

APELLIDOS: Damborenea Terreros
NOMBRE: Miguel
DNI: 79137172Z

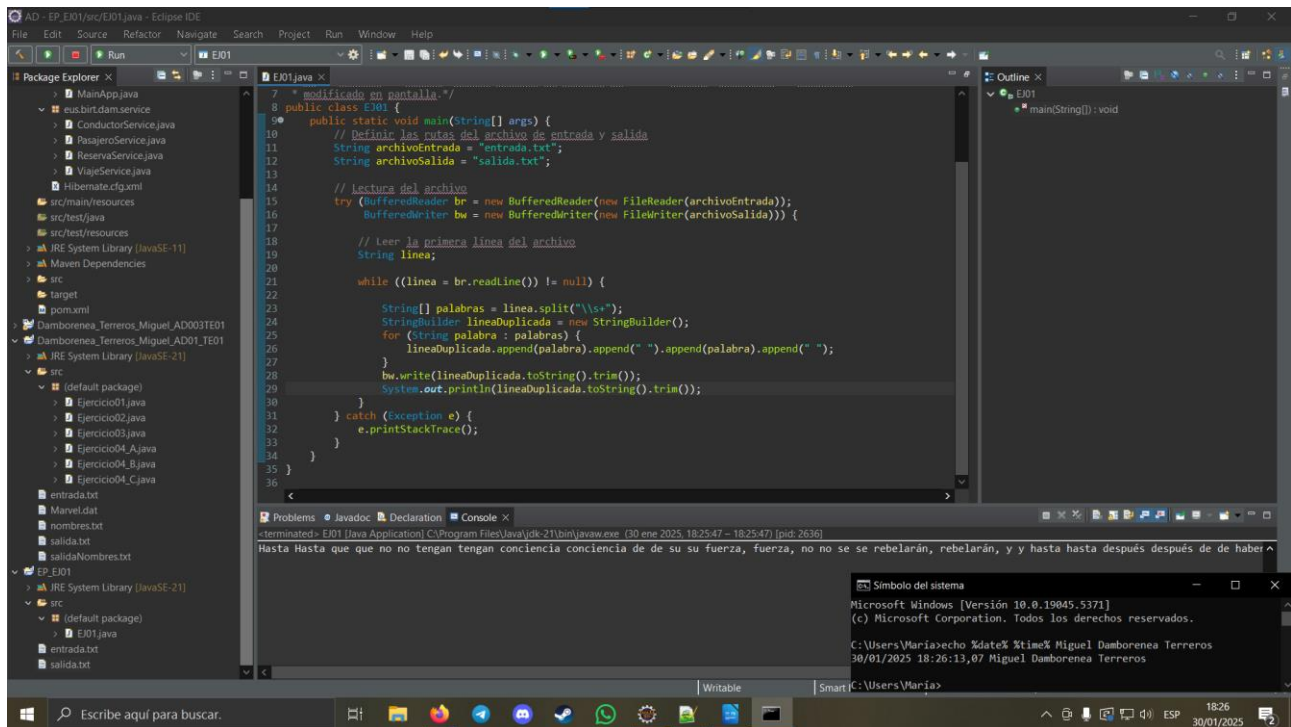
OBSERVACIONES

- Duración: **2,5 horas**
- El examen consta de 4 **preguntas**.
- Antes de comenzar, lee detenidamente todas las preguntas.
- Es posible utilizar como ayuda todo el código que hemos utilizado a lo largo del curso. Puedes navegar por el aula virtual para realizar los dos primeros ejercicios
- Pega en este mismo documento la **captura de pantalla** que muestra que el ejercicio se ha realizado correctamente. **Si no funciona no adjuntes captura**. Añade también el código del ejercicio 2.
- Sube este documento al aula virtual.
- Por otro lado, sube a GitHub los proyectos de Ficheros, JDBC e Hibernate. y copia el enlace al final del documento.

1. Ficheros (FileReader, FileWriter) (2 puntos) (adjunta captura de ejecución correcta)

Realiza una clase Java llamada *DuplicarPalabrasTexto*. Esta clase leerá el contenido de un archivo de texto y creará otro archivo donde cada palabra del texto original se duplicará (por ejemplo, "Hola mundo" se transformará en "Hola Hola mundo mundo"). Si el archivo no existe, deberá mostrar un mensaje de error. Además, imprimirá el contenido modificado en pantalla.

```
Hasta Hasta que que no no tengan tengan conciencia conciencia de de su su fuerza, fuerza, n
Ese Ese es es el el problema. problema.
```



```

7  * modificado en pantalla.*/
8  public class EJ01 {
9      public static void main(String[] args) {
10         // Definir las rutas del archivo de entrada y salida
11         String archivoEntrada = "entrada.txt";
12         String archivoSalida = "salida.txt";
13
14         // Lectura del archivo
15         try (BufferedReader br = new BufferedReader(new FileReader(archivoEntrada));
16             BufferedWriter bw = new BufferedWriter(new FileWriter(archivoSalida))) {
17
18             // Leer la primera línea del archivo
19             String linea;
20
21             while ((linea = br.readLine()) != null) {
22
23                 String[] palabras = linea.split("\\s+");
24                 StringBuilder lineaDuplicada = new StringBuilder();
25                 for (String palabra : palabras) {
26                     lineaDuplicada.append(palabra).append(" ");
27                 }
28                 bw.write(lineaDuplicada.toString().trim());
29                 System.out.println(lineaDuplicada.toString().trim());
30             }
31         } catch (Exception e) {
32             e.printStackTrace();
33         }
34     }
35 }
36

```

Problems • Javadoc • Declaration • Console X

<terminated> EJ01 [Java Application] C:\Program Files\Java\jdk-21\bin\javaw.exe (30 ene 2025, 18:25:47 - 18:25:47) [pid: 2636]

Hasta Hasta que que no no tengan tengan conciencia conciencia de de su su fuerza, fuerza, no no se se rebelarán, rebelarán, y y hasta hasta después después de de haber

Símbolo del sistema

Microsoft Windows [Versión 10.0.19045.5371]
(c) Microsoft Corporation. Todos los derechos reservados.

C:\Users\María>echo %date% %time% Miguel Damborenea Terreros
30/01/2025 18:26:13,07 Miguel Damborenea Terreros

Smart IC: Users\María>

2. **BASES DE DATOS OBJETO RELACIONALES (2,5 puntos)** (adjunta captura de ejecución correcta y código en cada punto)

1. Modela un **objeto** llamado **película** para almacenar sus características (título, director, duración en minutos). Incluye una función member llamada **calcular_duracion_horas** que devolverá la duración de la película en horas con un decimal.

CREATE TYPE pelicula AS OBJECT (

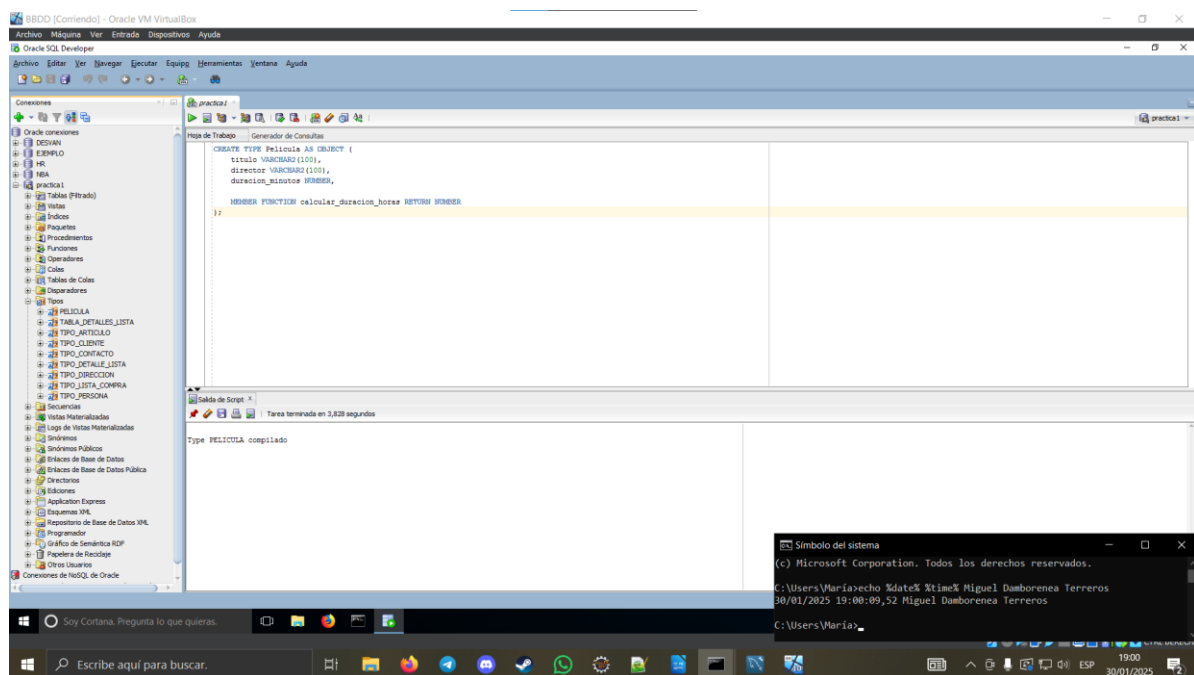
 titulo VARCHAR2(100),

 director VARCHAR2(100),

 duracion_minutos NUMBER,

 MEMBER FUNCTION calcular_duracion_horas RETURN NUMBER

);



2. Define la función **calcular_duracion_horas** para convertir la duración de minutos a horas.

CREATE TYPE pelicula AS OBJECT (

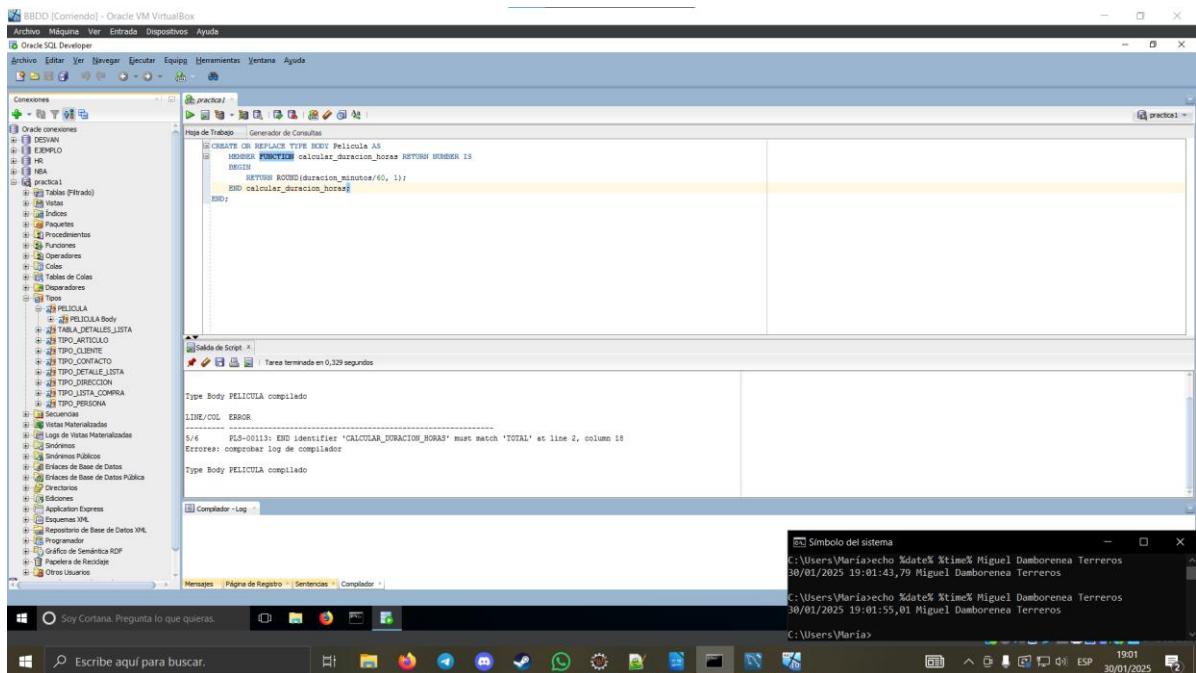
 titulo VARCHAR2(100),

 director VARCHAR2(100),

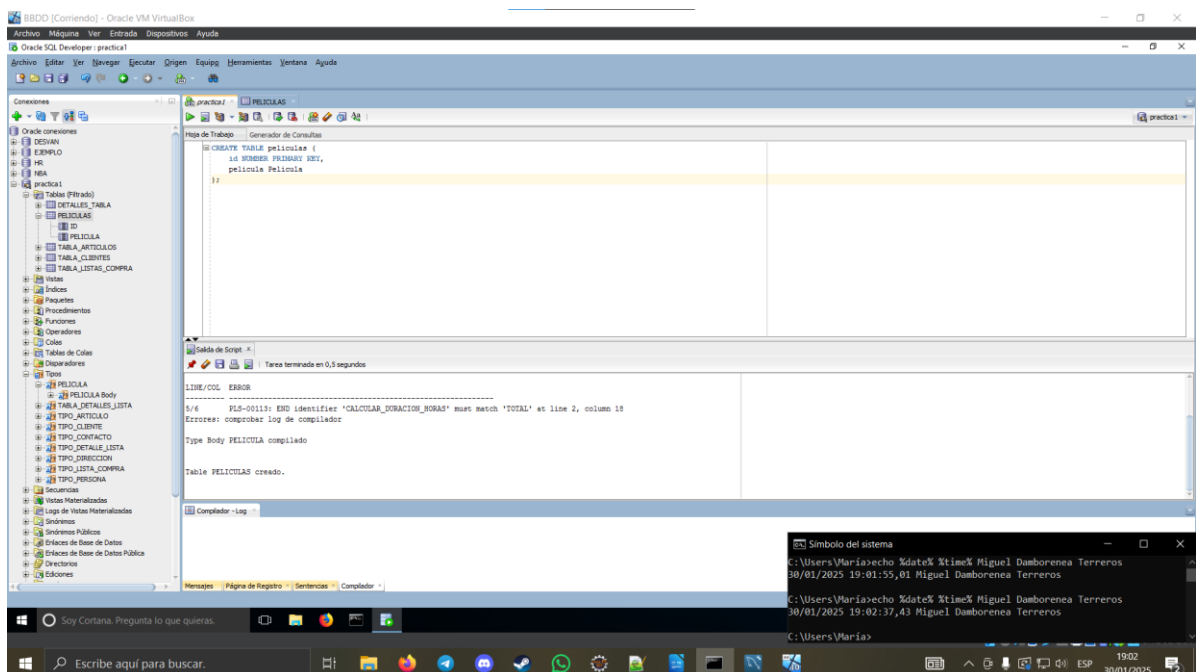
 duracion_minutos NUMBER,

 MEMBER FUNCTION calcular_duracion_horas RETURN NUMBER

);

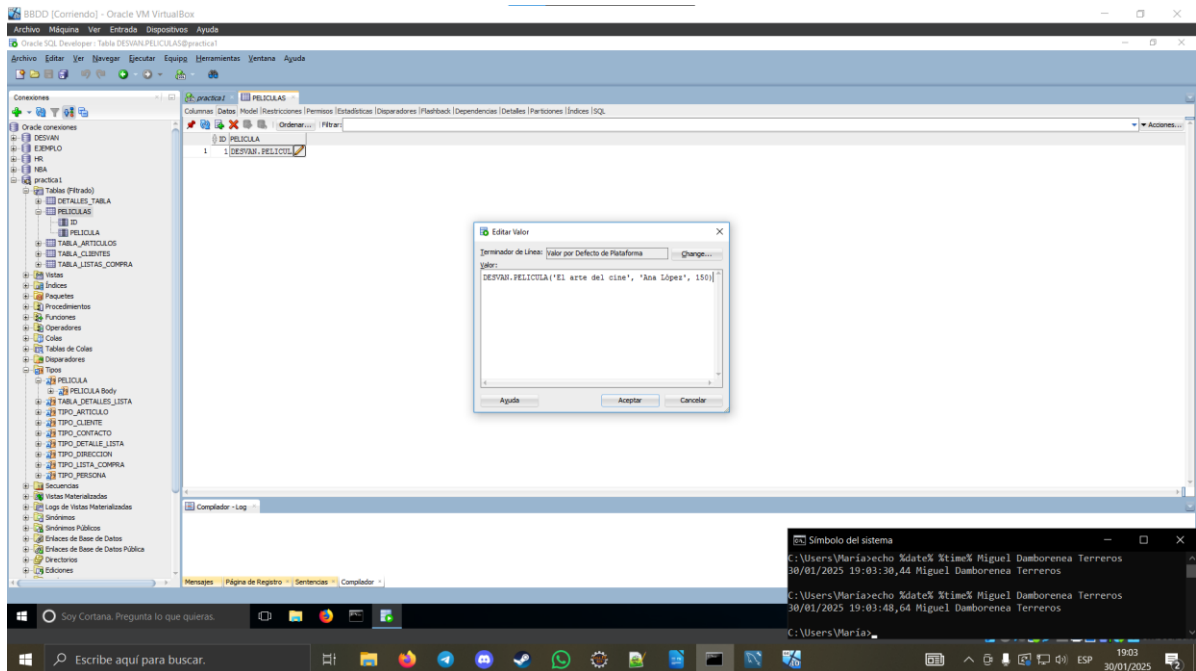


3. Crea una **tabla** para almacenar películas con un id de tipo NUMBER y un campo película del tipo tipo_pelicula.



4. Inserta una película con título "El arte del cine", director "Ana López" y duración de 150 minutos.

INSERT INTO Peliculas (id, pelicula) VALUES (1, Pelicula('El arte del cine', 'Ana López', 150));



5. Crea un programa en PL/SQL que invoque a la función **calcular_duracion_horas** para obtener la duración de la película en horas.
 - a. Recupera a una variable tipo `pelicula` una película.
 - b. Invoca la función.
 - c. Muestra la duración con `DBMS_OUTPUT.PUT_LINE`.

DECLARE

p Pelicula;

duracion_en_horas NUMBER;

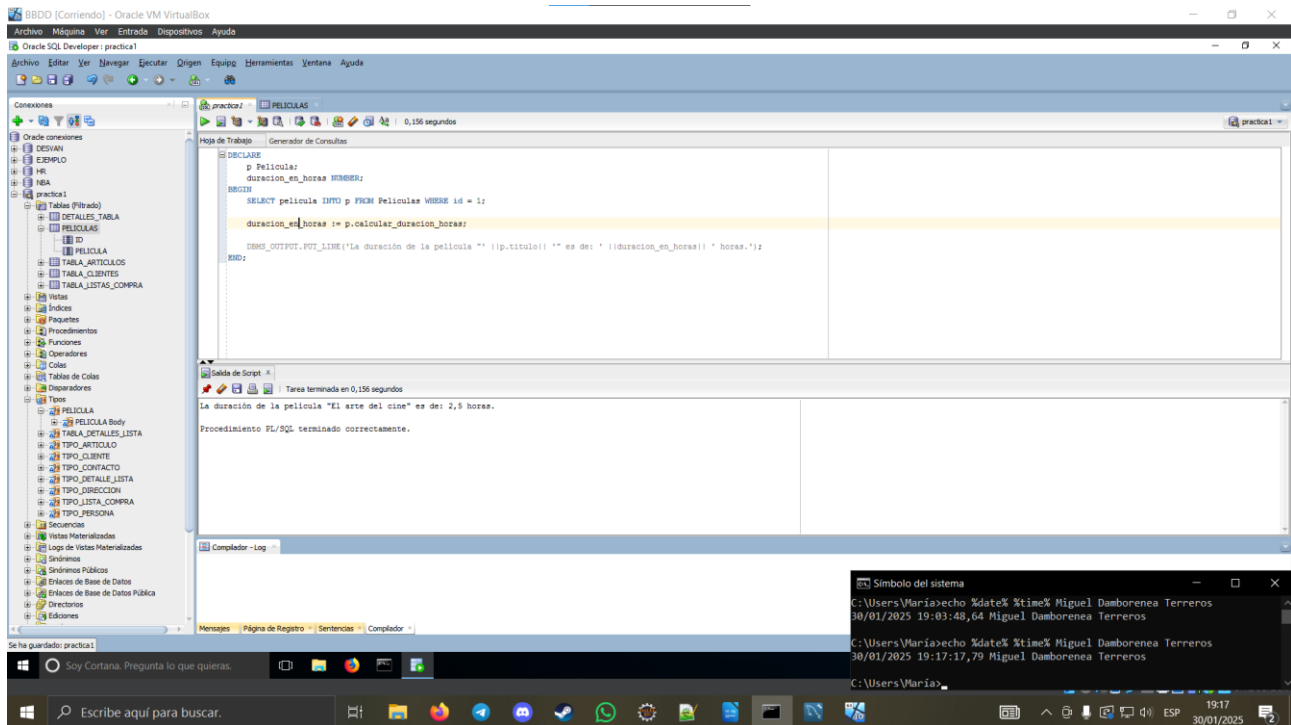
BEGIN

SELECT pelicula INTO p FROM Peliculas WHERE id = 1;

duracion_en_horas := p.calcular_duracion_horas;

DBMS_OUTPUT.PUT_LINE('La duración de la película "' || p.titulo || '" es de: ' || duracion_en_horas || ' horas.');

END;



The screenshot shows the Oracle SQL Developer interface. The main window displays a PL/SQL procedure named `calcula_duracion_horas` that takes a movie ID as input and returns its duration in hours. The procedure is executed, and the output shows the duration for the movie 'El arte del cine' as 2.6 hours.

```

DECLARE
  p Pelicula;
  duracion_en_horas NUMBER;
BEGIN
  SELECT pelicula INTO p FROM Peliculas WHERE id = 1;

  duracion_en_horas := p.calcular_duracion_horas;

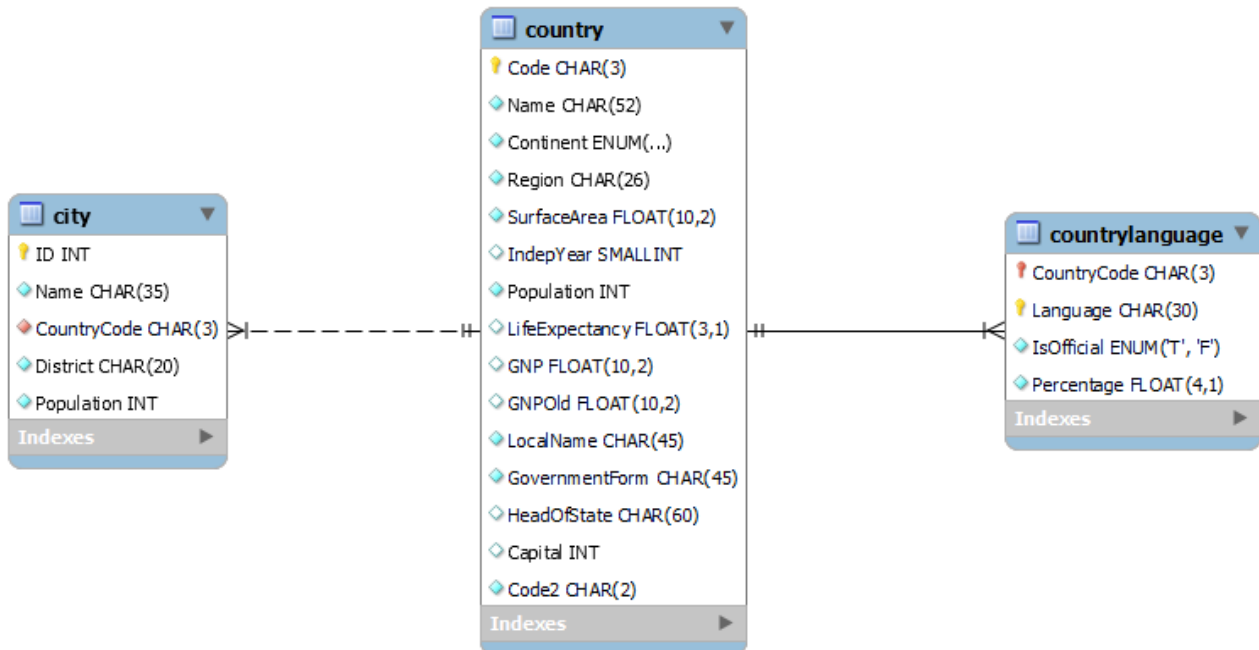
  DBMS_OUTPUT.PUT_LINE('La duración de la película "' || p.titulo || '" es de: ' || duracion_en_horas || ' horas.');
```

La duración de la película "El arte del cine" es de: 2.6 horas.

Procedimiento PL/SQL terminado correctamente.

3. JDBC (2,5 puntos) (adjunta captura de ejecución correcta)

Suponiendo que pasamos a trabajar con los datos almacenados en tablas de MySQL y JDBC, tenemos este esquema de BD llamado *world* cuyo script de creación podéis encontrarlo en el enunciado del examen.

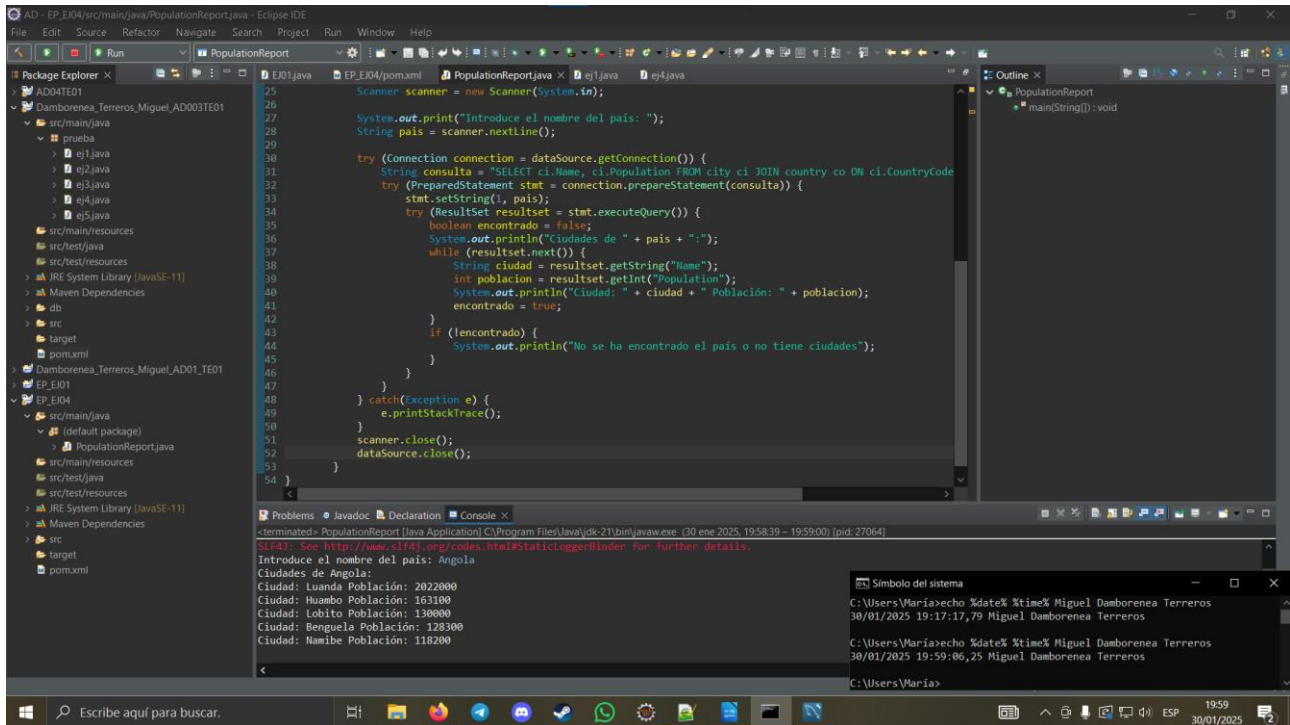


Crea una clase Java llamada *PopulationReport* que genere un informe del número total de habitantes en las ciudades de un país específico. El programa solicitará al usuario que introduzca el nombre del país y luego mostrará una lista de todas las ciudades junto con su población total.

Puedes usar un proyecto Java o Maven para realizar el ejercicio.

```

Introduce el nombre del país: China
Ciudad                Población
-----
Shanghai              9696300
Peking                7472000
Chongqing             6351600
Tianjin               5286800
Wuhan                 4344600
Harbin                4289800
Shenyang              4265200
Kanton [Guangzhou]    4256300
Chengdu               3361500
    
```



```

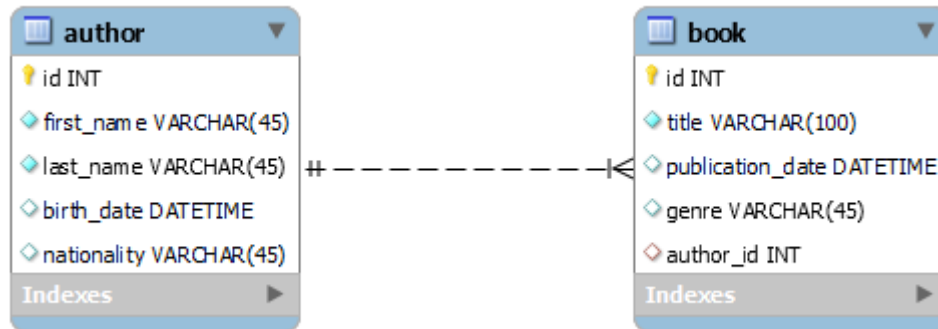
25 Scanner scanner = new Scanner(System.in);
26
27 System.out.print("Introduce el nombre del país: ");
28 String pais = scanner.nextLine();
29
30 try (Connection connection = dataSource.getConnection()) {
31     String consulta = "SELECT ci.Name, ci.Population FROM city ci JOIN country co ON ci.CountryCode = co.CountryCode";
32     try (PreparedStatement stmt = connection.prepareStatement(consulta)) {
33         stmt.setString(1, pais);
34         try (ResultSet resultSet = stmt.executeQuery()) {
35             boolean encontrado = false;
36             System.out.println("Ciudades de " + pais + ":");
37             while (resultSet.next()) {
38                 String ciudad = resultSet.getString("Name");
39                 int poblacion = resultSet.getInt("Population");
40                 System.out.println("Ciudad: " + ciudad + " Población: " + poblacion);
41                 encontrado = true;
42             }
43             if (!encontrado) {
44                 System.out.println("No se ha encontrado el país o no tiene ciudades");
45             }
46         }
47     } catch (Exception e) {
48         e.printStackTrace();
49     }
50     scanner.close();
51     dataSource.close();
52 }
53
54

```

<terminated> PopulationReport [Java Application] C:\Program Files\Java\jdk-21\bin\javaw.exe (30 ene 2025, 19:58:39 - 19:59:00) [pid: 27064]
 SLF4J: See <http://www.slf4j.org/codes.html#StaticLoggerBinder> for further details.
 Introduce el nombre del país: Angola
 Ciudades de Angola:
 Ciudad: Luanda Población: 2022000
 Ciudad: Huambo Población: 163100
 Ciudad: Lobito Población: 130000
 Ciudad: Benguela Población: 128300
 Ciudad: Namibe Población: 118200

4. HIBERNATE (3 puntos) (adjunta captura de ejecución correcta para cada ejercicio)

Pasamos a trabajar con Hibernate ORM.



Crea un proyecto Maven con las clases entidad *Author.java* y *Book.java* y realiza en ellos las anotaciones necesarias teniendo en cuenta la estructura relacional de la BD *library* mostrada en la imagen y cuyo script de creación podéis encontrar en el enunciado del examen.

OneToMany bidireccional.

- a) Crea una clase llamada *CreateAuthorWithBooks* que permita crear un autor con varios libros asociados.

```
Hibernate: insert into author (birth_date,first_name,last_name,nationality) values (?,?,,?)
Hibernate: insert into book (author_id,genre,publication_date,title) values (?,?,,?)
Hibernate: insert into book (author_id,genre,publication_date,title) values (?,?,,?)
Autor y libros creados exitosamente.
```

- b) Realiza una clase llamada *ListBooksByAuthor* que liste todos los libros escritos por un autor específico, dado su nombre. Puedes "hardcodear" el nombre en la clase directamente

```
Hibernate: select a1_0.id,a1_0.birth_date,a1_0.first_name,a1_0.last_
Libros de Gabriel García Márquez:
Hibernate: select b1_0.author_id,b1_0.id,b1_0.genre,b1_0.publication
- El amor en los tiempos del cólera
- Cien años de soledad
```

- c) Implementa una clase llamada *UpdateBookDetails* que permita actualizar los detalles de un libro (el título y el año de publicación) dado su id.

```
Introduce el ID del libro que deseas actualizar: 4
Introduce el nuevo título del libro: Cien años de soledad (Edición revisada)
Introduce el nuevo año de publicación (YYYY-MM-DD): 2025-01-07
Hibernate: select b1_0.id,b1_0.author_id,b1_0.genre,b1_0.publication_date,b1_0.title from book b1_0 where
Detalles del libro actualizados correctamente:
- Título: Cien años de soledad (Edición revisada)
- Fecha de publicación: 2025-01-07
Hibernate: update book set author_id=?,genre=?,publication_date=?,title=? where id=?
```



ENLACES GITHUB:

Ficheros:

JDBC:

Hibernate: