

## Entornos de Desarrollo

### Práctica 2 Unidad 3: Importación de librerías y exportación de programas. Consultas en MySQL.

Importación de librerías en Netbeans

Exportaciones en Netbeans

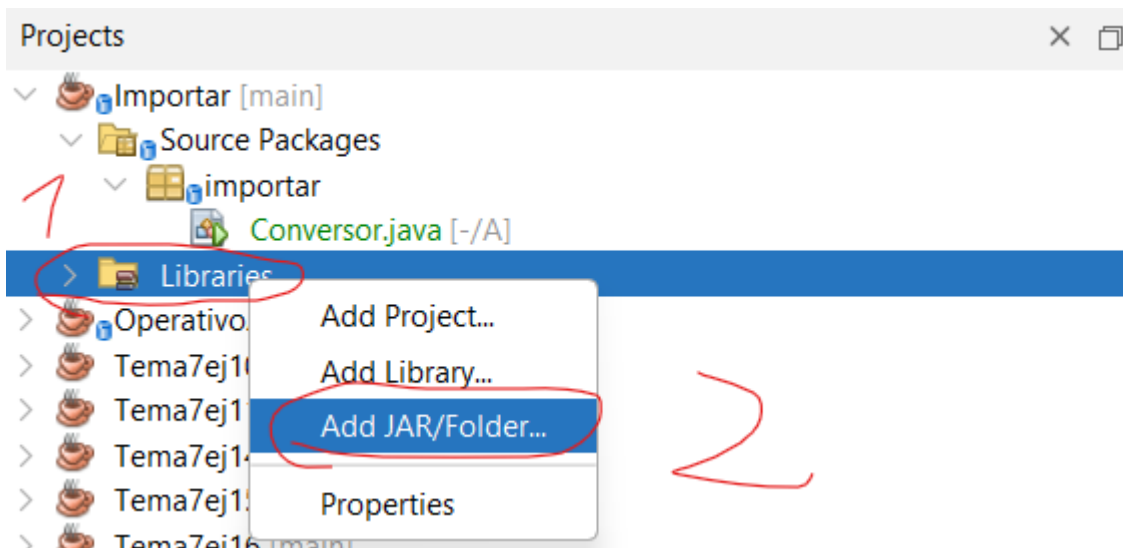
Importación y exportación en Eclipse

Consultas EN MySQL

#### Importación de librerías en Netbeans:

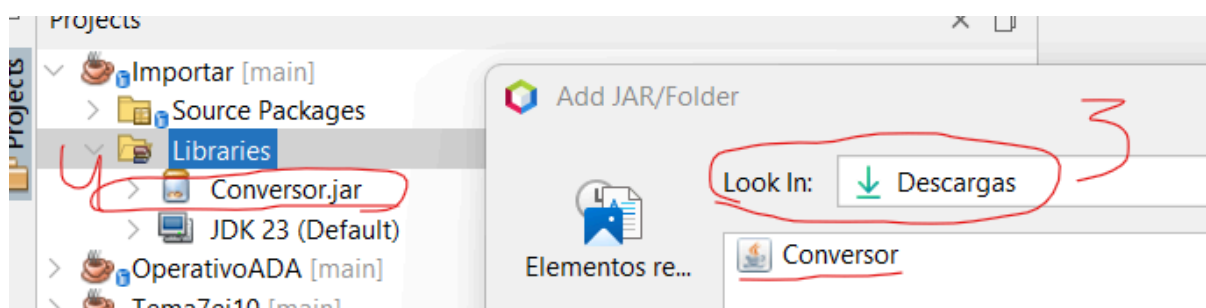
**1)** Tenemos creado el proyecto **Importar** con la clase **Conversor** (por el nombre de la librería que usaremos)

**2)** En Libraries hacemos click derecho y después en “**Add JAR/Folder...**”



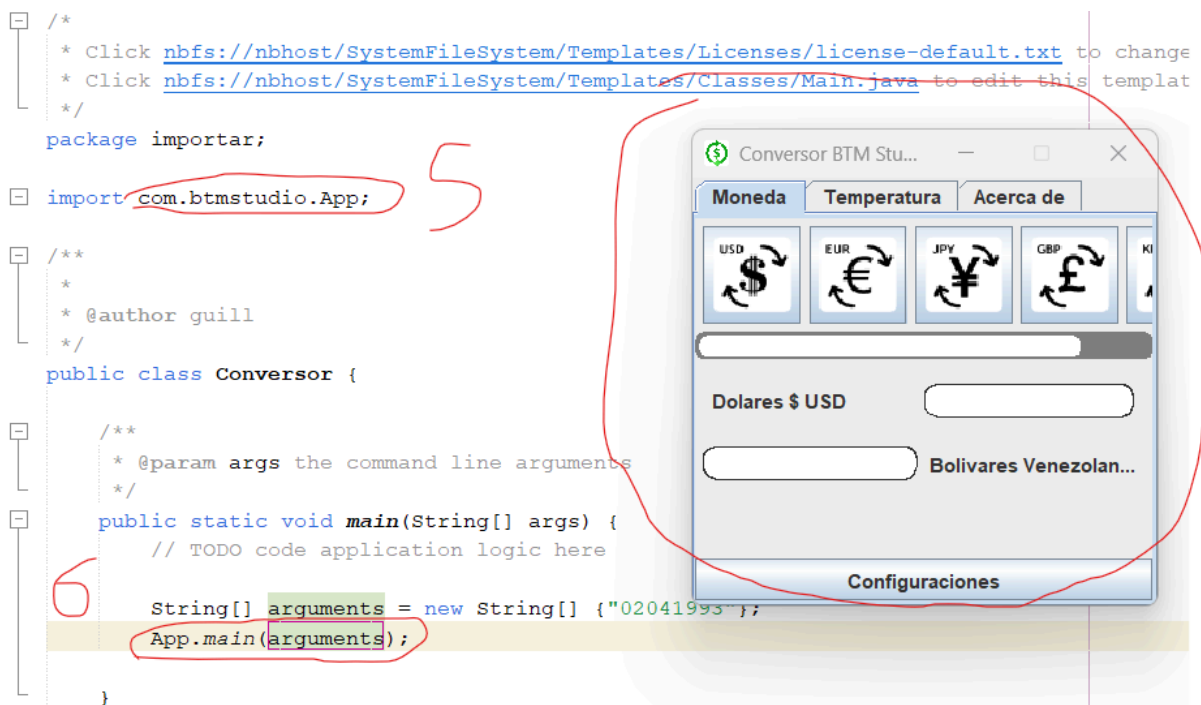
**3)** Buscamos en la carpeta donde tengamos guardada la librería, en mi caso **Descargas**.

**4)** Pulsamos ok y vemos que **Conversor.jar** ha sido añadido a **Libraries**.



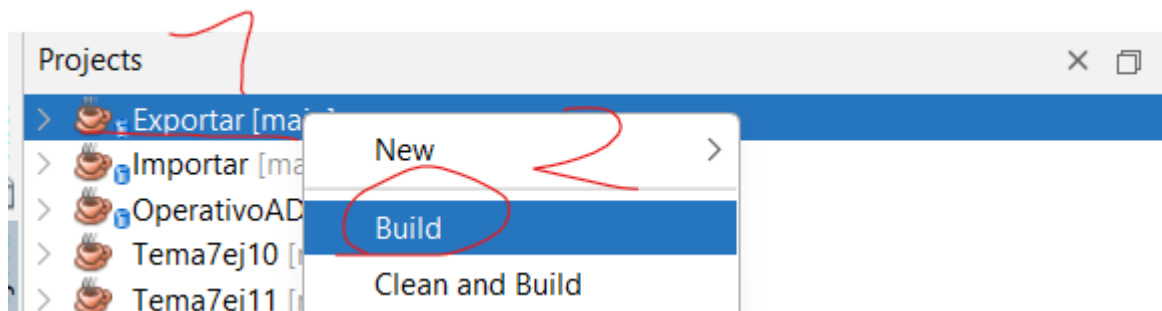
5) Ahora solo queda hacer el import con **import com.btmstudio.App;**

6) Llamamos al método main de App y le introducimos una variable en String que hemos creado. Al darle a run debería aparecerse el conversor.



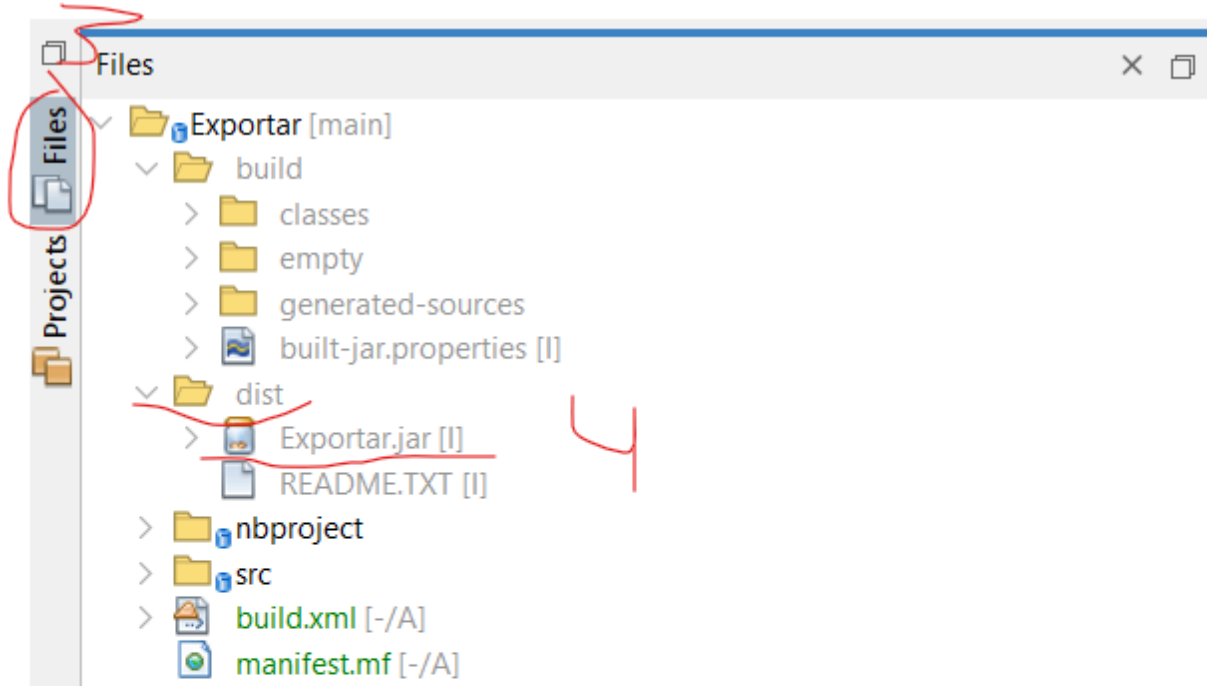
### Exportaciones en NetBeans:

- 1) Tenemos creado el proyecto Exportar. En este hemos creado un panel de botones que veremos más adelante.
- 2) Hacemos click derecho en el proyecto y pulsamos **Build**. Esto creará el archivo **Exportar.jar** en **Files**.



3) Para encontrar **Files** podemos pulsar **Control + 2**.

4) Como vemos, en la carpeta dist está creado **Exportar.jar**.

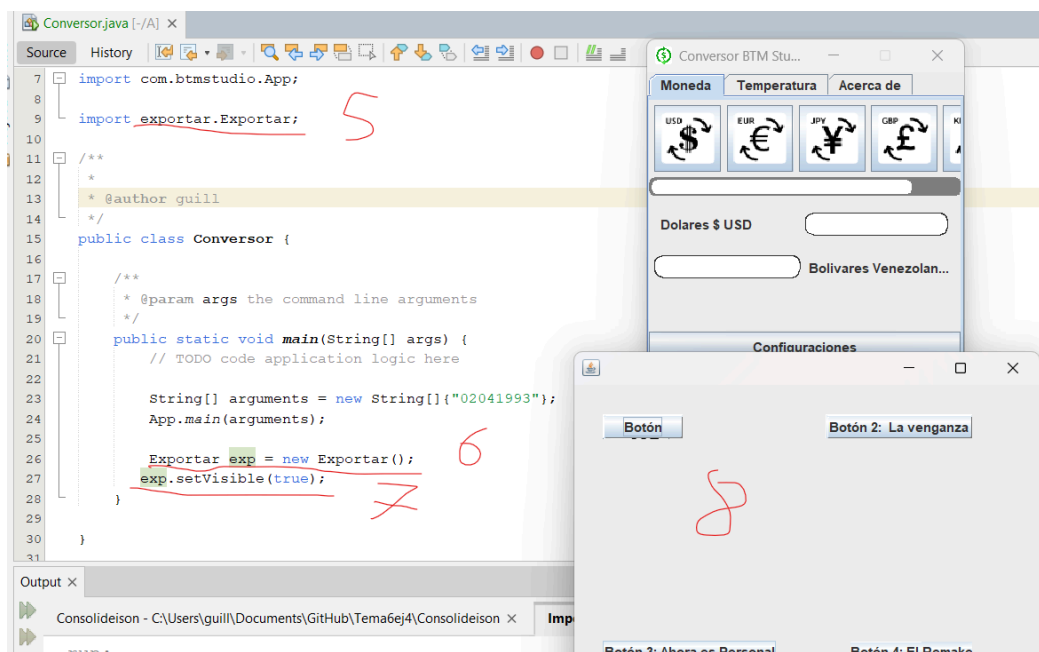


5) Ahora hemos vuelto al proyecto Importar e importamos el proyecto Exportar con **import exportar.Exportar;**

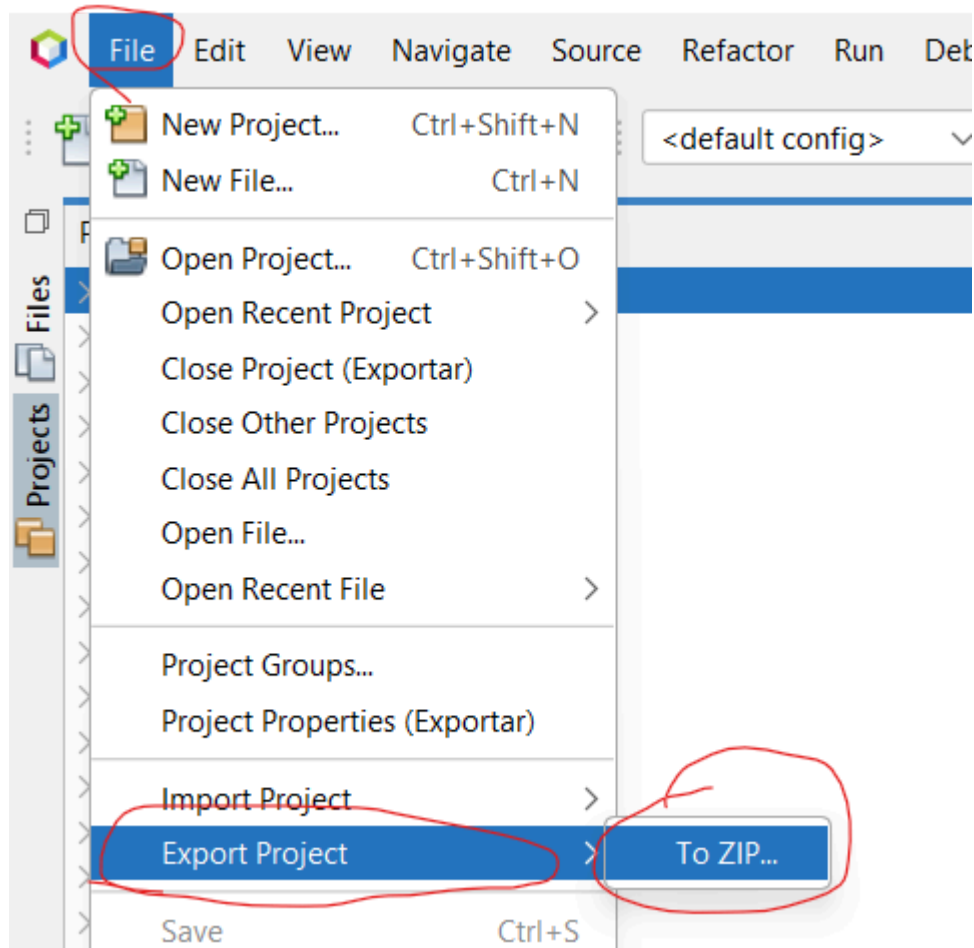
6) Lo llamamos creando un objeto con **Exportar exp = new Exportar();**

7) Para que los botones sean visibles debemos utilizar el método **setVisible** con **exp.setVisible(true);**

8) Como podemos ver, el menú de botones es visible.

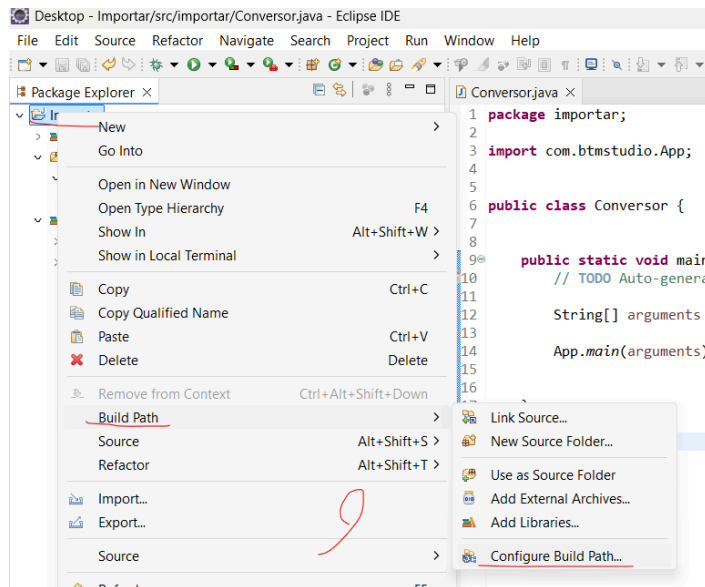


(Otra forma de exportar un proyecto es **Files > Export Project > To ZIP...**)



### Importación y exportación en Eclipse:

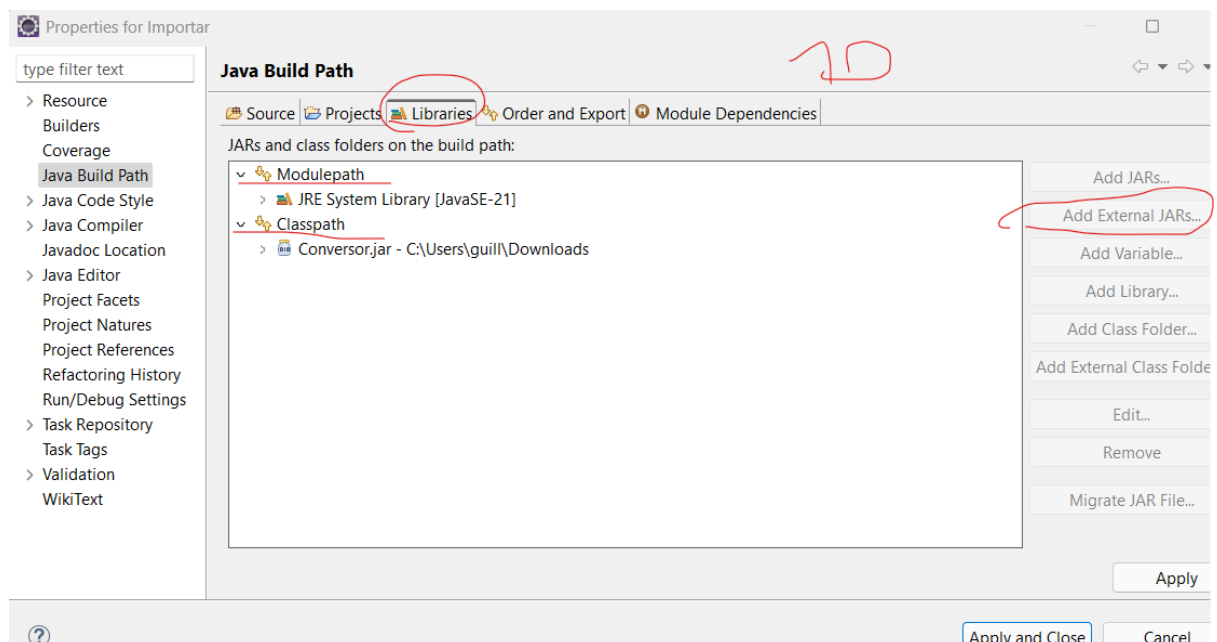
- 9) Para hacerlo en Eclipse, al crear el proyecto hay que hacer doble clic sobre el proyecto, de ahí **Build Path** y **Configure Build Path...**



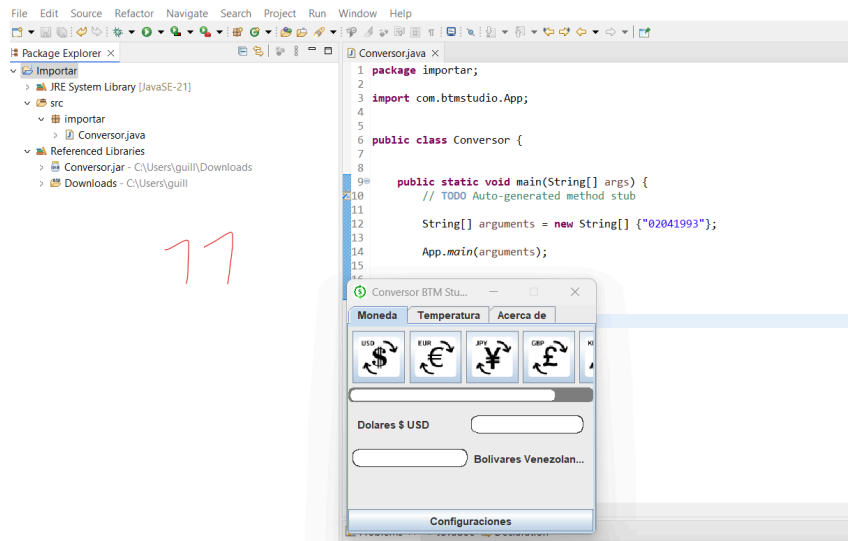
- 10)** En **Libraries**, seleccionamos **Modulepath** o **Classpath** (EXTREMADAMENTE IMPORTANTE: Si la librería no contiene un archivo llamado module-info, significa que NO está estructurado como un módulo.

POR TANTO hay que importar la librería en el Classpath y hacer que la Clase principal también esté ahí. De lo contrario Eclipse no te dejará hacer el import.

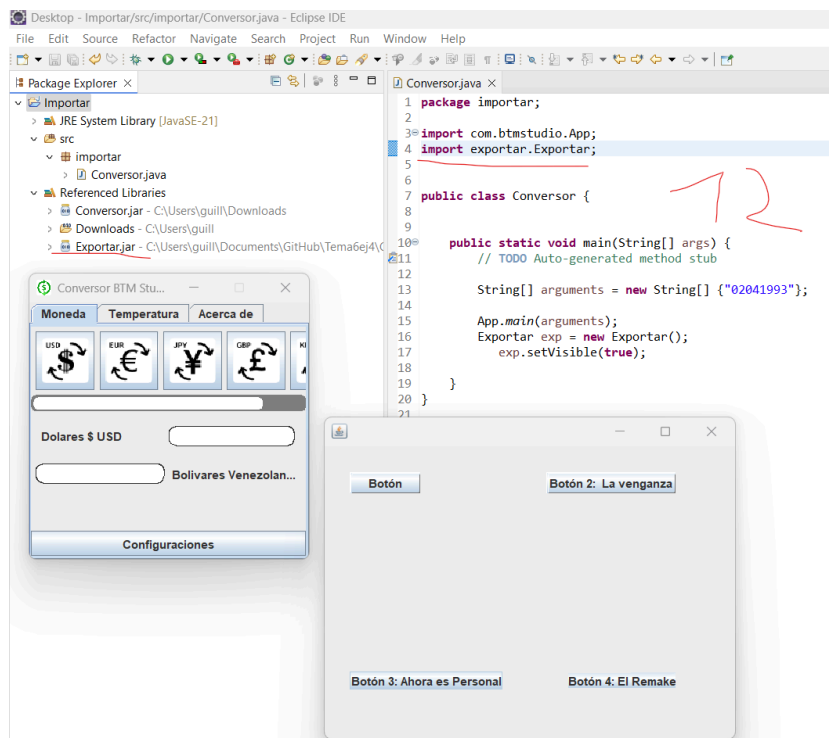
Hacemos clic en **Add External JARs...** y seleccionamos la librería Converter.



- 11)** Si la librería y el main están localizados en el path correspondiente (en nuestro caso, Classpath), debería arrancar como en la imagen:

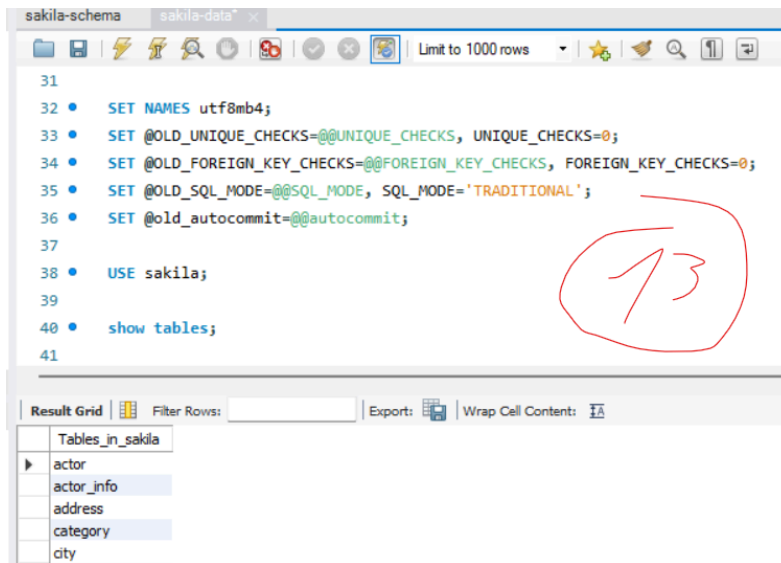


- 12) Hemos importado la librería creada en la práctica anterior **Exportar.jar** y hecho el código igual para poder ejecutar este archivo en Eclipse



### Consultas en MySQL:

- 13) Lo primero es descargar las queries de sakila y crearlas haciendo click en el icono del rayo arriba a la izquierda



```

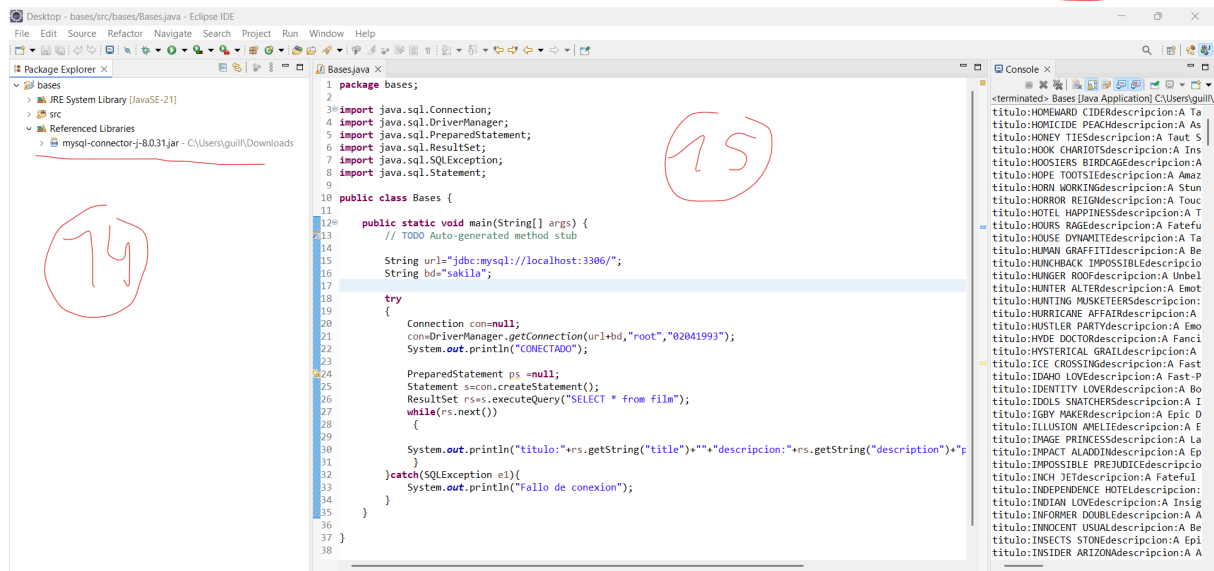
31
32 • SET NAMES utf8mb4;
33 • SET @@OLD_UNIQUE_CHECKS=@@UNIQUE_CHECKS, UNIQUE_CHECKS=0;
34 • SET @@OLD_FOREIGN_KEY_CHECKS=@@FOREIGN_KEY_CHECKS, FOREIGN_KEY_CHECKS=0;
35 • SET @@OLD_SQL_MODE=@@SQL_MODE, SQL_MODE='TRADITIONAL';
36 • SET @@old_autocommit=@@autocommit;
37
38 • USE sakila;
39
40 • show tables;
41

```

Result Grid | Filter Rows: | Export: | Wrap Cell Content:

Tables_in_sakila
actor
actor_info
address
category
city

- 14) Después conectarla importando la librería mysql-connector que se nos ha proporcionado.
- 15) Usamos el código apropiado (debemos asegurarnos de que el localhost, la contraseña y el nombre de usuario son correctos, así como de importar todas las herramientas necesarias) Hemos escrito el comando **SELECT \* from film**, para seleccionar todos los datos de la tabla llamada film.
- 16) Vemos que la consulta funciona

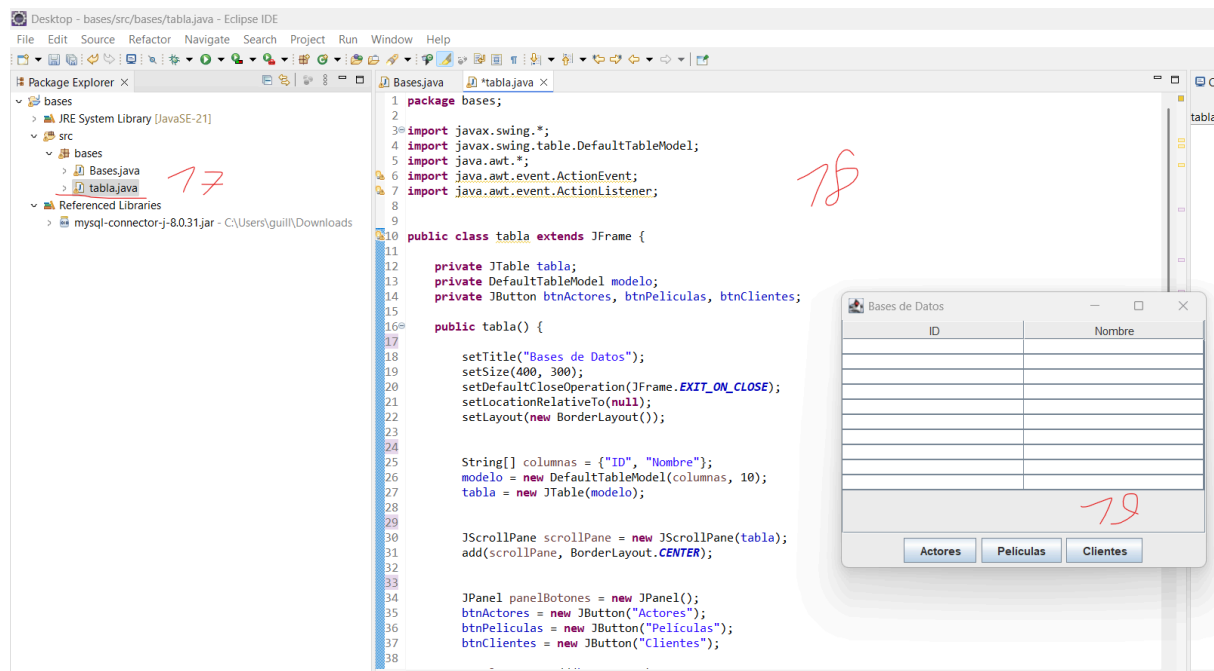


```

1 package bases;
2
3 import java.sql.Connection;
4 import java.sql.DriverManager;
5 import java.sql.PreparedStatement;
6 import java.sql.ResultSet;
7 import java.sql.SQLException;
8 import java.sql.Statement;
9
10 public class Bases {
11
12     public static void main(String[] args) {
13         // TODO Auto-generated method stub
14
15         String url="jdbc:mysql://localhost:3306/";
16         String bd="sakila";
17
18         try {
19             Connection con=null;
20             con=DriverManager.getConnection(url+bd,"root","02041993");
21             System.out.println("CONECTADO");
22
23             PreparedStatement ps =null;
24             Statement s=con.createStatement();
25             ResultSet rs=s.executeQuery("SELECT * from film");
26             while(rs.next())
27             {
28                 System.out.println("titulo:"+rs.getString("title")+"*"+rs.getString("description")+";");
29             }
30         } catch (SQLException e1) {
31             System.out.println("Fallo de conexion");
32         }
33     }
34 }
35
36
37
38

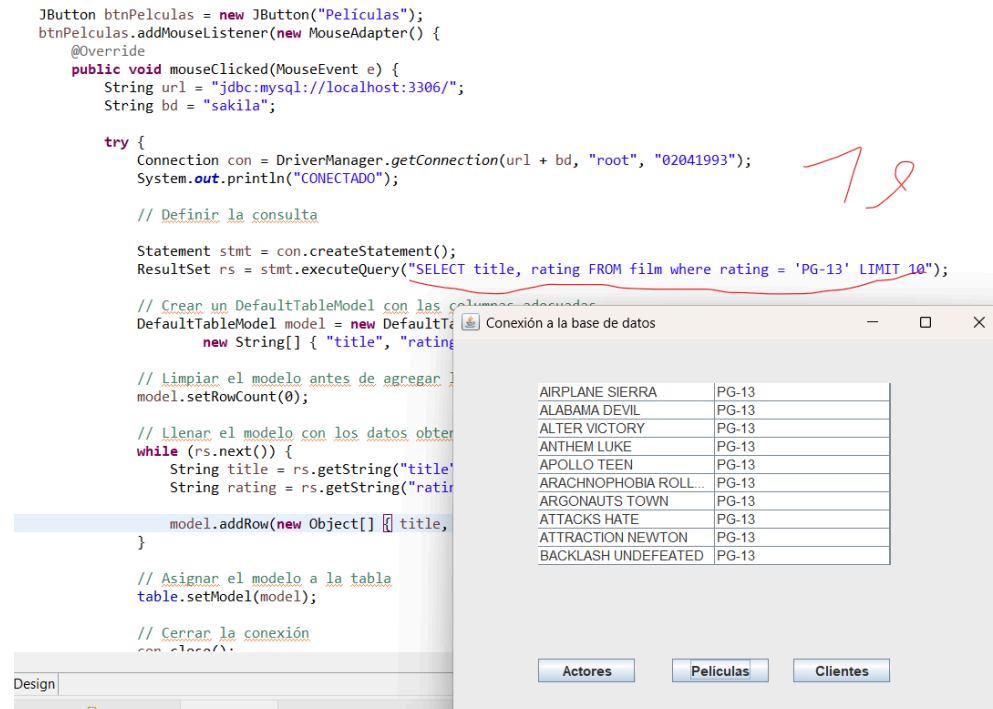
```

- 17) Ahora vamos a crear la tabla en Eclipse, tal como se nos ha pedido. Hemos creado la clase tabla, 18) hecho los imports necesarios y escrito el código.
- 19) A la hora de ejecutarlo, funciona.



**19) De la tabla film, seleccionar title y rating, siempre que sea PG-13**

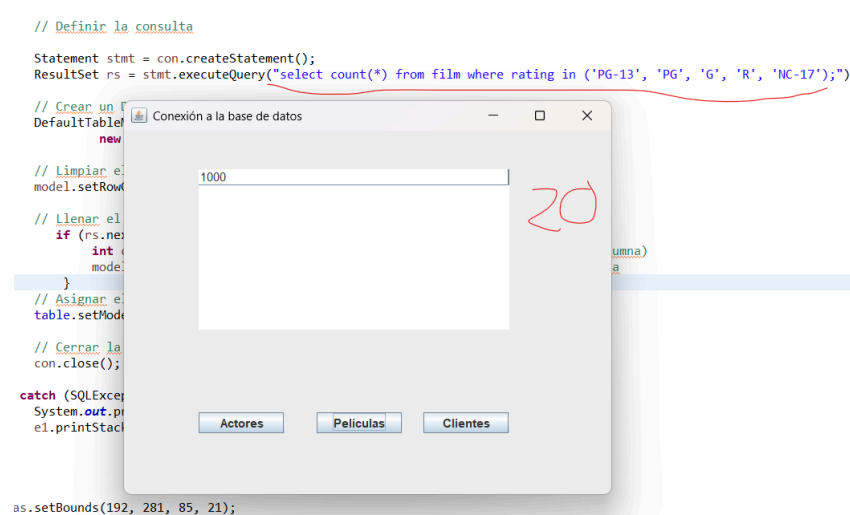
***SELECT TITLE, RATING FROM FILM WHERE RATING = 'PG-13' LIMIT 10"***



**20) Contar cuántas películas hay del Rating PG-13, PG, G, R, NC-17**

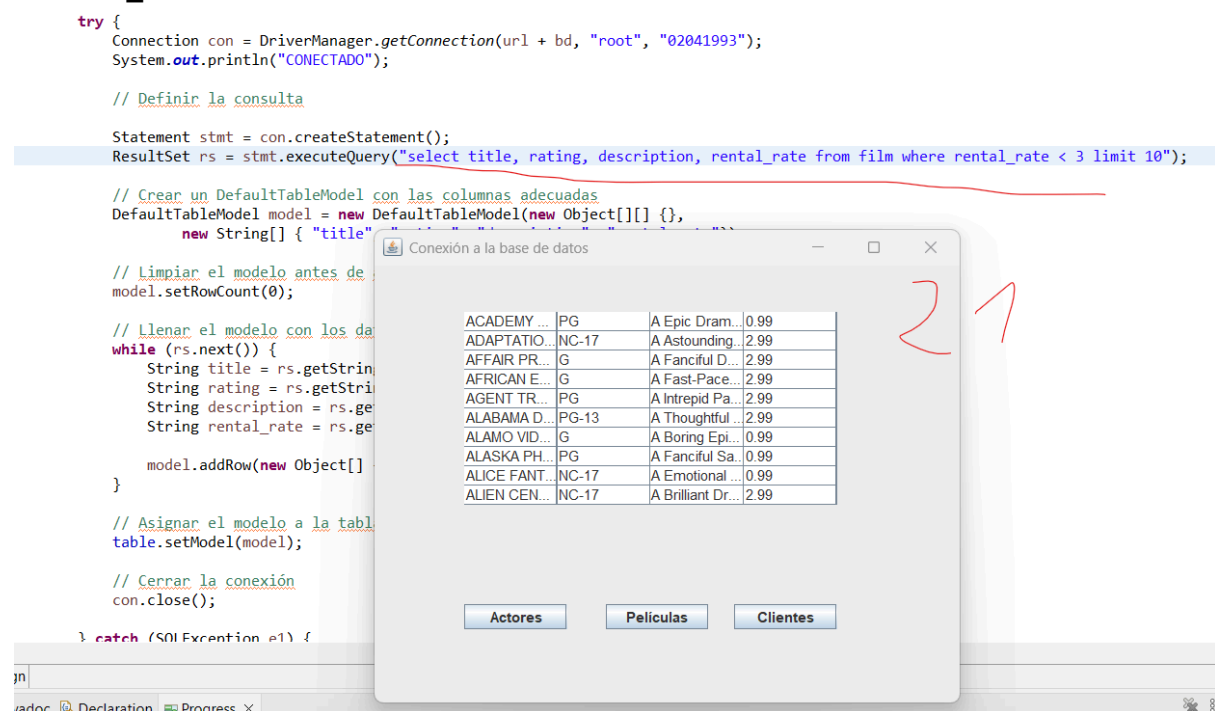
***SELECT COUNT(\*) FROM FILM WHERE RATING IN ('PG-13', 'PG', 'G', 'R', 'NV-17')***





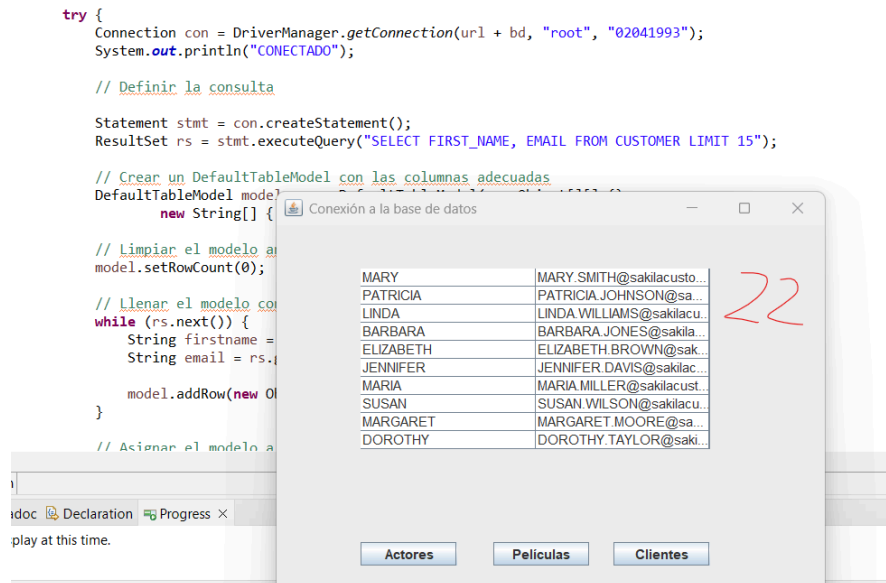
**21) Muestre los 10 primeras consultas de título, descripción y rating de una película cuando rental rate sea menor que 3**

**SELECT TITLE, RATING, DESCRIPTION, RENTAL\_RATE FROM FILM WHERE RENTAL\_RATE < 3 LIMIT 10**



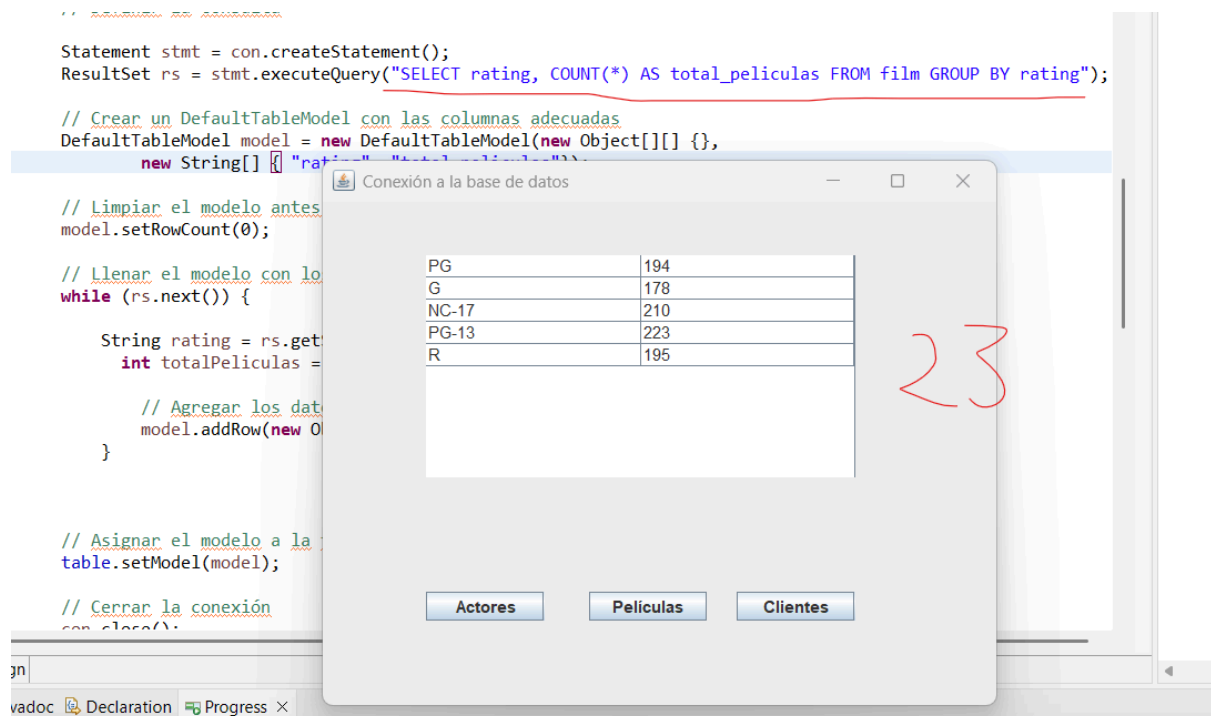
**22) De la tabla Clientes customer, escriba los nombres e emails de solo 15 clientes**

**SELECT FIRST\_NAME, EMAIL FROM CUSTOMER LIMIT 15**



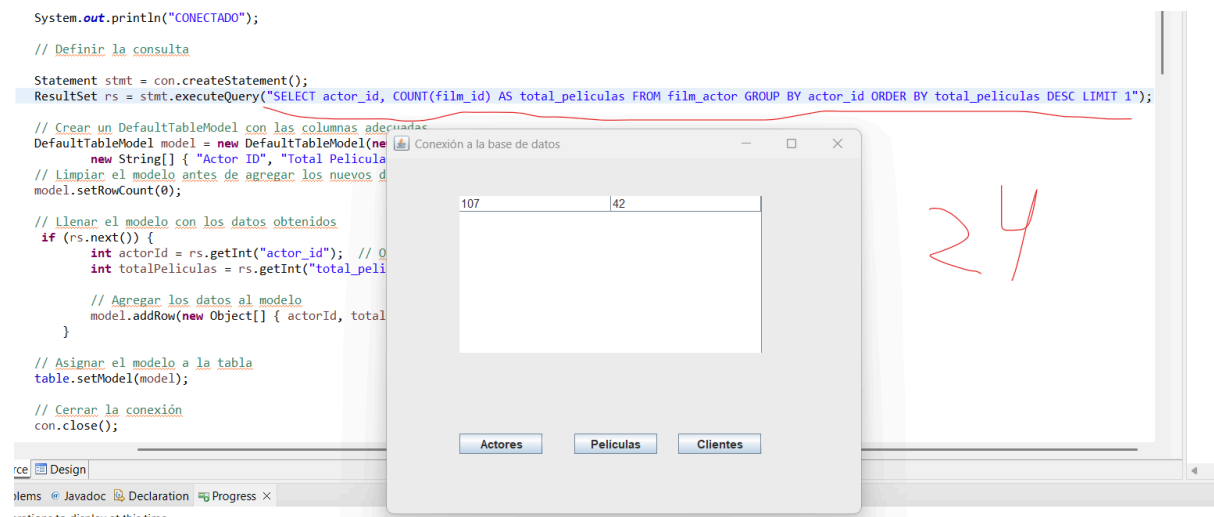
**23)** En la tabla film, sumar las películas de cada rating, mostrar el rating y su suma.

***SELECT RATING, COUNT(\*) AS TOTAL\_PELICULAS FROM FILM GROUP BY RATING***



**24)** En la tabla de actores, ¿de qué actor hay más películas?. Para contar usar COUNT, u ORDER BY ASC, ¿hay más de un actor, quiénes son?

***SELECT actor\_id, COUNT(film\_id) AS total\_peliculas FROM film\_actor GROUP BY actor\_id ORDER BY total\_peliculas DESC LIMIT 1***



**25)** Con la función MAX en la tabla de film, mostrar el máximo de rental\_duration

***SELECT MAX(rental\_duration) from film***

