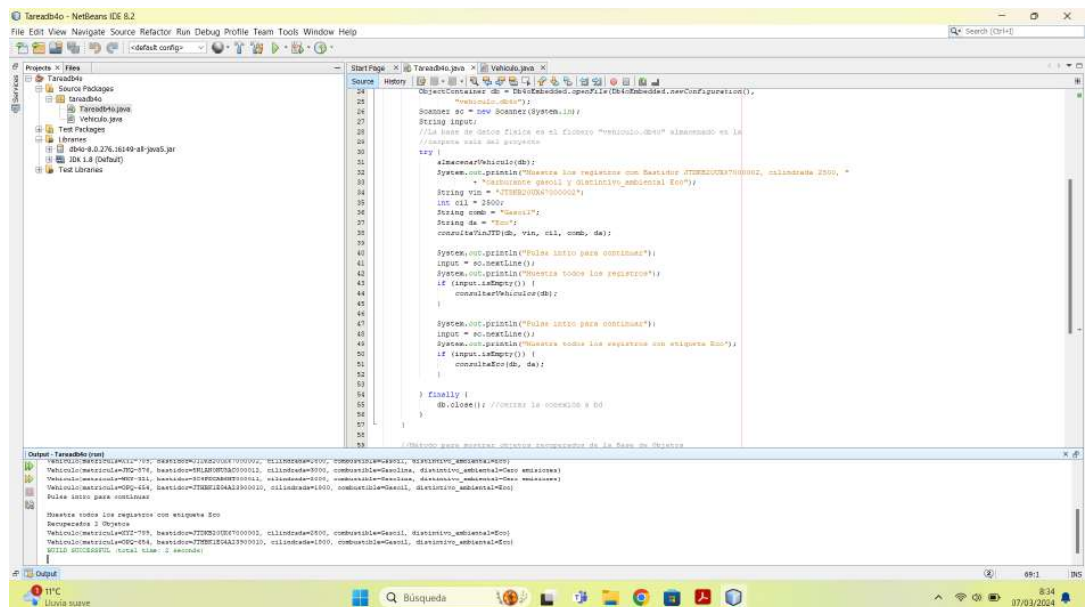
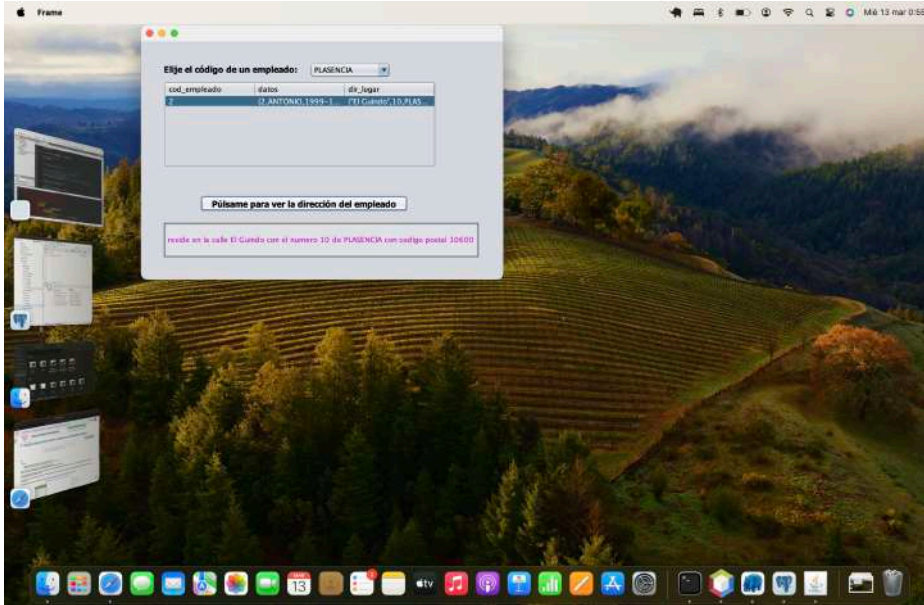


Tarea 5 para Acceso a Datos



Diego Manuel Carrasco Castañares

Detalles de la tarea de esta unidad.

TAREA 5

Hay que entregar un documento en pdf, con explicación y los pantallazos de la ejecución (muy importante) de la tarea. Si no se entrega, la nota tendrá una bajada de 2 puntos.

No te olvides controlar las excepciones y entregar el documento explicativo de la tarea con sus pantallazos de ejecución.

Con Netbeans y java debemos hacer lo siguiente:

1. Utilizando la base de datos orientado a objetos DB4o vamos a crear una base de datos llamada vehiculo.db4o para almacenar en ella los datos de vehículos. Usaremos una clase, lógicamente llamada Vehículo.

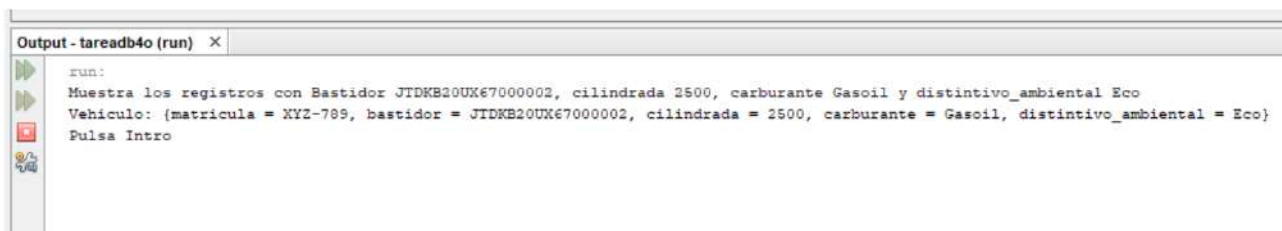
Almacenamos 5 registros por cada vehículo con los siguientes datos: matrícula, bastidor, cilindrada, carburante y distintivo ambiental. Estos son los datos a introducir.

```
"ABC-123", "1HGCM82633A400001", 1600, "Gasolina", "Cero emisiones"  
"XYZ-789", "JTDKB20UX67000002", 2500, "Gasoil", "Eco"  
"JKQ-876", "5N1AN0NU3AC000012", 3000, "Gasolina", "Cero emisiones"  
"WXY-321", "3C4PDCAB6HT000011", 2000, "Gasolina", "Cero emisiones"  
"OPQ-654", "JTHBK1EG4A23900010", 1800, "Gasoil", "Eco"
```

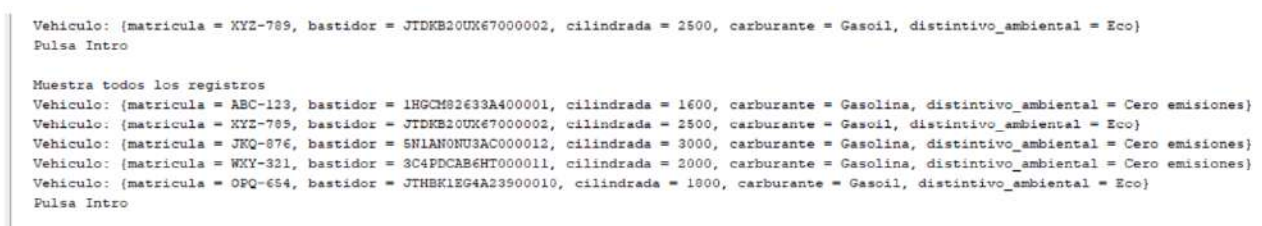
Clase vehículos 0,5 puntos

Luego mostramos por pantalla tres cosas.

1.- Los registros con Bastidor JTDKB20UX67000002 , cilindrada 2500, carburante Gasoil y distintivo_ambiental Eco 0,5 puntos



2.- Luego todos los registros 0,5 puntos



3.-Y por último los registros con distintivo_ambiental Eco 0,5 puntos

```
Vehiculo: {matricula = OPQ-654, bastidor = JTBEK1EG4A23900010, cilindrada = 1800, carburante = Gasoil, distintivo_ambiental = Eco}
Pulsa Intro

Muestra los registros con distintivo_ambiental Eco
Vehiculo: {matricula = XYZ-789, bastidor = JTDKB20UX670000002, cilindrada = 2500, carburante = Gasoil, distintivo_ambiental = Eco}
Vehiculo: {matricula = OPQ-654, bastidor = JTBEK1EG4A23900010, cilindrada = 1800, carburante = Gasoil, distintivo_ambiental = Eco}
BUILD SUCCESSFUL (total time: 7 seconds)
```

(2 puntos)

2. Consultas con Query Tool en pgAdmin 4 con la base de datos anaconda que hemos creado cuando hemos estudiado el apartado 7 de la unidad. (2 puntos)

2.1.- Muestra los datos de la tabla datos_meteo (0,25 puntos)

2.2.- Con la tabla datos_meteo muestra el identificador de las provincias con sus temperaturas mínimas y máximas. (0,25 puntos)

2.3.- Mostrar la temperatura mínima y máxima de la provincia cuyo identificador es el 1 (0,25 puntos)

2.4.- Muestra todos los datos de la tabla datos_meteo y provincias. Las tablas datos_meteo y provincias (0,25 puntos)

2.5.- Muestra el identificador de la provincia de datos_meteo, el nombre de las provincias, el identificador del mes y las precipitaciones. Con las tablas datos_meteo y provincias (0,5 puntos)

2.6.

- Modifica el sql anterior que aparezca también el nombre del mes no el identificador. Utiliza las 3 tablas ahora. (0,5 puntos)

3. Ahora ya pasamos a postgresql (sistema de gestión de bases de datos relacional orientado a objetos) Conectar) utilizando el driver JDBC y en pgAdmin 4 creamos una base de datos llamada frutería.

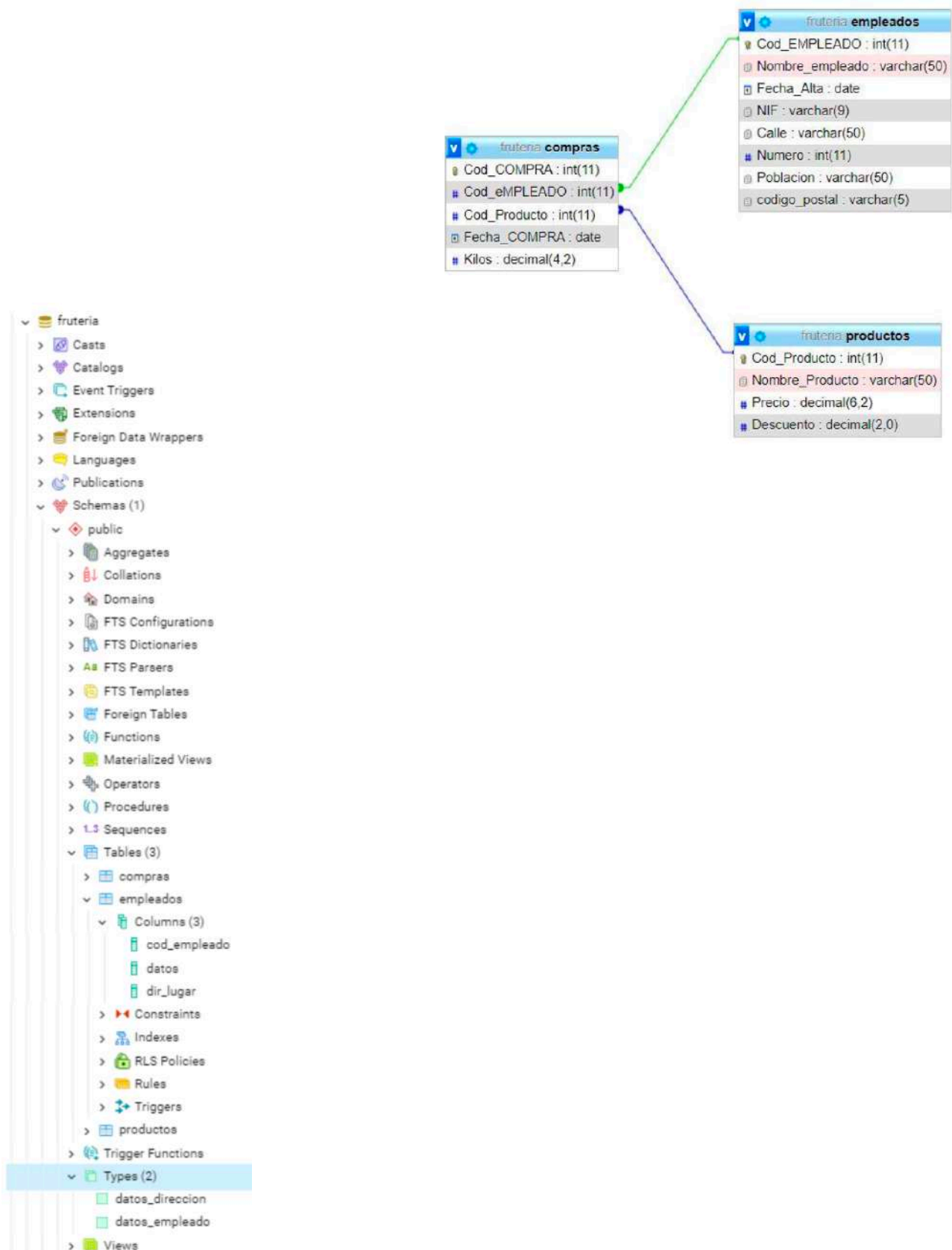
Crea también las tabla compras y productos con sus columnas.

La tabla empleados la crearemos en un proyecto, en Netbeans con java, que llamaremos TareaPostgresql.

La tabla tendrá tres campos: el código del empleado de tipo entero, datos de tipo datos_direccion y lugar de tipo datos_direccion. Insertaremos, manipularemos y mostraremos los datos de los empleados.

También debéis crear un función en base de datos fruteríaMira la segunda imagen para ver como deberá quedar una vez realizado el apartado.

Las consultas, recuerda con la clase Statement. Haz los siguientes apartados para conseguir la puntuación de (6 puntos)



3.1.- Crear la base de datos frutería con las dos tablas, compras y productos en pgAdmin 4.

Demuestra este proceso e imagen de que han sido creadas, la base de datos y las dos tablas con sus columnas (0,25 puntos)

3.2.- Explica la siguiente sentencia y para qué se utiliza: (0,25 puntos)

```
Statement stmt = conn.createStatement();
```

A continuación coge el proyecto que se adjunta llamado Tareapostgresql y rellena lo que falta.

3.3.

- Rellena el String de URL de la conexión a la base de datos frutería (0,5 puntos)

3.4.

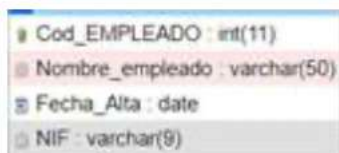
- Rellena los dos datos que faltan en los datos de tu conexión para que con esté correcta (0,5 puntos)

3.5.

- En el método Borrar_tabla escribe el String para: borrar la función mensaje que luego se construirá con los datos de la dirección del empleado, la tabla empleados y los tipos datos_empleado y datos_direccion (0,5 puntos)

3.6.

- En el método Crear_Tipo escribe o rellena el String con la consulta: Crea un tipo llamado datos_empleado con los siguientes campos (0,5 puntos)



3.7.

- En el método Crear_Tipo escribe o rellena el String llamado consulta2 con la consulta:

Crea un tipo llamado datos_direccion con los siguientes atributos. Créalo calle, número, código_postal y población (0,5 puntos)



3.8.

- En el método `Crear_Funcion`: Rellena el `String` para una función llamada `mensaje` para mostrar luego datos en `(jTextField)` del interfaz los datos de la dirección de un empleado seleccionado, con los campos de la calle, número, código postal y población. (0,5 puntos)

3.9.

- En el método `Crear_Tabla`: Rellena el `String` para crear una tabla llamada `empleados` con tres campos: `cod_empleado` de tipo `serial` y clave primaria, `datos_empleado` de tipo `datos_empleado` y `dir_lugar` de tipo `datos_direccion`. (0,5 puntos)

3.10.

- En el método `Insertar_Registros`: Rellena el `String` para insertar en la tabla `empleados` los siguientes datos. Utilizando los dos tipos creados anteriormente (recuerda dos `row` por cada empleado), es decir, por ejemplo un `row` sería
'JOSE','1999/02/12','111111111' y el otro 'El Peral',12,'CACERES','10100' (0,5 puntos)

```
(1,'JOSE','1999/02/12','111111111','El
Peral',12,'CACERES','10100'),
(2,'ANTONIO','1999/12/02','111111112','El
Guindo',10,'PLASENCIA','10600'),
(3,'MARIA','2009/02/12','111111113','La
Plaza',4,'CACERES','10100'),
(4,'LISA','2000/1/05','111111114','El
Rio',5,'NAVALMORAL','10300'),
(5,'JOSEMARIA','2014/02/15','111000101','La
Mano',6,'CACERES','10100'),
(6,'CARLOS','2015/02/02','111111116','Antonio
Soria',7,'CACERES','10100');
```

3.11.

- En el método `Rellenar_ComboBox`: Rellena el `String` para seleccionar las poblaciones, sin repetir, de la tabla `empleados` en orden ascendente. (0,5 puntos)



3.12.

- En el método `jComboBoxItemStateChanged`: pon de nuevo tu conexión y en método `rellenar_JTable` rellena el `String` para seleccionar todos los empleados de la población seleccionada en el `JComboBox`. (0,5 puntos)

3.13.

- En el método `mensaje`: pon de nuevo tu conexión y rellena el `String` de consulta para mostrar la función `mensaje` que correspondería al código del empleado que se ha seleccionado en la tabla. (0,5 puntos)

Recursos necesarios para realizar la Tarea.

Además de los contenidos y ejemplos de la unidad, te proporcionamos el archivo `AD05_Recurso_Tarea.zip` con la interfaz gráfica basada en `Swing` que debes utilizar.

Consejos y recomendaciones.

Sigue los pasos indicados en el enunciado. Solo debes ir completando las sentencias `SQL` de los comandos `Statement` indicados.

Indicaciones de entrega.

Una vez realizada la tarea elaborarás un único documento donde figuren el proyecto y un documento `pdf` donde figure la explicación y ejecución del proyecto. El envío se realizará a través de la plataforma de la forma establecida para ello, y el archivo se nombrará siguiendo las siguientes pautas:

`apellido1_apellido2_nombre_ADxx_Tareaxx_Entregaxx`

Asegúrate que el nombre no contenga la letra ñ, tildes ni caracteres especiales extraños. Así por ejemplo la alumna Begoña Sánchez Mañas para la primera unidad del MP de AD, debería nombrar esta tarea como...

`sanchez_manas_begona_AD05_Tarea05_Entrega01`

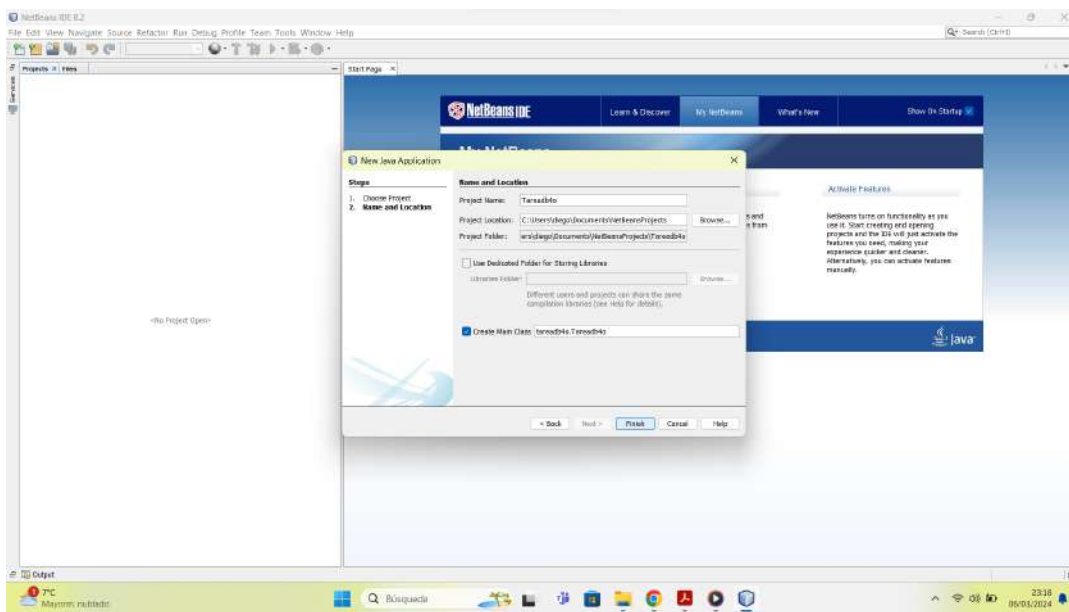
Indice:

1.- Creación de bases de datos.....	09
1.1.- Inserción de datos.....	10
1.2.- Mostrar tres datos.....	11
1.3.- Mostrar todos los registros.....	12
2.- Consultas con Query Tool en pgAdmin.....	13
2.1.- Mostrar datos tabla datos_meteo.....	13
2.2.- Mostrar identificador y temperaturas.....	13
2.3.- Mostrar temperatura de provincia 1.....	14
2.4.- Mostrar datos tablas datos_meteos y provincias.....	15
2.5.- Mostrar datos identificador, mes y precipitaciones.....	15
2.6.- Modificar consulta anterior para mostrar nombre.....	16
3.- Consultas postgresql.....	16
3.1.- Crear base de datos.....	16
3.2.- Explicar sentencia y para que se utiliza.....	17
3.3.- Rellenar String URL.....	17
3.4.- Rellenar datos que faltan.....	18
3.5.- String para borrar datos en método borrar_tabla.....	18
3.6.- String para método Crear_Tipo consulta.....	19
3.7.- String para método Crear_Tipo consulta2.....	19
3.8.- String para método Crear_Funcion.....	20
3.9.- String para método Crear_Tabla.....	20
3.10.- String para método Insertar_Registros.....	21
3.11.- String para método Rellenar_ComboBox.....	22
3.12.- Método jcomboBoxItemStateChanged.....	23
3.13.- Método mensaje.....	24

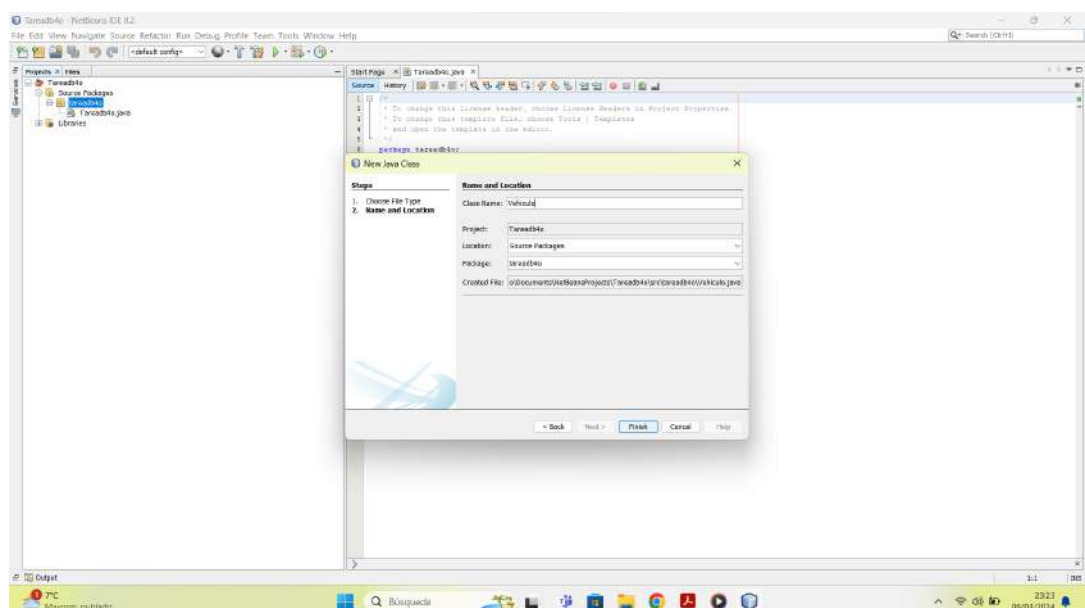
1. Utilizando la base de datos orientado a objetos DB4o vamos a crear una base de datos llamada `vehiculo.db4o` para almacenar en ella los datos de vehículos. Usaremos una clase, lógicamente llamada `Vehículo`.

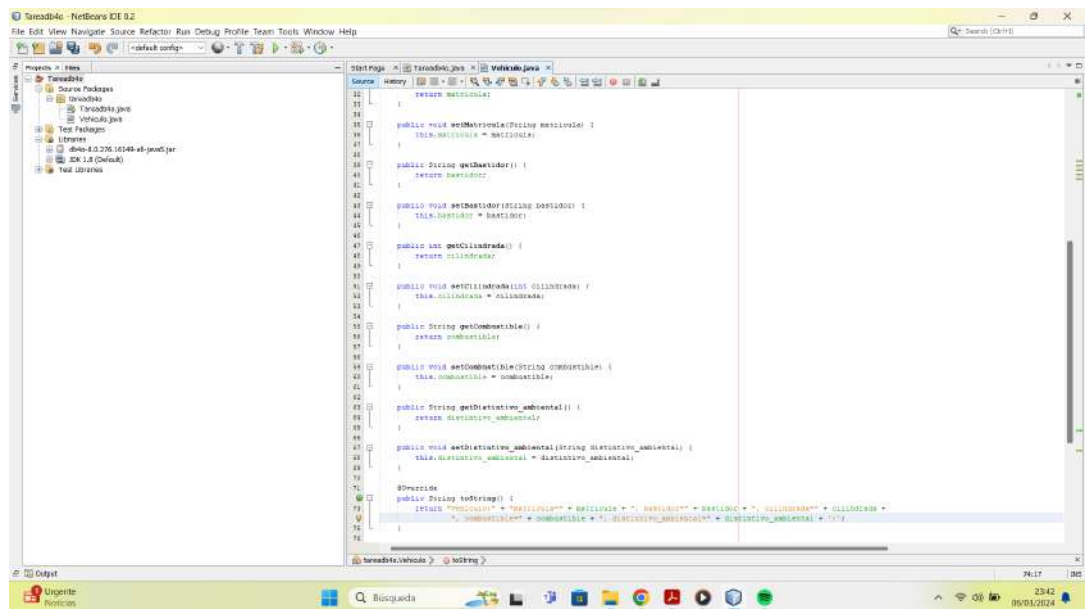
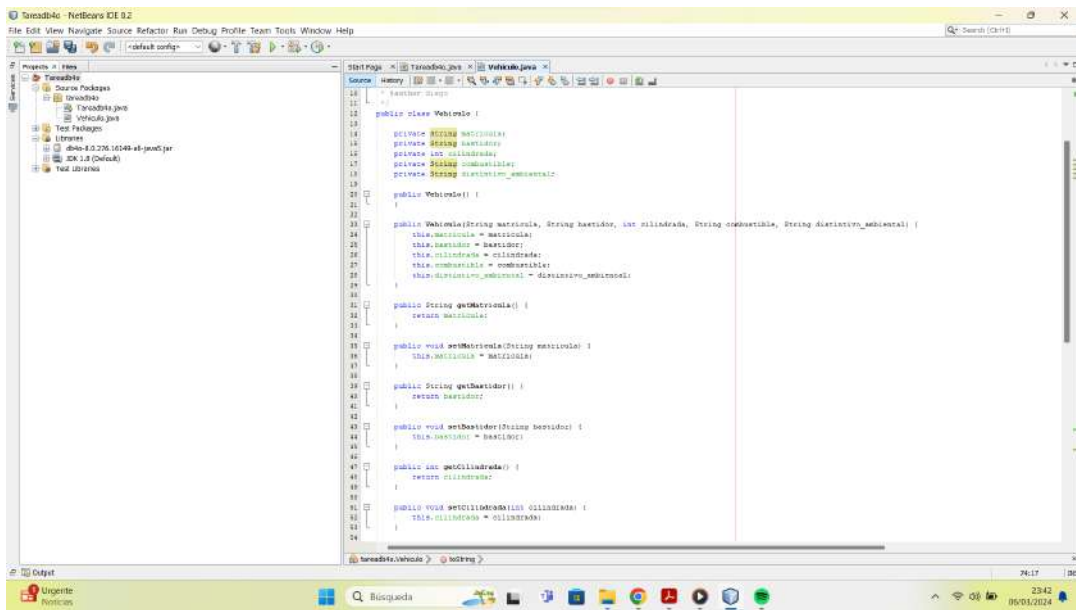
Almacenamos 5 registros por cada vehículo con los siguientes datos: matrícula, bastidor, cilindrada, carburante y distintivo ambiental.

Empezaremos creando el proyecto en NetBeans, al cual llamaremos `Tareadb4o`.

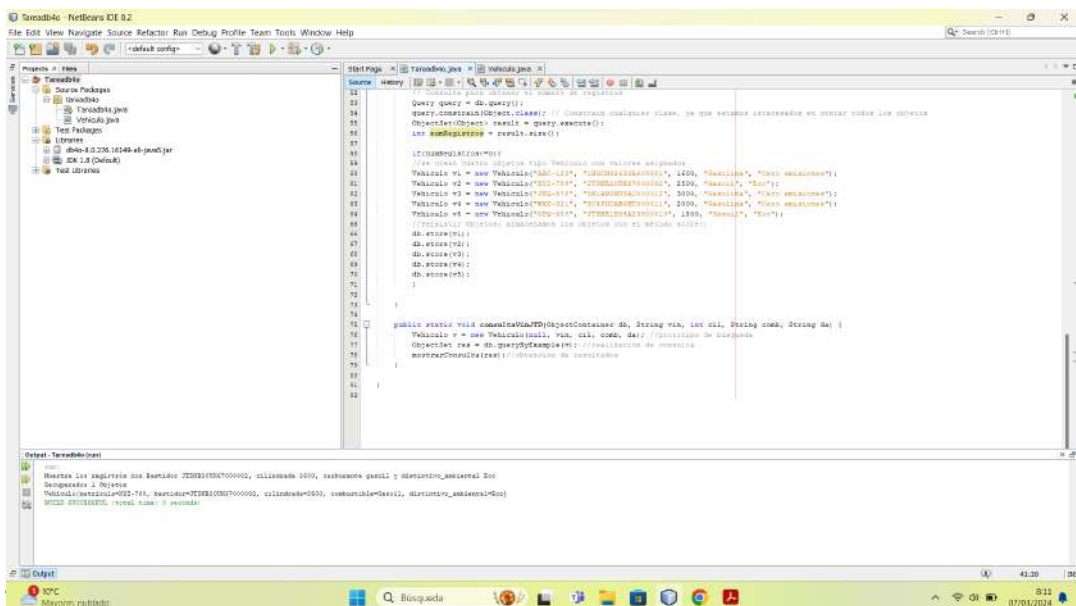


Posteriormente crearemos la clase `Vehículo` y la compondremos con los atributos mencionados en el enunciado, el constructor vacío y otro con todos los parámetros, los getter and setter y el método `toString`.

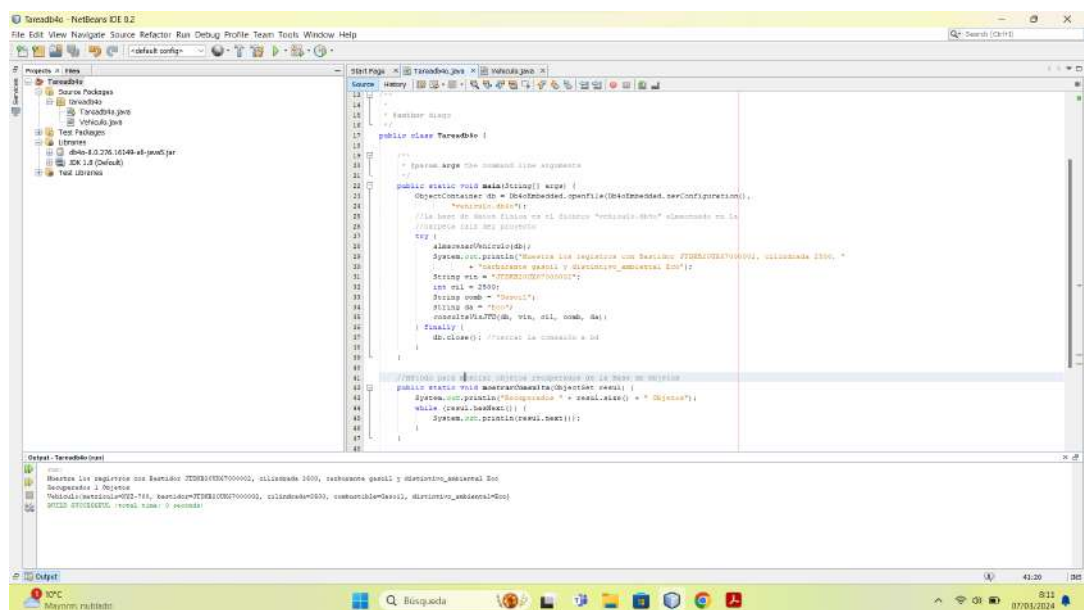
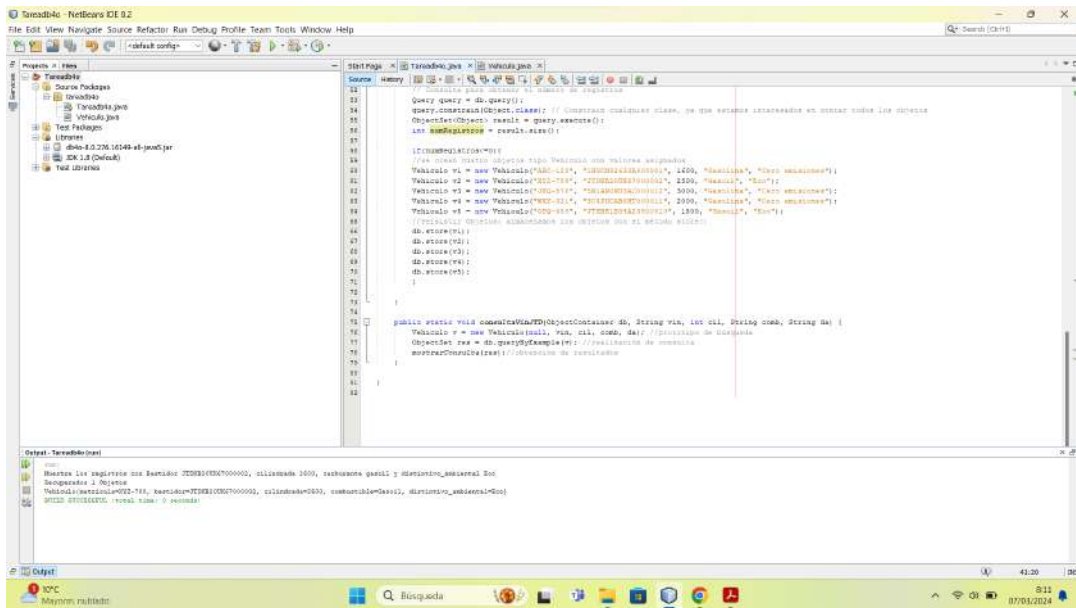




En la siguiente imagen podemos ver la inserción de los datos.



1.1.- Los registros con Bastidor JTDKB20UX67000002 , cilindrada 2500, carburante Gasoil y distintivo ambiental Eco 0,5 puntos



1.2.- Luego todos los registros 0,5 puntos

```

1  Tarea5.java
2  // Clase Tarea5
3  // Autor: Diego Manuel Carrasco Castaños
4  // Fecha: 07/03/2024
5
6  import java.sql.*;
7
8  public class Tarea5 {
9
10     // Método para insertar registros en la base de datos
11     public void insertarRegistros() {
12         // Conexión a la base de datos
13         String url = "jdbc:mysql://localhost:3306/tarea5";
14         String user = "root";
15         String pass = "";
16         try {
17             Class.forName("com.mysql.cj.jdbc.Driver");
18             Connection con = DriverManager.getConnection(url, user, pass);
19             Statement stmt = con.createStatement();
20             String sql = "INSERT INTO vehiculos (matricula, marca, modelo, año, color, precio) VALUES ('ABC123', 'Toyota', 'Corolla', 2020, 'Rojo', 15000);";
21             stmt.executeUpdate(sql);
22             con.close();
23         } catch (Exception e) {
24             e.printStackTrace();
25         }
26     }
27
28     // Método para actualizar registros en la base de datos
29     public void actualizarRegistros() {
30         // Conexión a la base de datos
31         String url = "jdbc:mysql://localhost:3306/tarea5";
32         String user = "root";
33         String pass = "";
34         try {
35             Class.forName("com.mysql.cj.jdbc.Driver");
36             Connection con = DriverManager.getConnection(url, user, pass);
37             Statement stmt = con.createStatement();
38             String sql = "UPDATE vehiculos SET marca = 'Ford' WHERE matricula = 'ABC123';";
39             stmt.executeUpdate(sql);
40             con.close();
41         } catch (Exception e) {
42             e.printStackTrace();
43         }
44     }
45
46     // Método para eliminar registros en la base de datos
47     public void eliminarRegistros() {
48         // Conexión a la base de datos
49         String url = "jdbc:mysql://localhost:3306/tarea5";
50         String user = "root";
51         String pass = "";
52         try {
53             Class.forName("com.mysql.cj.jdbc.Driver");
54             Connection con = DriverManager.getConnection(url, user, pass);
55             Statement stmt = con.createStatement();
56             String sql = "DELETE FROM vehiculos WHERE matricula = 'ABC123';";
57             stmt.executeUpdate(sql);
58             con.close();
59         } catch (Exception e) {
60             e.printStackTrace();
61         }
62     }
63
64     // Método para mostrar todos los registros en la base de datos
65     public void mostrarRegistros() {
66         // Conexión a la base de datos
67         String url = "jdbc:mysql://localhost:3306/tarea5";
68         String user = "root";
69         String pass = "";
70         try {
71             Class.forName("com.mysql.cj.jdbc.Driver");
72             Connection con = DriverManager.getConnection(url, user, pass);
73             Statement stmt = con.createStatement();
74             String sql = "SELECT * FROM vehiculos;";
75             ResultSet rs = stmt.executeQuery(sql);
76             while (rs.next()) {
77                 String matricula = rs.getString("matricula");
78                 String marca = rs.getString("marca");
79                 String modelo = rs.getString("modelo");
80                 int año = rs.getInt("año");
81                 String color = rs.getString("color");
82                 double precio = rs.getDouble("precio");
83                 System.out.println(matricula + " " + marca + " " + modelo + " " + año + " " + color + " " + precio);
84             }
85             con.close();
86         } catch (Exception e) {
87             e.printStackTrace();
88         }
89     }
90
91     // Método principal
92     public static void main(String[] args) {
93         Tarea5 tarea5 = new Tarea5();
94         tarea5.insertarRegistros();
95         tarea5.actualizarRegistros();
96         tarea5.eliminarRegistros();
97         tarea5.mostrarRegistros();
98     }
99 }

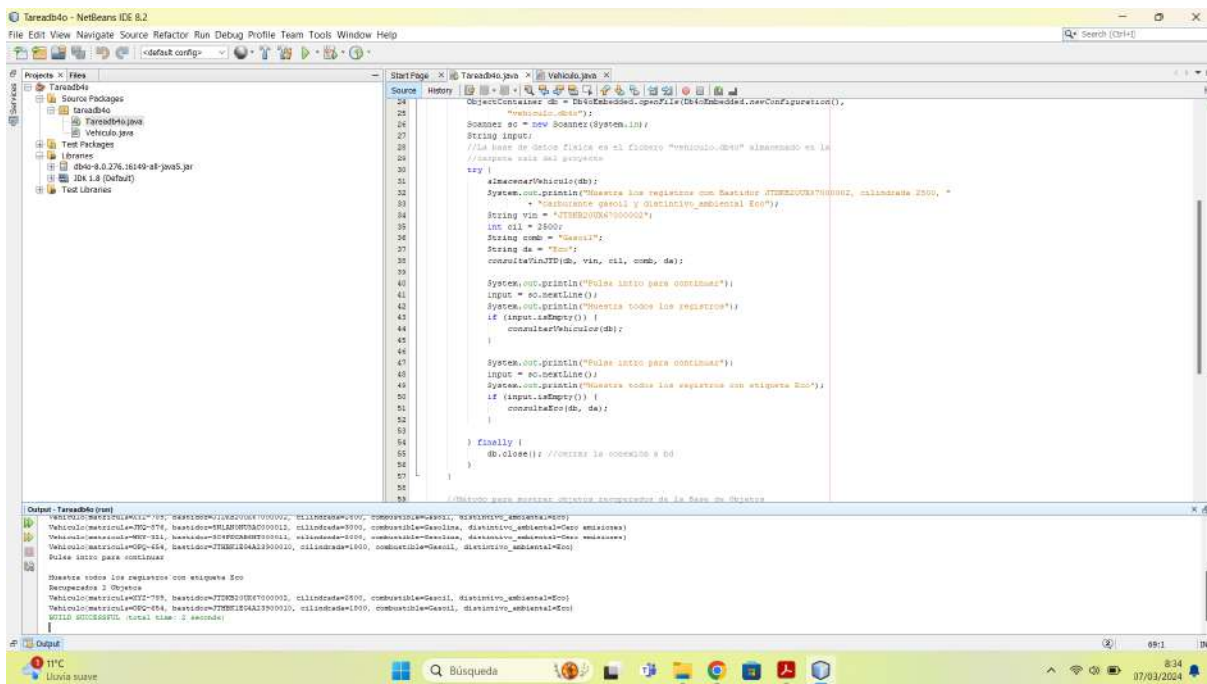
```

1.3.-Y por último los registros con distintivo_ambiental Eco 0,5

```

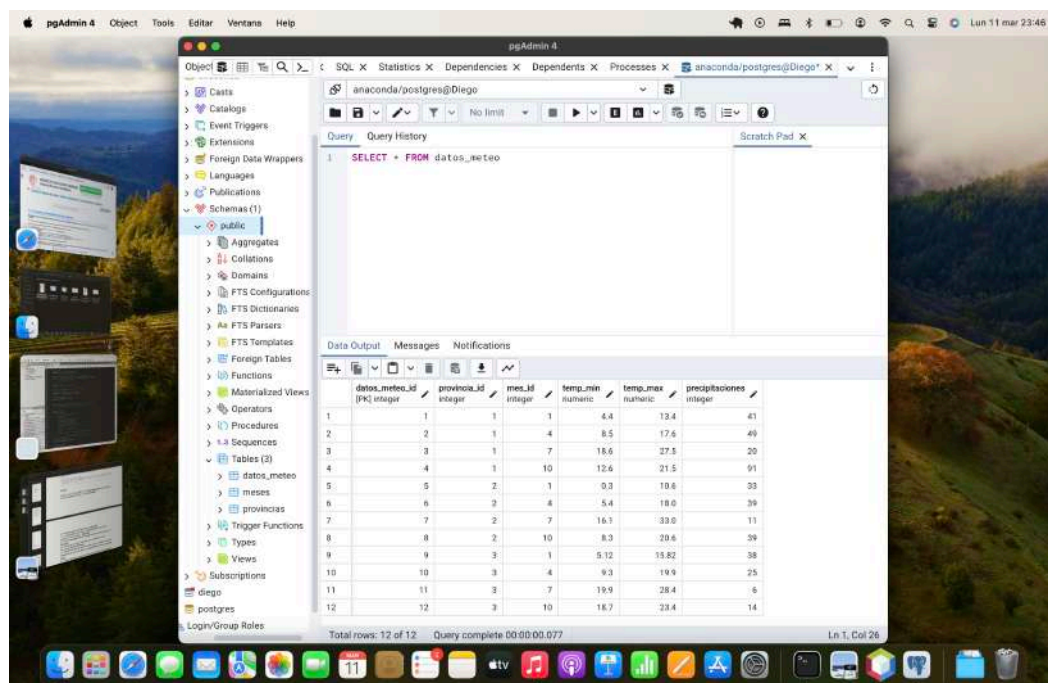
1  Tarea5.java
2  // Clase Tarea5
3  // Autor: Diego Manuel Carrasco Castaños
4  // Fecha: 07/03/2024
5
6  import java.sql.*;
7
8  public class Tarea5 {
9
10     // Método para insertar registros en la base de datos
11     public void insertarRegistros() {
12         // Conexión a la base de datos
13         String url = "jdbc:mysql://localhost:3306/tarea5";
14         String user = "root";
15         String pass = "";
16         try {
17             Class.forName("com.mysql.cj.jdbc.Driver");
18             Connection con = DriverManager.getConnection(url, user, pass);
19             Statement stmt = con.createStatement();
20             String sql = "INSERT INTO vehiculos (matricula, marca, modelo, año, color, precio, distintivo_ambiental) VALUES ('ABC123', 'Toyota', 'Corolla', 2020, 'Rojo', 15000, 'Eco');";
21             stmt.executeUpdate(sql);
22             con.close();
23         } catch (Exception e) {
24             e.printStackTrace();
25         }
26     }
27
28     // Método para actualizar registros en la base de datos
29     public void actualizarRegistros() {
30         // Conexión a la base de datos
31         String url = "jdbc:mysql://localhost:3306/tarea5";
32         String user = "root";
33         String pass = "";
34         try {
35             Class.forName("com.mysql.cj.jdbc.Driver");
36             Connection con = DriverManager.getConnection(url, user, pass);
37             Statement stmt = con.createStatement();
38             String sql = "UPDATE vehiculos SET marca = 'Ford' WHERE matricula = 'ABC123';";
39             stmt.executeUpdate(sql);
40             con.close();
41         } catch (Exception e) {
42             e.printStackTrace();
43         }
44     }
45
46     // Método para eliminar registros en la base de datos
47     public void eliminarRegistros() {
48         // Conexión a la base de datos
49         String url = "jdbc:mysql://localhost:3306/tarea5";
50         String user = "root";
51         String pass = "";
52         try {
53             Class.forName("com.mysql.cj.jdbc.Driver");
54             Connection con = DriverManager.getConnection(url, user, pass);
55             Statement stmt = con.createStatement();
56             String sql = "DELETE FROM vehiculos WHERE matricula = 'ABC123';";
57             stmt.executeUpdate(sql);
58             con.close();
59         } catch (Exception e) {
60             e.printStackTrace();
61         }
62     }
63
64     // Método para mostrar todos los registros en la base de datos
65     public void mostrarRegistros() {
66         // Conexión a la base de datos
67         String url = "jdbc:mysql://localhost:3306/tarea5";
68         String user = "root";
69         String pass = "";
70         try {
71             Class.forName("com.mysql.cj.jdbc.Driver");
72             Connection con = DriverManager.getConnection(url, user, pass);
73             Statement stmt = con.createStatement();
74             String sql = "SELECT * FROM vehiculos;";
75             ResultSet rs = stmt.executeQuery(sql);
76             while (rs.next()) {
77                 String matricula = rs.getString("matricula");
78                 String marca = rs.getString("marca");
79                 String modelo = rs.getString("modelo");
80                 int año = rs.getInt("año");
81                 String color = rs.getString("color");
82                 double precio = rs.getDouble("precio");
83                 String distintivo_ambiental = rs.getString("distintivo_ambiental");
84                 System.out.println(matricula + " " + marca + " " + modelo + " " + año + " " + color + " " + precio + " " + distintivo_ambiental);
85             }
86             con.close();
87         } catch (Exception e) {
88             e.printStackTrace();
89         }
90     }
91
92     // Método principal
93     public static void main(String[] args) {
94         Tarea5 tarea5 = new Tarea5();
95         tarea5.insertarRegistros();
96         tarea5.actualizarRegistros();
97         tarea5.eliminarRegistros();
98         tarea5.mostrarRegistros();
99     }
100 }

```

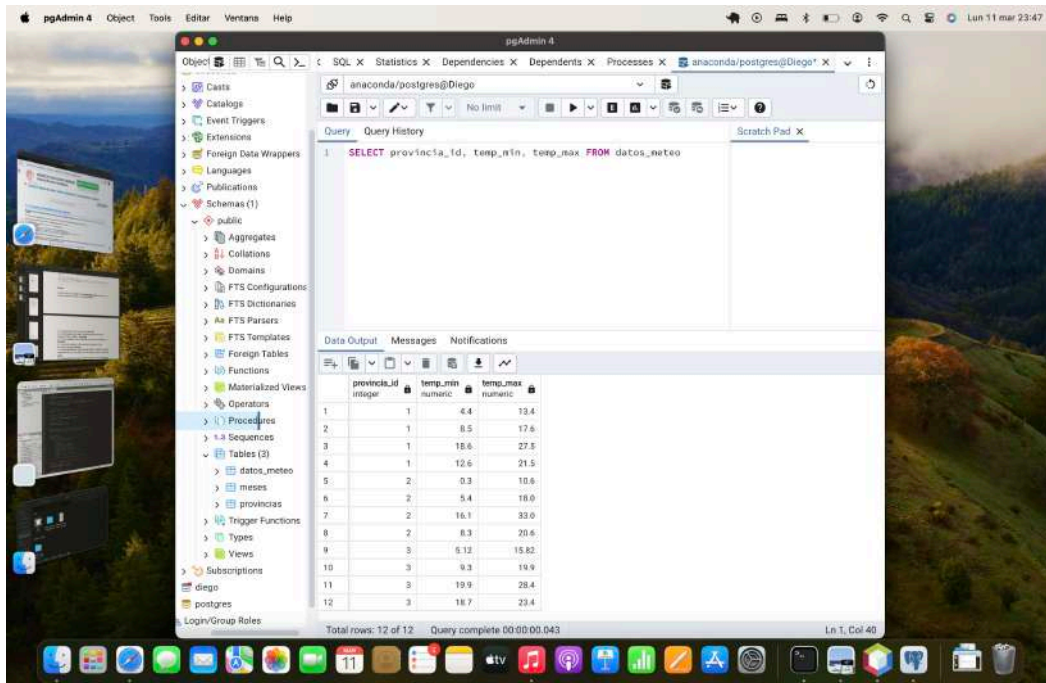
2.1.- Muestra los datos de la tabla datos meteo (0,25 puntos)

Para ello usaremos la sentencia `SELECT * FROM datos meteo`



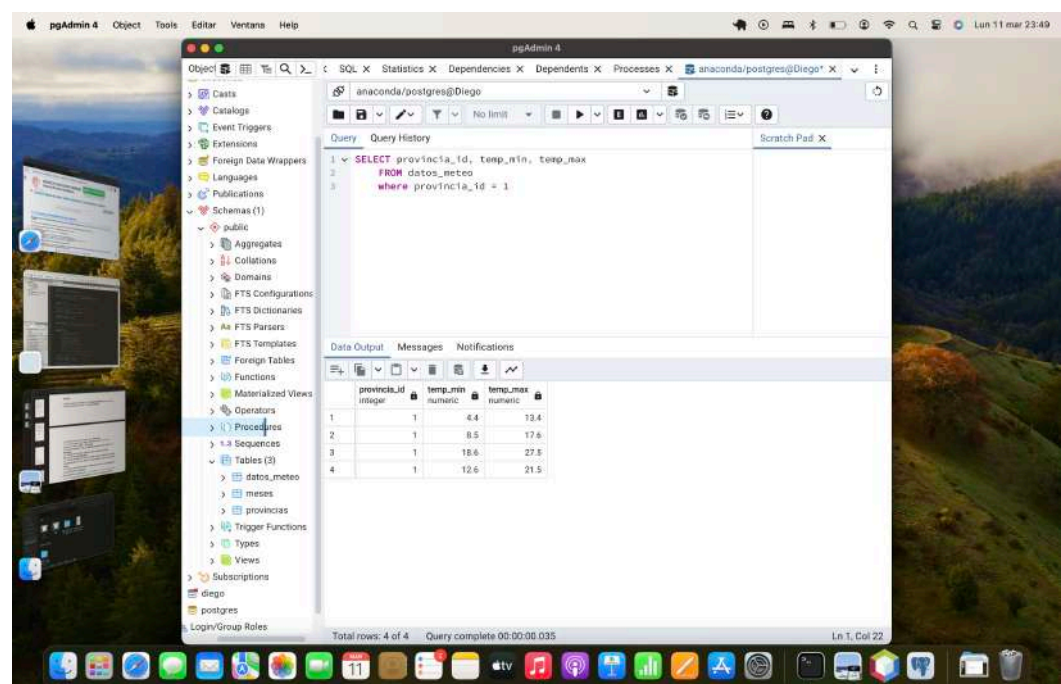
2.2.- Con la tabla `datos_meteo` muestra el identificador de las provincias con sus temperaturas mínimas y máximas. (0,25 puntos)

Para ello usaremos la sentencia `SELECT provincia_id, temp_min, temp_max FROM datos_meteo`



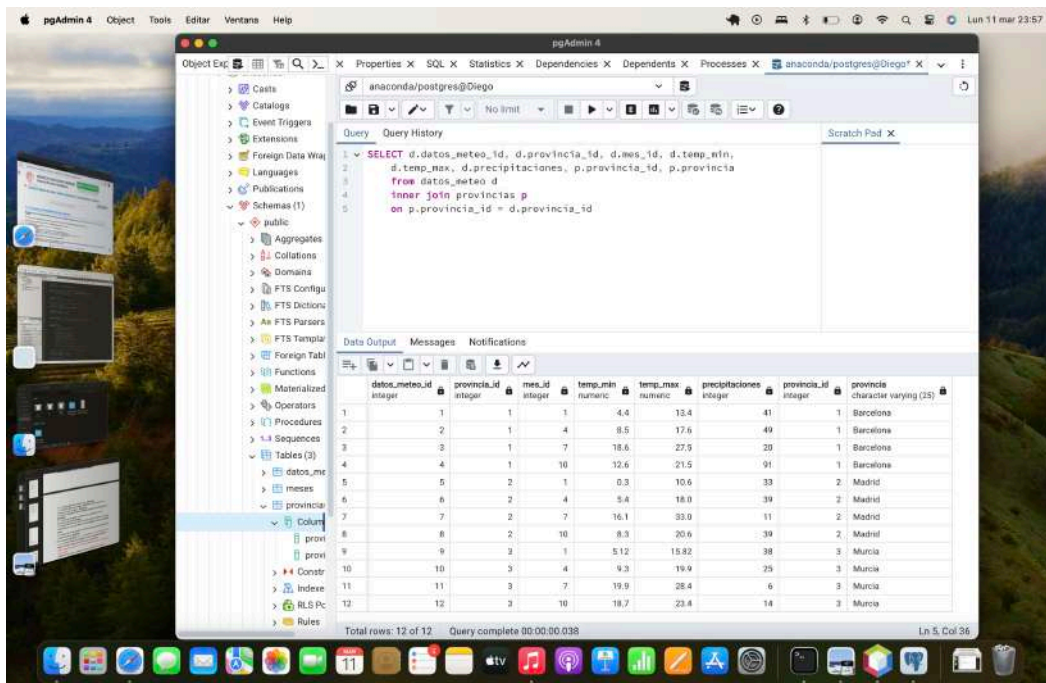
2.3.- Mostrar la temperatura mínima y máxima de la provincia cuyo identificador es el 1 (0,25 puntos)

Para ello utilizaremos la sentencia `SELECT provincia_id, temp_min, temp_max FROM datos_meteo WHERE provincia_id = 1`



2.4.- Muestra todos los datos de la tabla `datos_meteo` y provincias. Las tablas `datos_meteo` y `provincias` (0,25 puntos)

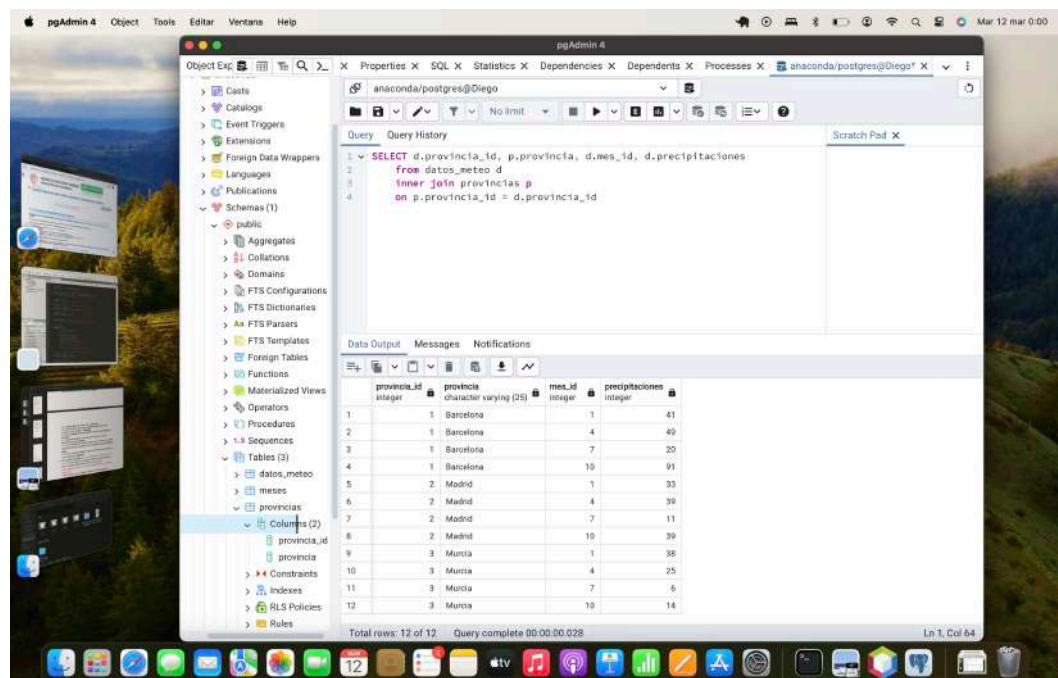
Para ello usaremos la sentencia `SELECT d.datos_meteo_id, d.provincia_id, d.mes_id, d.temp_min, d.temp_max, d.precipitaciones, p.provincia_id, p.provincia FROM datos_meteo d INNER JOIN provincias p ON p.provincia_id = d.provincia_id`



	datos_meteo_id	provincia_id	mes_id	temp_min	temp_max	precipitaciones	provincia_id	provincia
1	1	1	1	4.4	13.4	41	1	Barcelona
2	2	1	4	8.5	17.6	49	1	Barcelona
3	3	1	7	18.6	27.9	20	1	Barcelona
4	4	1	10	12.6	21.5	91	1	Barcelona
5	5	2	1	0.3	10.6	33	2	Madrid
6	6	2	4	5.4	18.0	39	2	Madrid
7	7	2	7	16.1	33.0	11	2	Madrid
8	8	2	10	8.3	20.6	39	2	Madrid
9	9	3	1	5.12	15.82	38	3	Murcia
10	10	3	4	9.3	19.9	25	3	Murcia
11	11	3	7	19.9	28.4	6	3	Murcia
12	12	3	10	18.7	23.4	14	3	Murcia

2.5.- Muestra el identificador de la provincia de `datos_meteo`, el nombre de las provincias, el identificador del mes y las precipitaciones. Con las tablas `datos_meteo` y `provincias`

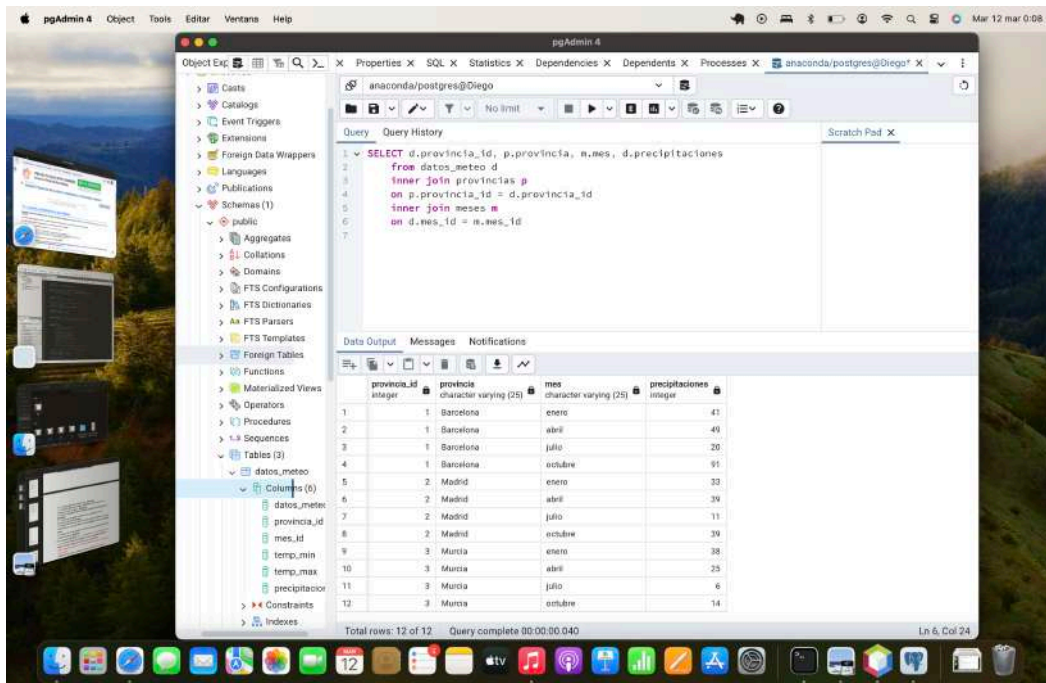
Para ello usaremos la sentencia `SELECT d.provincia_id, p.provincia, d.mes_id, d.precipitaciones FROM datos_meteo d INNER JOIN provincias p ON p.provincia_id = d.provincia_id`



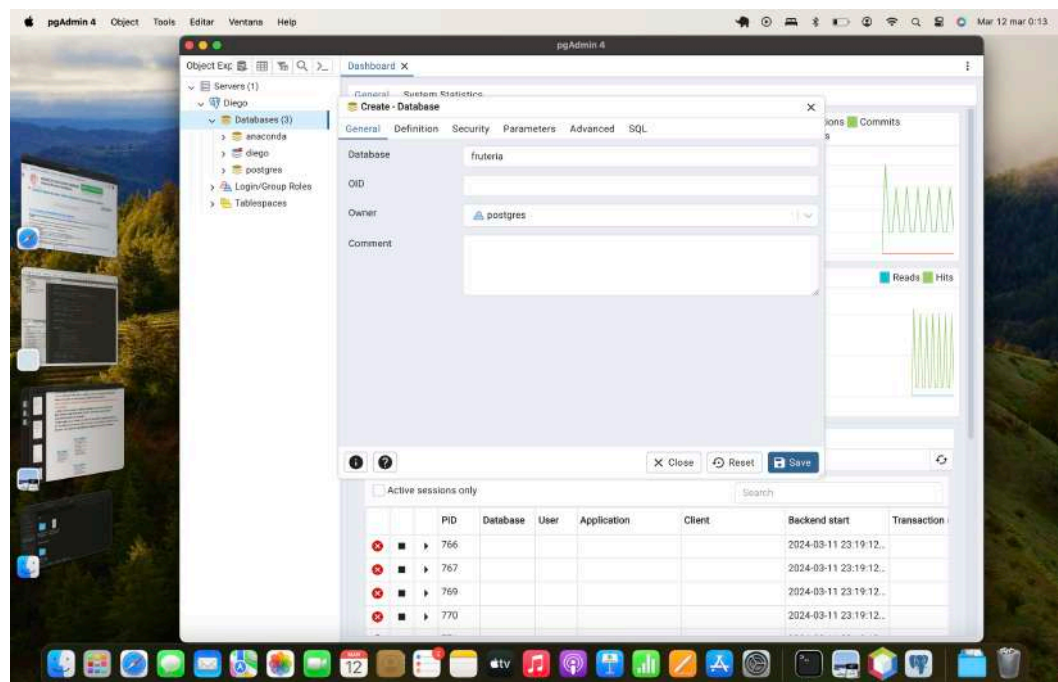
	provincia_id	provincia	mes_id	precipitaciones
1	1	Barcelona	1	41
2	1	Barcelona	4	49
3	1	Barcelona	7	20
4	1	Barcelona	10	91
5	2	Madrid	1	33
6	2	Madrid	4	39
7	2	Madrid	7	11
8	2	Madrid	10	39
9	3	Murcia	1	38
10	3	Murcia	4	25
11	3	Murcia	7	6
12	3	Murcia	10	14

2.6.- Modifica el sql anterior que aparezca también el nombre del mes no el identificador. Utiliza las 3 tablas ahora. (0,5 puntos)

Para ello usaremos la sentencia SELECT d.provincia_id, p.provincia, m.mes, d.precipitaciones FROM datos_meteo d INNER JOIN provincias p ON p.provincia_id = d.provincia_id INNER JOIN meses m ON d.mes_id = m.mes_id



3. Ahora ya pasamos a postgresql (sistema de gestión de bases de datos relacional orientado a objetos) Conectar) utilizando el driver JDBC y en pgAdmin 4 creamos una base de datos llamada frutería.



3.2.- Explica la siguiente sentencia y para qué se utiliza: (0,25 puntos)

```
Statement stmt = conn.createStatement();
```

La sentencia se utiliza para crear un objeto de tipo Statement.

conn es un objeto de tipo Connection que representa una conexión activa a una base de datos.

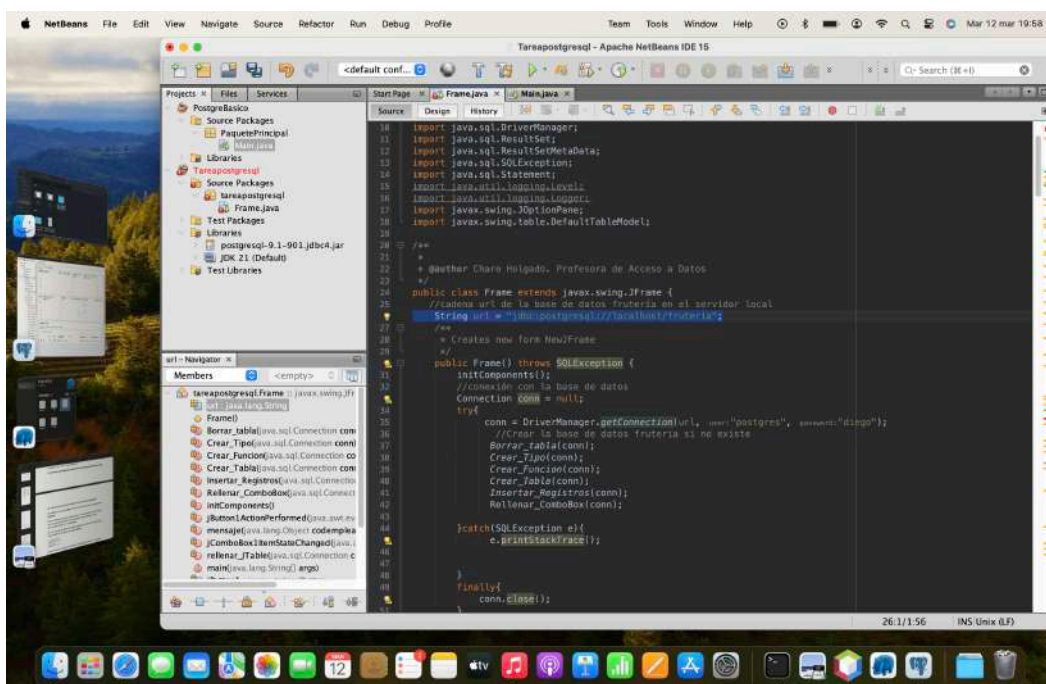
conn.createStatement() es un método que se llama en el objeto conn. Este método crea y devuelve un nuevo objeto de tipo Statement. Este objeto Statement se utilizará para enviar consultas SQL a la base de datos asociada con la conexión conn.

Statement stmt = conn.createStatement(); asigna el objeto Statement devuelto por conn.createStatement() a la variable stmt. Después de esta línea de código, la variable stmt puede utilizarse para enviar consultas SQL a la base de datos asociada con la conexión conn.

3.3.- Rellena el String de URL de la conexión a la base de datos frutería (0,5 puntos)

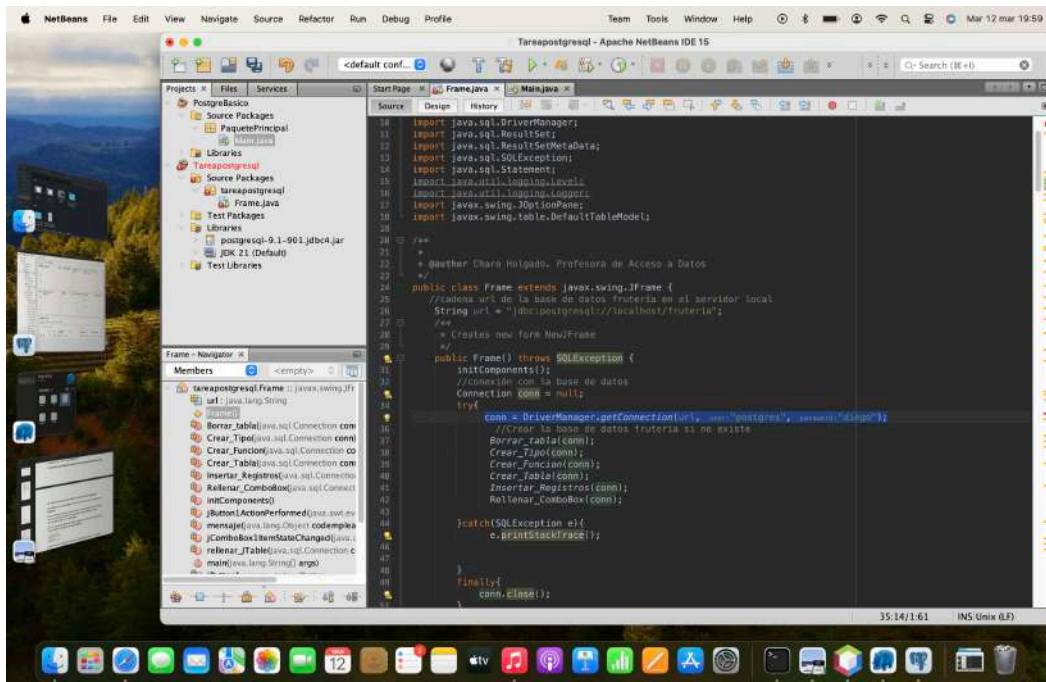
Para ello usaremos la siguiente sentencia:

```
String url = "jdbc:postgresql://localhost/fruteria";
```



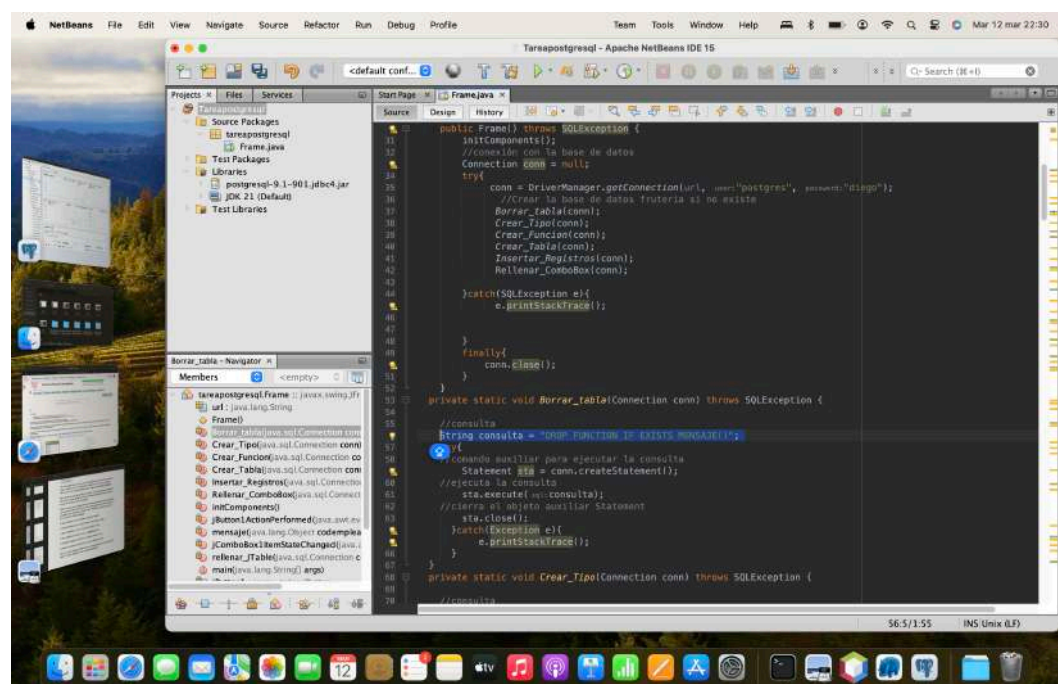
3.4.- Rellena los dos datos que faltan en los datos de tu conexión para que con esté correcta (0,5 puntos)

```
conn = DriverManager.getConnection(url, "postgres", "diego");
```



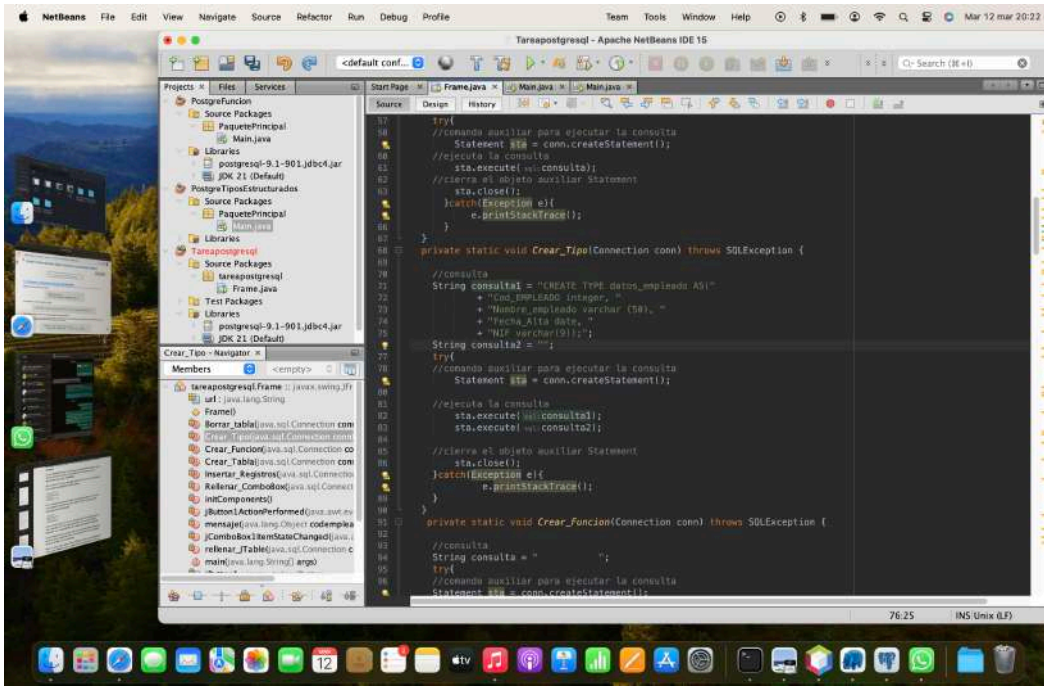
3.5. En el método `Borrar_tabla` escribe el String para: borrar la función mensaje que luego se construirá con los datos de la dirección del empleado, la tabla empleados y los tipos datos_empleado y datos_direccion (0,5 puntos)

```
String consulta = "DROP FUNCTION IF EXISTS mensaje()";
```



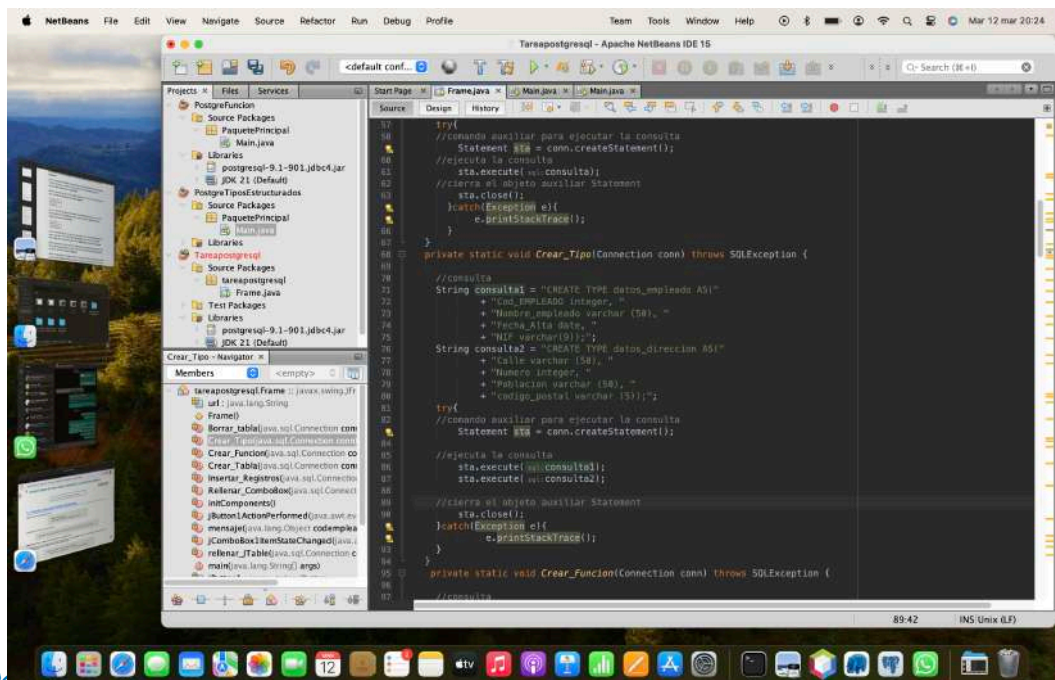
3.6.- En el método Crear_Tipo escribe o rellena el String con la consulta: Crea un tipo llamado datos_empleado con los siguientes campos (0,5 puntos)

```
String consulta1 = "CREATE TYPE datos_empleado AS
+ "( Nombre_empleado varchar(50), Fecha_Alta date,
NIF varchar(9));";
```



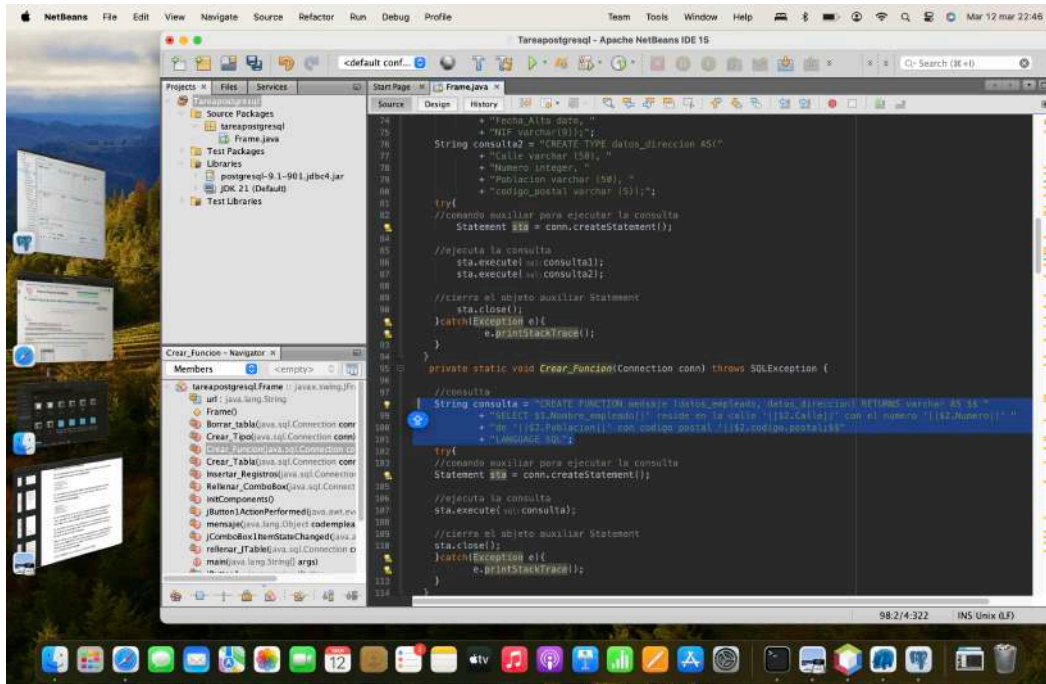
3.7.- En el método Crear_Tipo escribe o rellena el String llamado consulta2 con la consulta:

Crea un tipo llamado datos_direccion con los siguientes atributos. Créalo calle, número, código_postal y población (0,5 puntos)



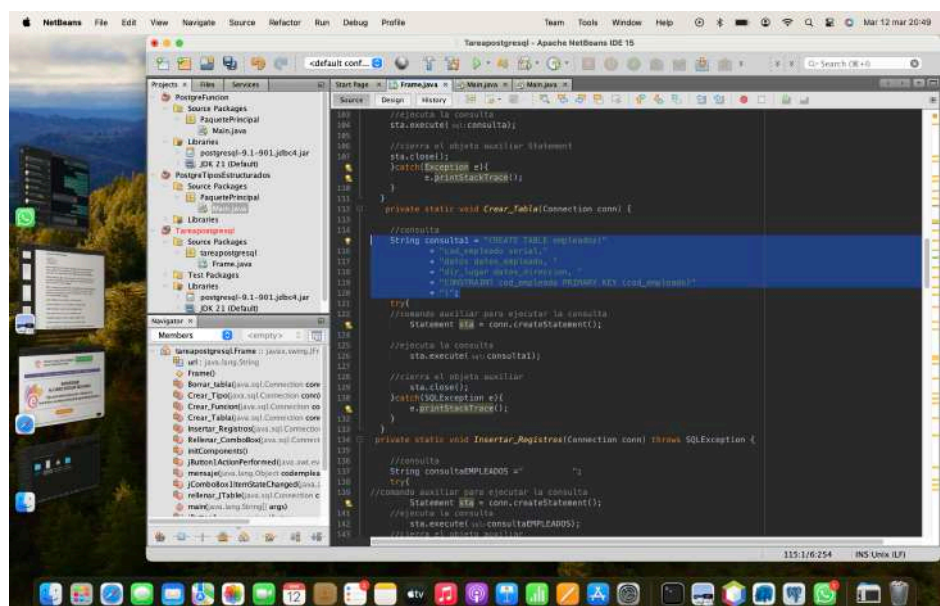
3.8.- En el método Crear_Funcion: Rellena el String para una función llamada mensaje para mostrar luego datos en jTextField del interfaz los datos de la dirección de un empleado seleccionado, con los campos de la calle, número, código postal y población.

```
String consulta2 = "CREATE TYPE datos_direccion AS" + "(Calle  
varchar(50), Numero integer, Poblacion varchar(50), codigo_postal  
varchar(5));";
```

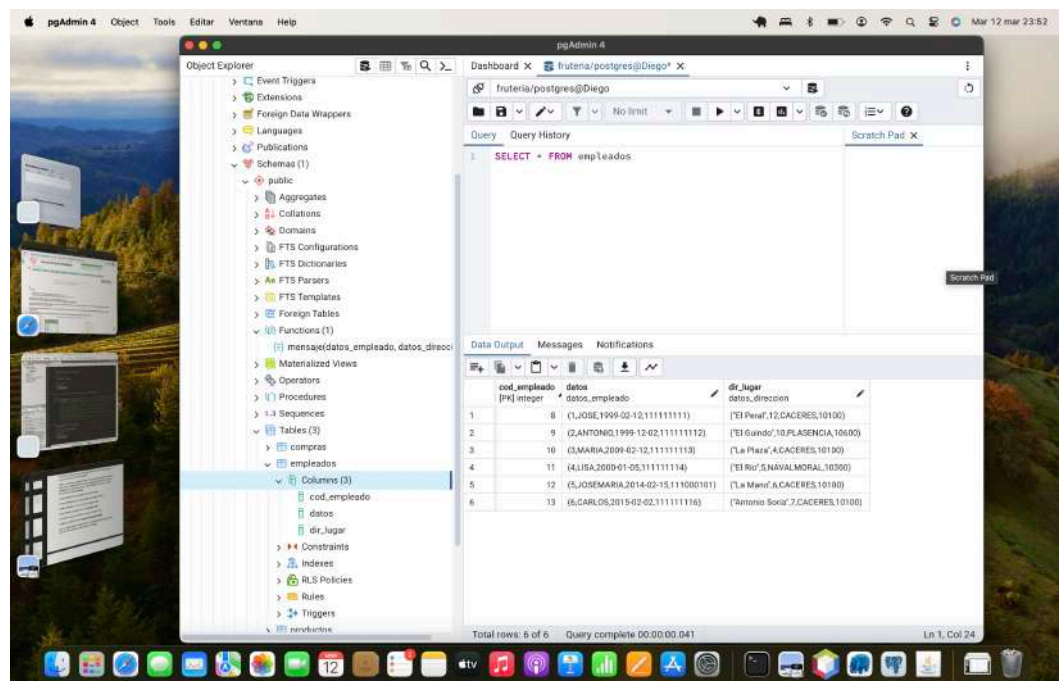
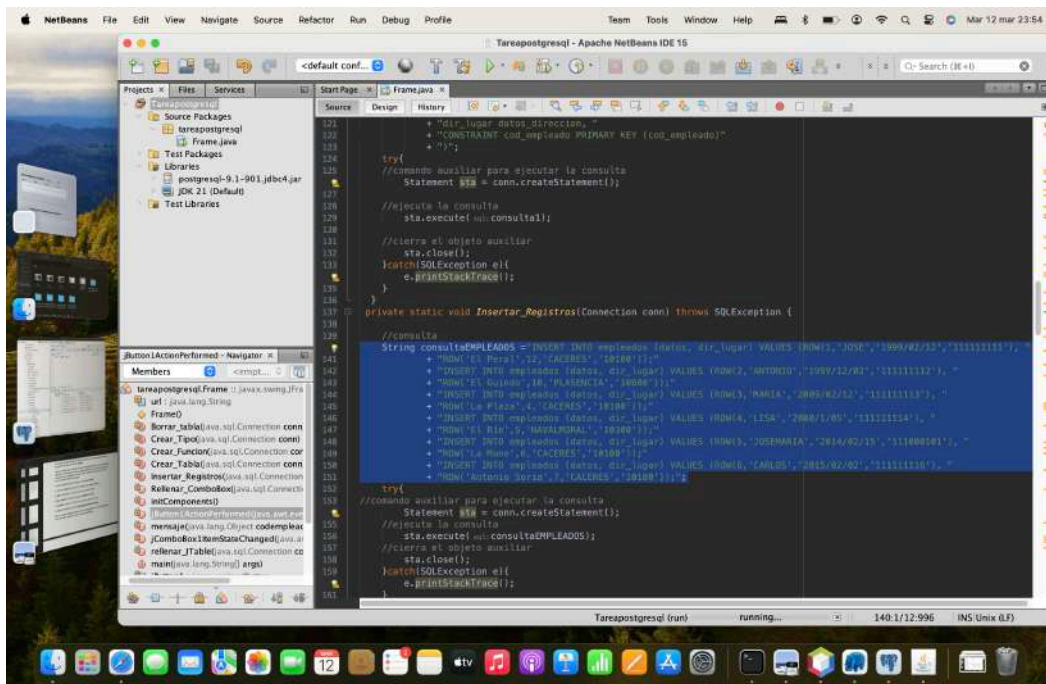


3.9.- En el método Crear_Tabla: Rellena el String para crear una tabla llamada empleados con tres campos: cod_empleado de tipo serial y clave primaria, datos de tipo datos_empleado y dir_lugar de tipo datos_direccion. (0,5 puntos)

```
String consulta1 = "CREATE TABLE empleados (cod_empleado serial  
PRIMARY KEY, datos datos_empleado, dir_lugar datos_direccion);";
```

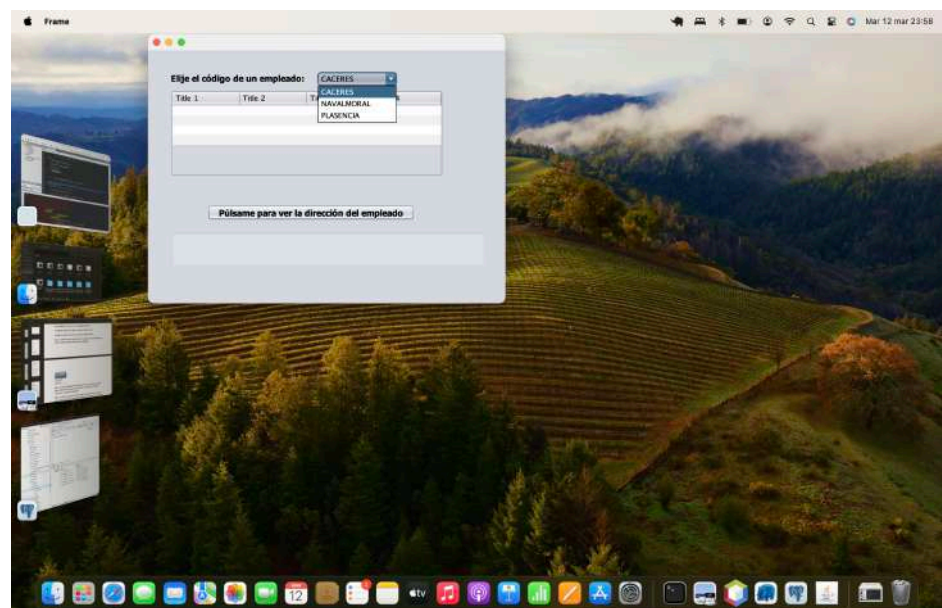
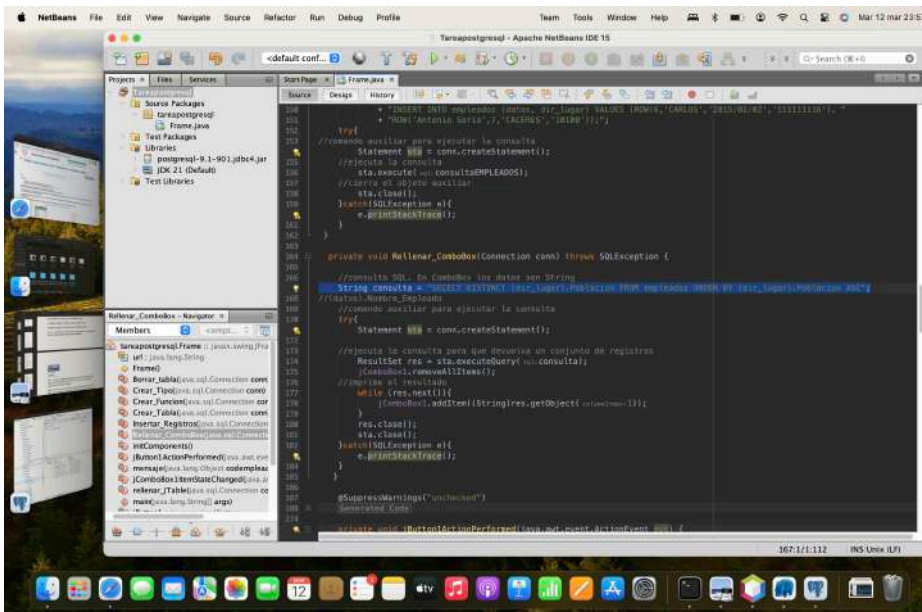


3.10.- En el método Insertar_Registros: Rellena el String para insertar en la tabla empleados los siguientes datos. Utilizando los dos tipos creados anteriormente (recuerda dos row por cada empleado), es decir, por ejemplo un row sería 'JOSE','1999/02/12','111111111' y el otro 'El Peral','12','CACERES','10100' (0,5 puntos)



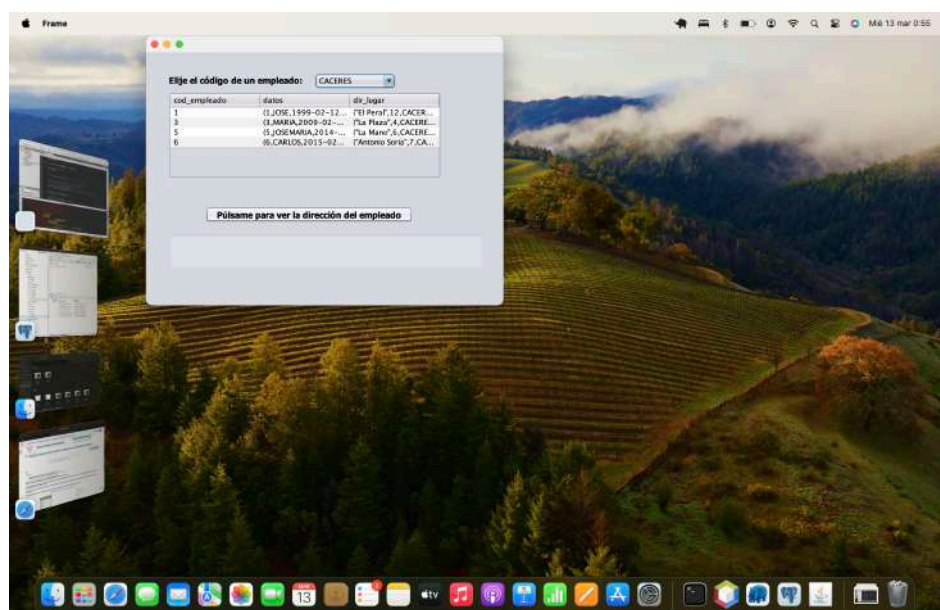
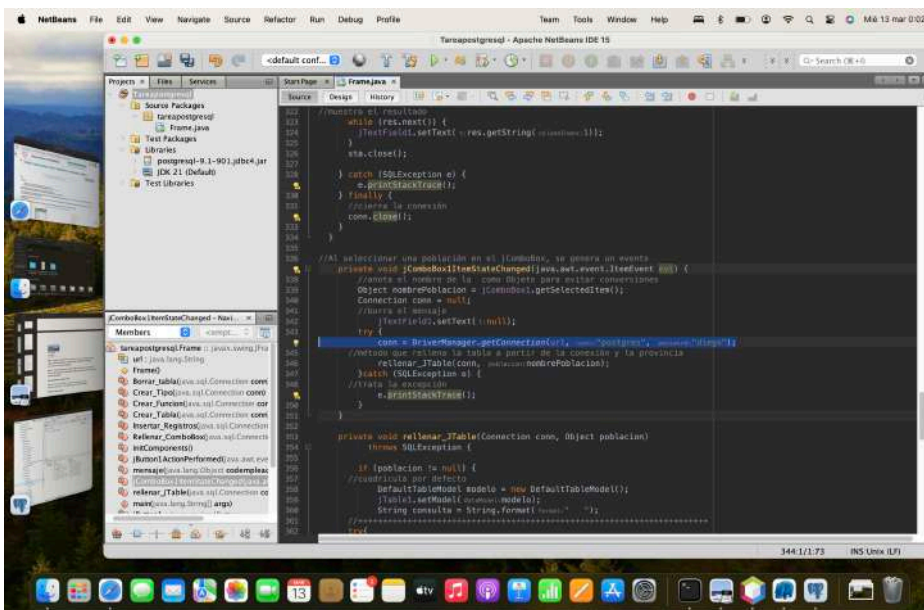
3.11.- En el método Rellenar_ComboBox: Rellena el String para seleccionar las poblaciones, sin repetir, de la tabla empleados en orden ascendente. (0,5 puntos)

```
String consulta = "SELECT DISTINCT (dir_lugar).Poblacion FROM  
empleados ORDER BY (dir_lugar).Poblacion ASC";
```



3.12.- En el método `jComboBoxItemStateChanged`: pon de nuevo tu conexión y en método `rellenar_JTable` rellena el String para seleccionar todos los empleados de la población seleccionada en el `JComboBox`. (0,5 puntos)

```
"SELECT DISTINCT (dir_lugar).Poblacion FROM empleados ORDER BY (dir_lugar).Poblacion ASC";
```



3.13.- En el método mensaje: pon de nuevo tu conexión y rellena el String de consulta para mostrar la función mensaje que correspondería al código del empleado que se ha seleccionado en la tabla. (0,5 puntos)

