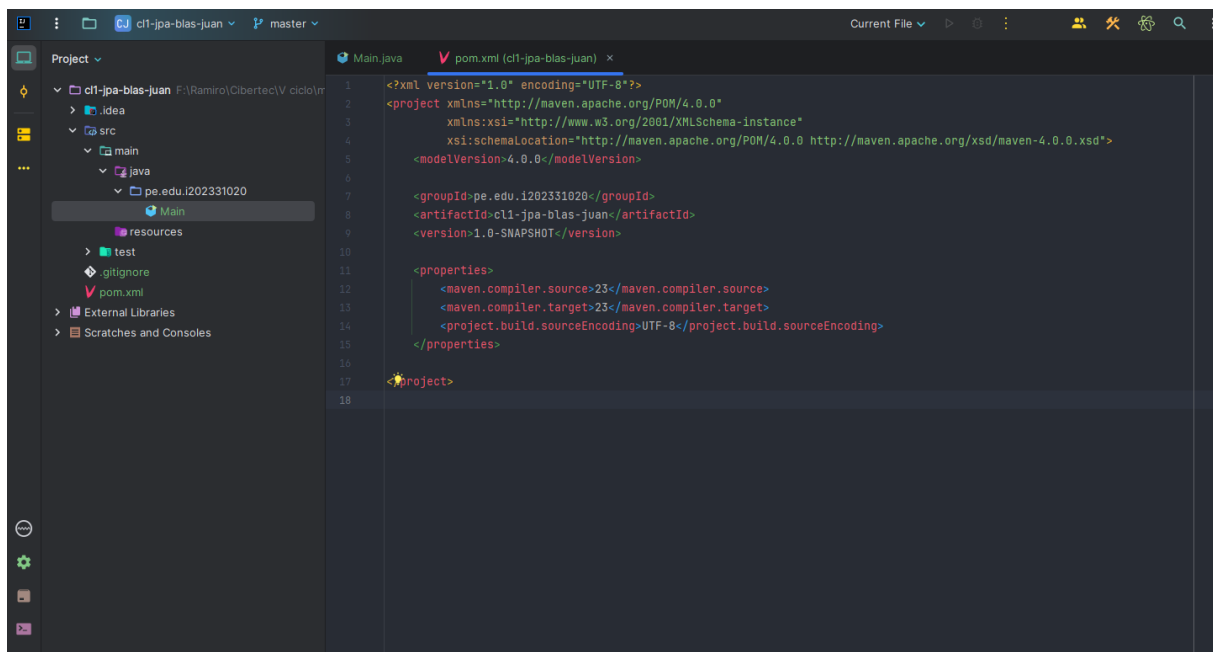
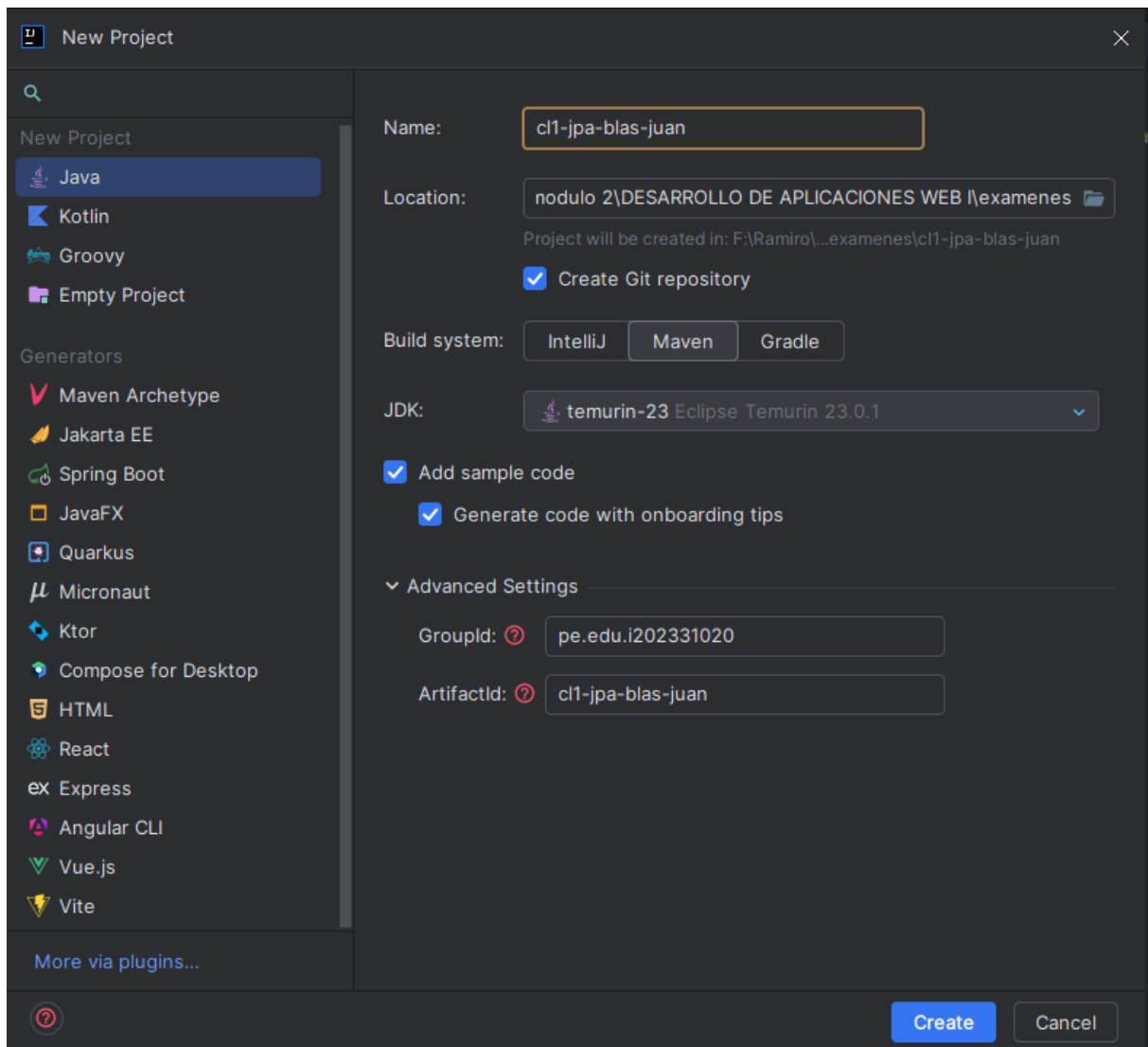
mysql> select * from city limit 20;
```

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>



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

```

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.org/xsd/maven-4.0.0.xsd">
5      <modelVersion>4.0.0</modelVersion>
6
7      <groupId>pe.edu.i202331020</groupId>
8      <artifactId>cl1-jpa-blas-juan</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>
32

```

```

PS F:\Ramiro\Cibertec\V ciclo\modulo 2\DESARROLLO DE APLICACIONES WEB I\exámenes\cl1-jpa-blas-juan> git config --list
diff.astextplain.textconv=astextplain
filter.lfs.clean=git-lfs clean -- %f
filter.lfs.smudge=git-lfs smudge -- %f
filter.lfs.process=git-lfs filter-process
filter.lfs.required=true
http.sslbackend=openssl
http.sslcainfo=C:/Program Files/Git/mingw64/ssl/certs/ca-bundle.crt
core.autocrlf=true
core.fsmonitor=true
core.symbols=false
pull.rebase=false
credential.helper=manager-core
credential.https://dev.azure.com.usehttppath=true
init.defaultBranch=master
user.name=Ramiro Blas
user.email=ramiro060694@gmail.com
core.editor=vim
color.ui=true

```



```
Project ▾ Main.java pom.xml (cl1-jpa-blas-juan) ×
Terminal Local × + ↗
PS F:\Ramiro\Cibertec\V ciclo\modulo 2\DESARROLLO DE APLICACIONES WEB I\exámenes\cl1-jpa-blas-juan> git status
On branch master
nothing to commit, working tree clean
PS F:\Ramiro\Cibertec\V ciclo\modulo 2\DESARROLLO DE APLICACIONES WEB I\exámenes\cl1-jpa-blas-juan> git log
commit 6cc0d773b2e3d80075f1458c66f55ec216759e11 (HEAD -> master)
Author: Ramiro Blas <ramiro060694@gmail.com>
Date:   Wed Nov 20 02:22:55 2024 -0500

    feat: configuracion inicial del proyecto
PS F:\Ramiro\Cibertec\V ciclo\modulo 2\DESARROLLO DE APLICACIONES WEB I\exámenes\cl1-jpa-blas-juan>
```

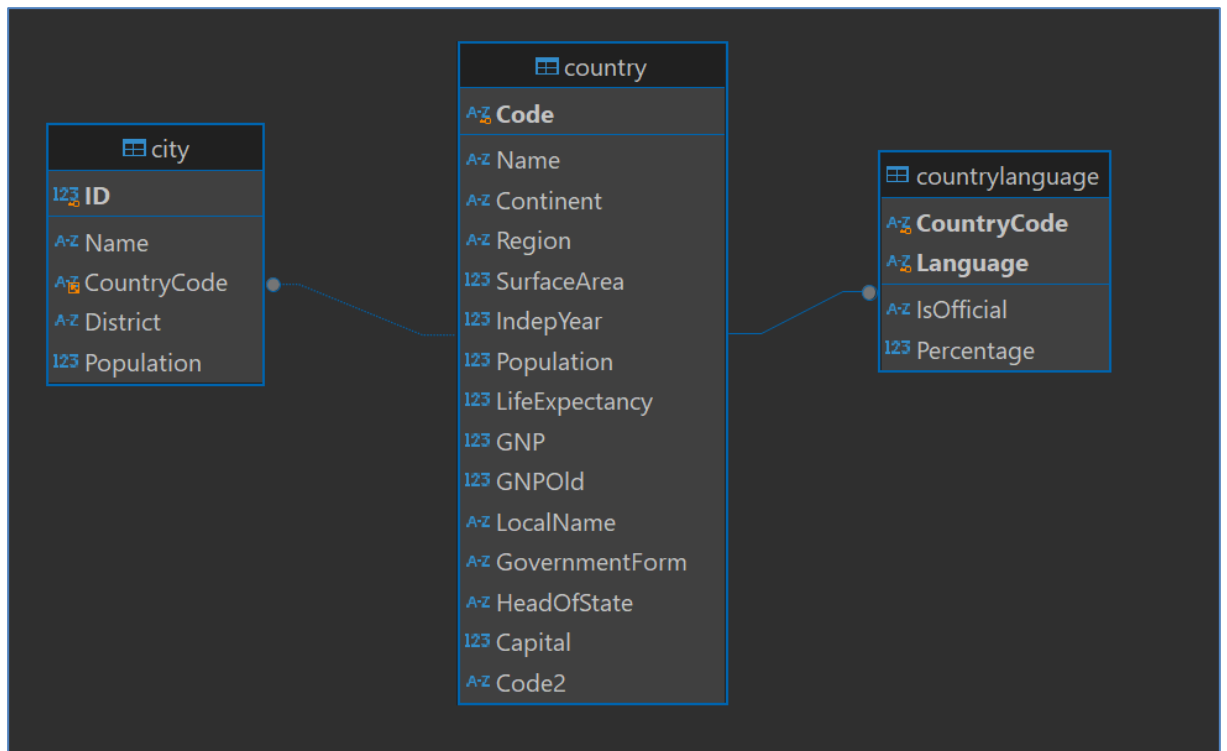
```
Project ▾ Main.java pom.xml (cl1-jpa-blas-juan) ×
Terminal Local × + ↗
PS F:\Ramiro\Cibertec\V ciclo\modulo 2\DESARROLLO DE APLICACIONES WEB I\exámenes\cl1-jpa-blas-juan> git push -u origin master
Enumerating objects: 16, done.
Counting objects: 100% (16/16), done.
Delta compression using up to 8 threads
Compressing objects: 100% (10/10), done.
Writing objects: 100% (16/16), 2.54 KiB | 868.00 KiB/s, done.
Total 16 (delta 0), reused 0 (delta 0), pack-reused 0
To https://github.com/RamiroBlas/cl1-jpa-blas-juan.git
 * [new branch]      master -> master
branch 'master' set up to track 'origin/master'.
PS F:\Ramiro\Cibertec\V ciclo\modulo 2\DESARROLLO DE APLICACIONES WEB I\exámenes\cl1-jpa-blas-juan>
```

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

```
ure/cl1-jpa-hibernate ▾ Current File ▾ ▶ ⌕ ⚙ ⚡ 🔍 ⋮
Main.java pom.xml (cl1-jpa-blas-juan) persistence.xml ×
1 <?xml version="1.0" encoding="UTF-8"?>
2 <persistence version="2.1" xmlns="http://xmlns.jcp.org/xml/ns/persistence"
3     xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
4     xsi:schemaLocation="http://xmlns.jcp.org/xml/ns/persistence http://xmlns.jcp.org/xml/ns/persistence/persistence_2.1.xs
5
6     <persistence-unit name="world" transaction-type="RESOURCE_LOCAL">
7
8         <properties>
9
10             <!-- Database connection settings -->
11             <property name="jakarta.persistence.jdbc.driver" value="com.mysql.cj.jdbc.Driver"/>
12             <property name="jakarta.persistence.jdbc.url" value="jdbc:mysql://localhost:3306/world"/>
13             <property name="jakarta.persistence.jdbc.user" value="root"/>
14             <property name="jakarta.persistence.jdbc.password" value="060694"/>
15
16             <!-- Echo all executed SQL to console -->
17             <property name="hibernate.show_sql" value="true"/>
18             <property name="hibernate.format_sql" value="true"/>
19             <property name="hibernate.highlight_sql" value="true"/>
20
21         </properties>
22     </persistence-unit>
23
24 </persistence>
25
```



- 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.
- 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”.



```

    where
        id=?
    [Hibernate]
    delete
    from
        city
    where
        id=?
    [Hibernate]
    delete
    from
        countrylanguage
    where
        CountryCode=?
        and language=?
    [Hibernate]
    delete
    from
        countrylanguage
    where
        CountryCode=?
        and language=?
    [Hibernate]
    delete
    from
        country
    where
        code=?
    Process finished with exit code 0
  
```

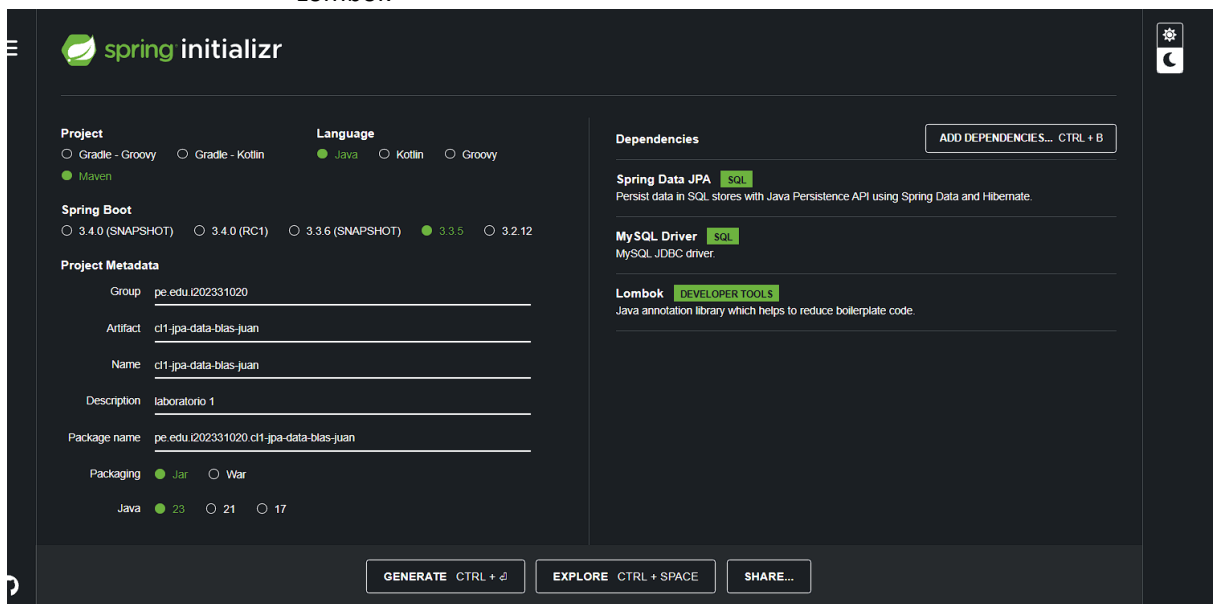
- Crear una clase “JPAFind” y en ella realice una sola consulta a la entidad “Country” (Busque el código “PER” usando find) y en base al resultado imprima el nombre de las ciudades peruanas con población > 700k. Deberá usar una función lambda para discriminar el resultado.

```

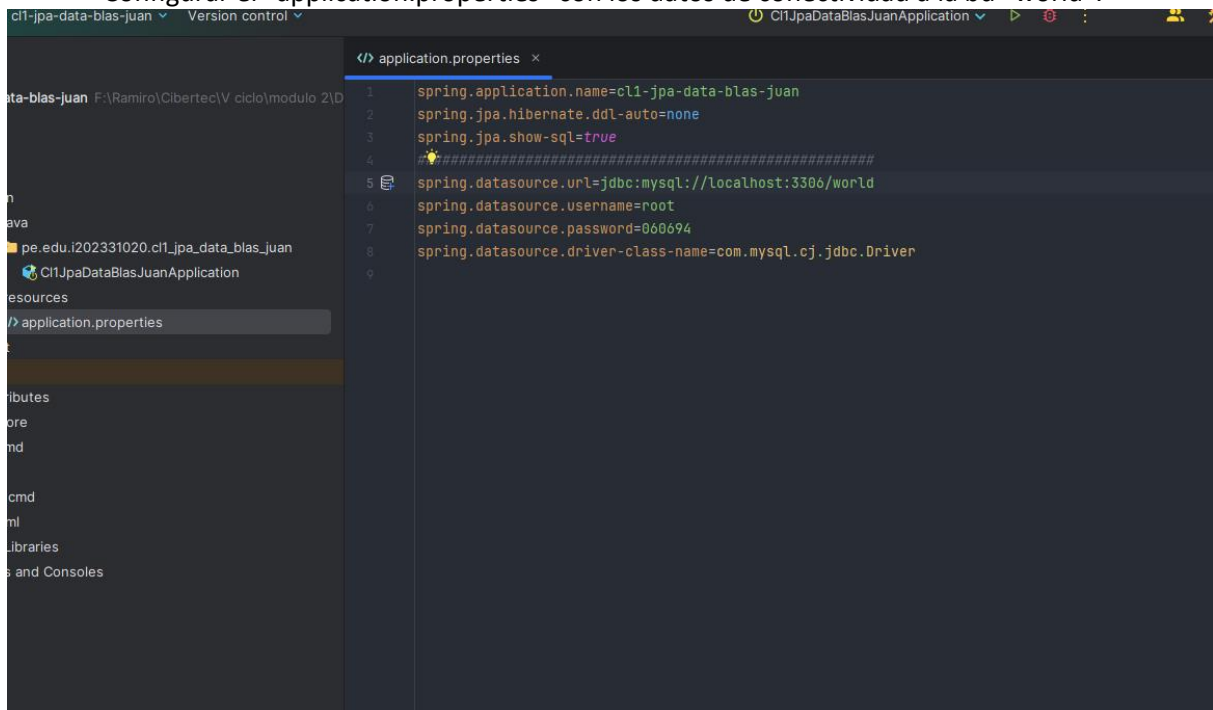
    c1_0.gnp,
    c1_0.gnpOld,
    c1_0.governmentForm,
    c1_0.headOfState,
    c1_0.indepYear,
    c1_0.lifeExpectancy,
    c1_0.localName,
    c1_0.name,
    c1_0.population,
    c1_0.region,
    c1_0.surfaceArea
    from
        country c1_0
    where
        c1_0.code=?
    [Hibernate]
    select
        c1_0.CountryCode,
        c1_0.id,
        c1_0.district,
        c1_0.name,
        c1_0.population
    from
        city c1_0
    where
        c1_0.CountryCode=?
    ciudad es: Lima
    ciudad es: Arequipa
    Process finished with exit code 0
  
```

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

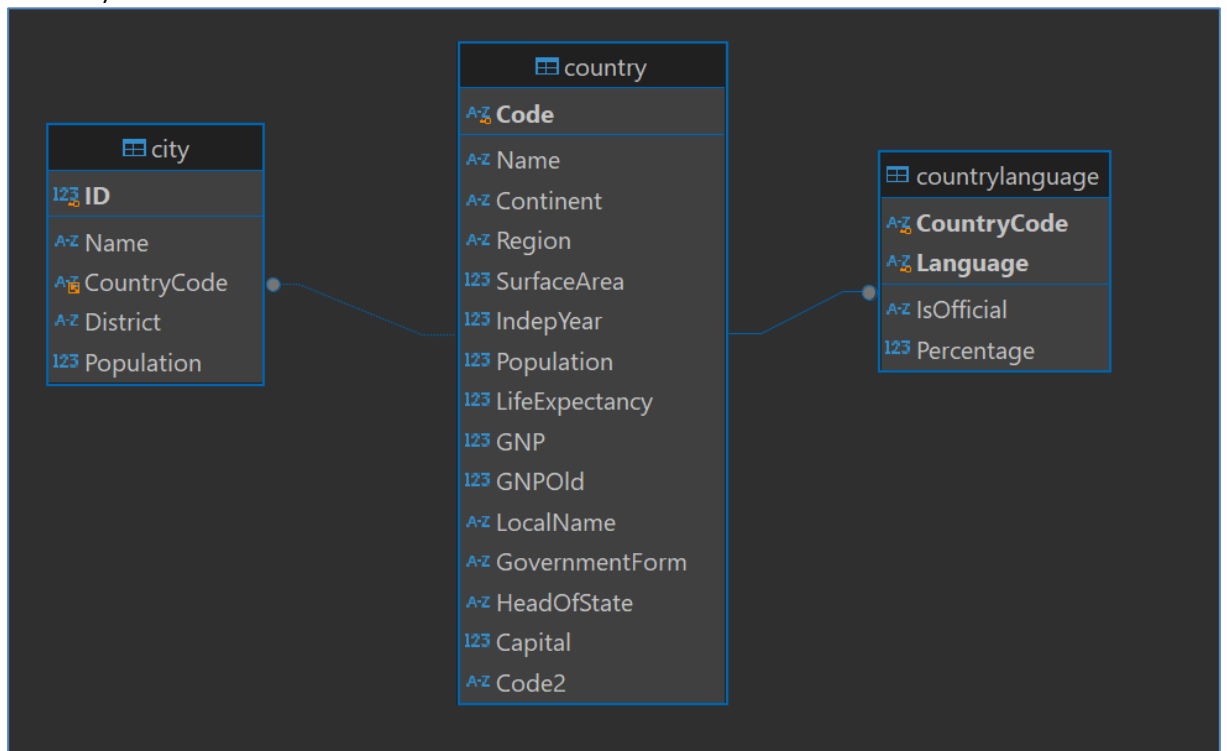
- Generar un proyecto con Spring Data JPA desde <https://start.spring.io/> con las siguientes características:
  - Project: Maven
  - Language: Java
  - Spring Boot: 3.3.5
  - Group: pe.edu.<codigoEstudiante>
  - Artifact: cl1-jpa-data-<apellidoPaterno-primerNombre>
  - Packaging: Jar
  - Java: 23
  - Dependencies:
    - Spring Data JPA
    - MySQL Driver
    - Lombok



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



- 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.

```

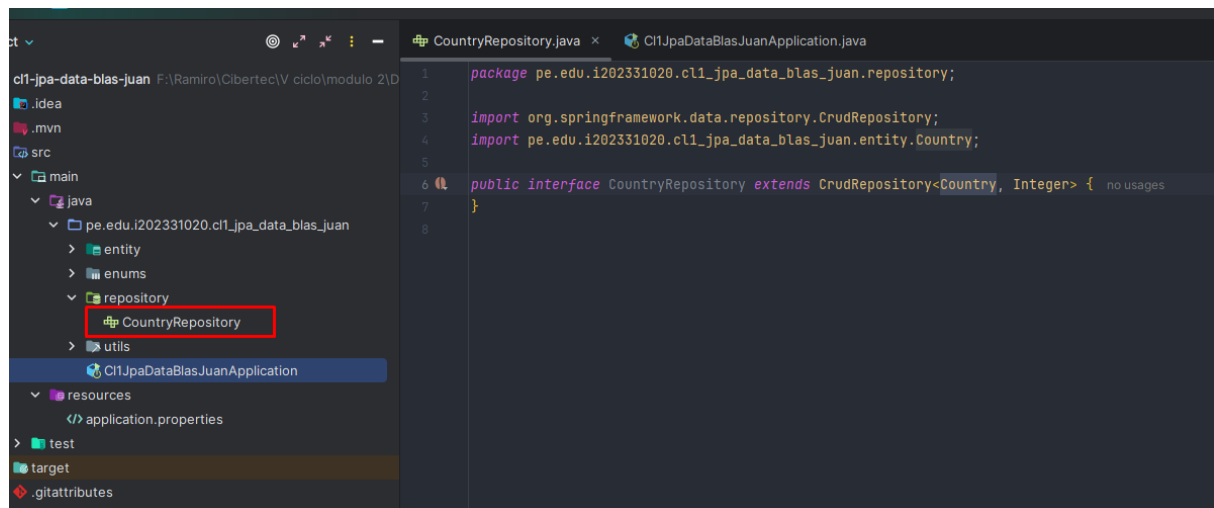
City.java
1 package pe.edu.i202331020.cl1_jpa_da;
2
3 import jakarta.persistence.*;
4 import lombok.*;
5
6 @Entity
7 @Table(name = "city")
8 @Setter
9 @Getter
10 @NoArgsConstructor
11 @AllArgsConstructor
12 @ToString
13 public class City {
14     @Id
15     @GeneratedValue(strategy = GenerationType.IDENTITY)
16     private int id;
17     private String name;
18     private String district;
19     private int population;
20     @ManyToOne(cascade = {CascadeType.PERSIST, CascadeType.REMOVE})
21     @JoinColumn(name = "CountryCode")
22     private Country country;
23 }

Country.java
1 import java.util.List;
2
3 @Entity
4 @Table(name = "country")
5 @Setter
6 @Getter
7 @NoArgsConstructor
8 @AllArgsConstructor
9 @ToString
10 public class Country {
11     @Id
12     private String code;
13     private String name;
14     @Convert(converter = ContinentEnumConverter.class)
15     private ContinentEnum continent;
16     private String region;
17     private Double surfaceArea;
18     private Integer indepYear;
19     private Integer population;
20     private Double lifeExpectancy;
21     private Double gnp;
22     private Double gnpOld;
23     private String localName;
24     private String governmentForm;
25     private String headOfState;
26     private Integer capital;
27     private String code2;
28     @OneToMany(mappedBy = "country", cascade = {CascadeType.PERSIST, CascadeType.REMOVE})
29     private List<City> cities;
30     @OneToMany(mappedBy = "country", cascade = {CascadeType.PERSIST, CascadeType.REMOVE})
31     private List<CountryLanguage> languages;
32 }

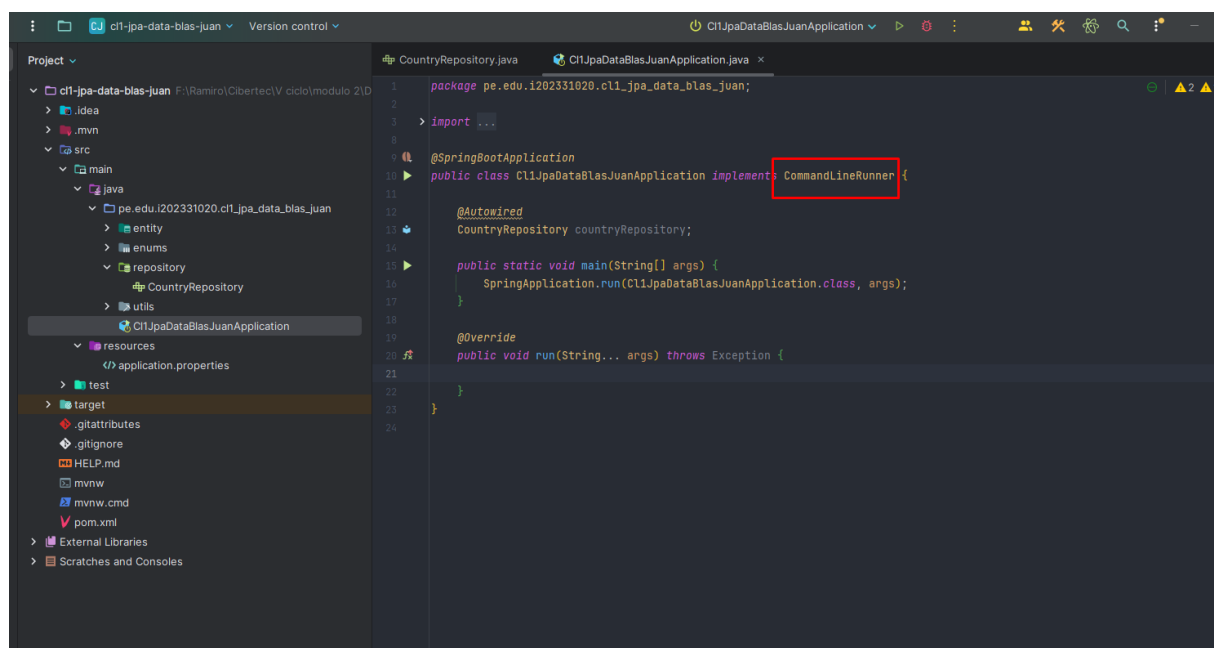
CountryLanguage.java
1 package pe.edu.i202331020.cl1_jpa_data_bl;
2
3 import jakarta.persistence.*;
4 import lombok.*;
5 import pe.edu.i202331020.cl1_jpa_data_blas_juan.enums.IsOfficialEnum;
6
7 @Entity
8 @Table(name = "countrylanguage")
9 @Setter
10 @Getter
11 @NoArgsConstructor
12 @AllArgsConstructor
13 @ToString
14 public class CountryLanguage {
15     @Id
16     private String language;
17     @Enumerated(EnumType.STRING)
18     private IsOfficialEnum isOfficial;
19     private Double percentage;
20     @ManyToOne(cascade = {CascadeType.PERSIST, CascadeType.REMOVE})
21     @JoinColumn(name = "CountryCode")
22     private Country country;
23 }

```

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



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



- 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).





- ConnectionTimeout: 45 segundos

```

14     @Value("${DB_WORLD_USER}")
15     private String dbWorldUser;
16     @Value("${DB_WORLD_PASS}")
17     private String dbWorldPass;
18     @Value("${DB_WORLD_DRIVER}")
19     private String dbWorldDriver;
20
21     @Bean new *
22     public HikariDataSource hikariDataSource() {
23         HikariConfig config = new HikariConfig();
24
25         config.setJdbcUrl(dbWorldUrl);
26         config.setUsername(dbWorldUser);
27         config.setPassword(dbWorldPass);
28         config.setDriverClassName(dbWorldDriver);
29
30         config.setMaximumPoolSize(30);
31         config.setMinimumIdle(4);
32         config.setIdleTimeout(240000);
33         config.setConnectionTimeout(45000);
34
35         System.out.println("##### HikariCP initialized #####");
36
37         return new HikariDataSource(config);
38     }
39
40

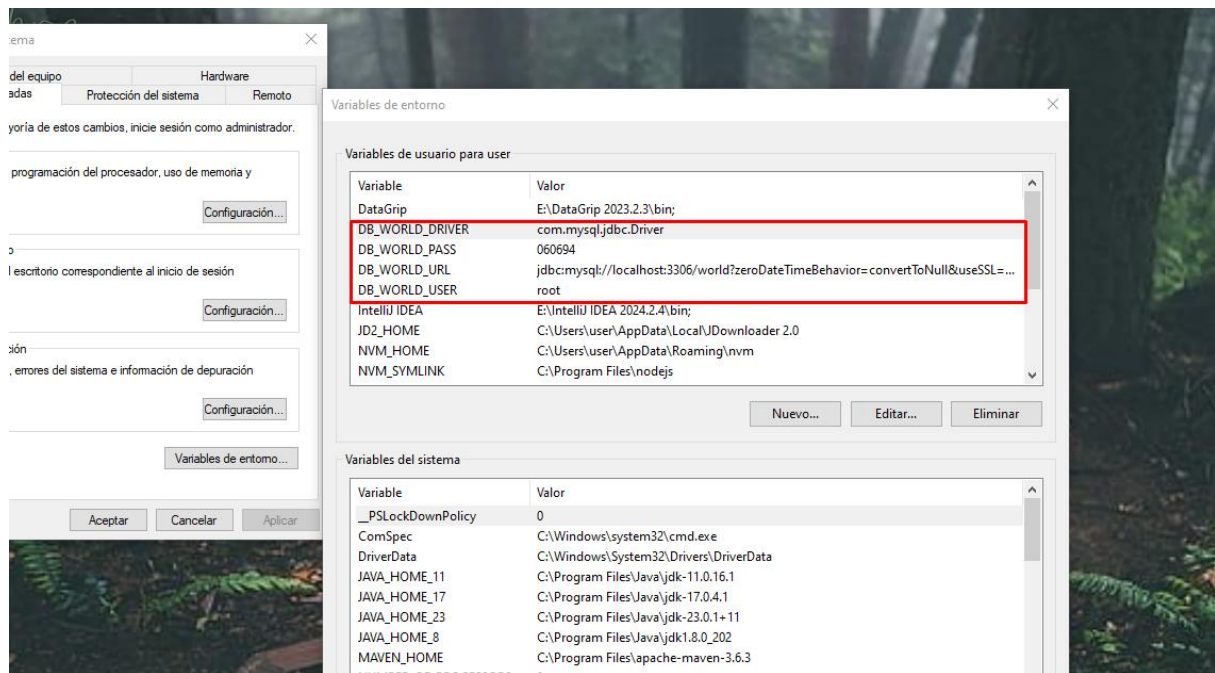
```

- 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

```

5     import org.springframework.beans.factory.annotation.Value;
6     import org.springframework.context.annotation.Bean;
7     import org.springframework.context.annotation.Configuration;
8
9     @Configuration new *
10     public class ConexionesConfig {
11
12         @Value("${DB_WORLD_URL}")
13         private String dbWorldUrl;
14         @Value("${DB_WORLD_USER}")
15         private String dbWorldUser;
16         @Value("${DB_WORLD_PASS}")
17         private String dbWorldPass;
18         @Value("${DB_WORLD_DRIVER}")
19         private String dbWorldDriver;
20
21     @Bean new *
22     public HikariDataSource hikariDataSource() {
23         HikariConfig config = new HikariConfig();
24
25         config.setJdbcUrl(dbWorldUrl);
26         config.setUsername(dbWorldUser);
27         config.setPassword(dbWorldPass);
28         config.setDriverClassName(dbWorldDriver);
29
30         config.setMaximumPoolSize(30);
31         config.setMinimumIdle(4);
32         config.setIdleTimeout(240000);
33         config.setConnectionTimeout(45000);
34
35         System.out.println("##### HikariCP initialized #####");
36
37         return new HikariDataSource(config);
38     }
39
40

```



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

. ya no es necesario brindar las credenciales en el properties porque esta configurado desde las variables de entorno, para mayor seguridad de datos.

Repositorio1 : <https://github.com/RamiroBlas/cl1-jpa-blas-juan>

Repositorio 2: <https://github.com/RamiroBlas/cl1-jpa-data-blas-juan>

