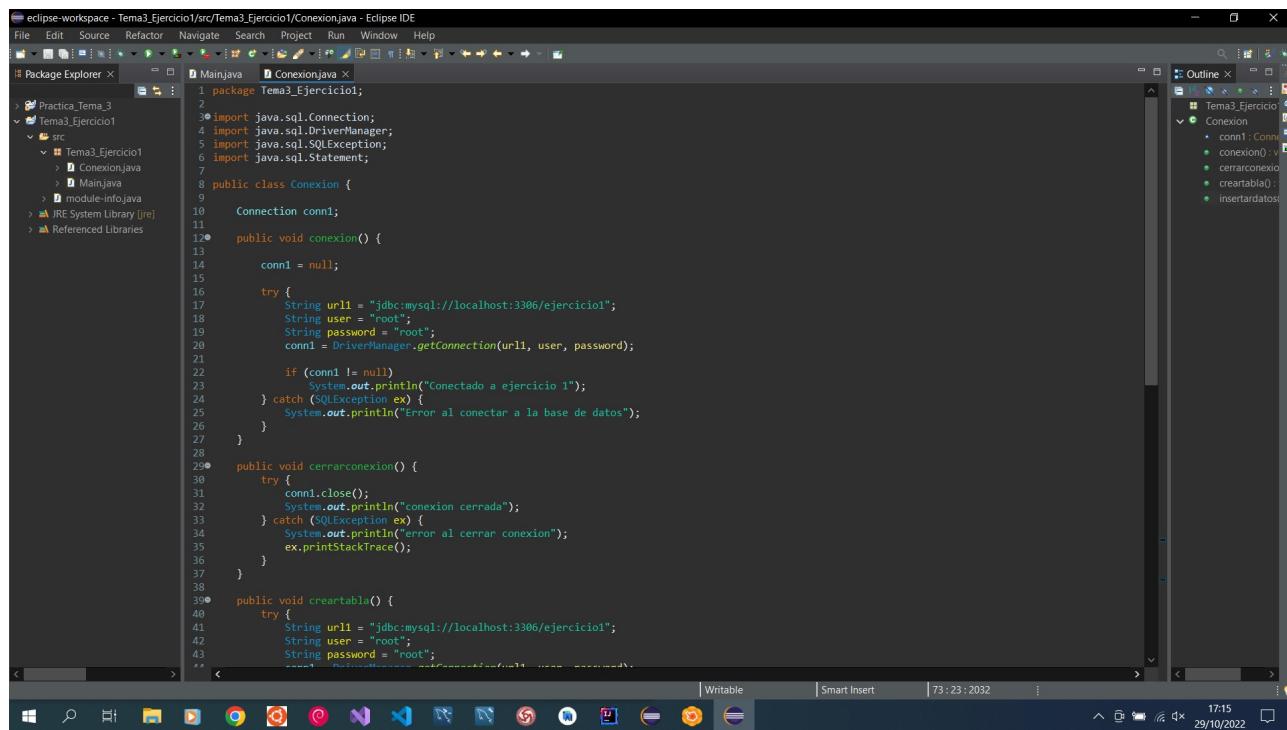


## Ejercicios Tema 3: BBDD Relacionales

Adjunto imágenes del programa y su ejecución, donde también mandaré el proyecto junto con el PDF.

### 3.4 – JAVA DATABASE CONNECTIVITY (JDBC)

a) Haz un programa que haga los cambios necesarios para que los contenidos de la tabla CLIENTES sean los siguientes: ('78901234X', 'NADALES', '44126'), ('89012345E', 'ROJAS', null), ('56789012B', 'SAMPER', '29730'), partiendo de los contenidos de la tabla resultantes de la ejecución del ejemplo anterior. El programa debe utilizar sentencias UPDATE y DELETE.



The screenshot shows the Eclipse IDE interface with the following details:

- File Menu:** File, Edit, Source, Refactor, Navigate, Search, Project, Run, Window, Help.
- Project Explorer:** Shows a workspace named "eclipse-workspace - Tema3\_Ejercicio1" containing "Practica\_Tema\_3", "Tema3\_Ejercicio1", and "Referenced Libraries".
- Code Editor:** Displays the file "Conexion.java" with the following code:

```
1 package Tema3_Ejercicio1;
2
3 import java.sql.Connection;
4 import java.sql.DriverManager;
5 import java.sql.SQLException;
6 import java.sql.Statement;
7
8 public class Conexion {
9
10     Connection conn1;
11
12     public void conexion() {
13
14         conn1 = null;
15
16         try {
17             String url1 = "jdbc:mysql://localhost:3306/ejercicio1";
18             String user = "root";
19             String password = "root";
20             conn1 = DriverManager.getConnection(url1, user, password);
21
22             if (conn1 != null)
23                 System.out.println("Conectado a ejercicio 1");
24         } catch (SQLException ex) {
25             System.out.println("Error al conectar a la base de datos");
26         }
27     }
28
29     public void cerrarconexion() {
30
31         try {
32             conn1.close();
33         } catch (SQLException ex) {
34             System.out.println("Error al cerrar conexión");
35             ex.printStackTrace();
36         }
37     }
38
39     public void creartabla() {
40
41         try {
42             String url1 = "jdbc:mysql://localhost:3306/ejercicio1";
43             String user = "root";
44             String password = "root";
45
46             Statement st = conn1.createStatement();
47
48             st.executeUpdate("CREATE TABLE IF NOT EXISTS clientes ("
49                         + "id VARCHAR(10) PRIMARY KEY, "
50                         + "nombre VARCHAR(50), "
51                         + "edad INT);");
52
53             st.close();
54         } catch (SQLException ex) {
55             System.out.println("Error al crear la tabla");
56         }
57     }
58
59     public void insertar() {
60
61         try {
62             String url1 = "jdbc:mysql://localhost:3306/ejercicio1";
63             String user = "root";
64             String password = "root";
65
66             Statement st = conn1.createStatement();
67
68             st.executeUpdate("INSERT INTO clientes VALUES ('78901234X', 'NADALES', 44126);");
69             st.executeUpdate("INSERT INTO clientes VALUES ('89012345E', 'ROJAS', null);");
70             st.executeUpdate("INSERT INTO clientes VALUES ('56789012B', 'SAMPER', 29730);");
71
72             st.close();
73         } catch (SQLException ex) {
74             System.out.println("Error al insertar los datos");
75         }
76     }
77
78     public void consultar() {
79
80         try {
81             String url1 = "jdbc:mysql://localhost:3306/ejercicio1";
82             String user = "root";
83             String password = "root";
84
85             Statement st = conn1.createStatement();
86
87             String sql = "SELECT * FROM clientes";
88
89             ResultSet rs = st.executeQuery(sql);
90
91             while (rs.next()) {
92                 System.out.println(rs.getString("id") + " " +
93                                 rs.getString("nombre") + " " +
94                                 rs.getInt("edad"));
95             }
96
97             st.close();
98         } catch (SQLException ex) {
99             System.out.println("Error al consultar los datos");
100        }
101    }
102}
```

The code implements a JDBC connection to a MySQL database named "ejercicio1" using the root user and password. It includes methods for connecting, closing the connection, creating a table named "clientes" with columns "id", "nombre", and "edad", inserting three records into the table, and querying all records from the table.

eclipse-workspace - Tema3\_Ejercicio1/src/Tema3\_Ejercicio1/Conexion.java - Eclipse IDE

```

File Edit Source Refactor Navigate Project Run Window Help
Package Explorer X MainJava ConexionJava X Outline X
Practica_Tema_3 Tema3_Ejercicio1 Tema3_Ejercicio1
src Tema3_Ejercicio1
MainJava ConexionJava MainJava ConexionJava
module-info.java cerrarconexion
IRE System Library [RE] insertardatos
Referenced Libraries
37 }
38 }
39 public void creartabla() {
40     try {
41         String url1 = "jdbc:mysql://localhost:3306/ejercicio1";
42         String user = "root";
43         String password = "root";
44         conn1 = DriverManager.getConnection(url1, user, password);
45
46         Statement stm = conn1.createStatement();
47         stm.execute("CREATE TABLE CLIENTES (DNI CHAR(9) NOT NULL,
48             + "APELLOS VARCHAR(32) NOT NULL, CP CHAR(5),
49             + " PRIMARY KEY (DNI');");
50
51         System.out.println("Tabla creado correctamente");
52     } catch (SQLException ex) {
53         System.out.println("Error al conectar a la base de datos");
54     } catch (Exception e) {
55         e.printStackTrace(System.err);
56     }
57 }
58
59 public void insertardatos() {
60     try {
61         String url1 = "jdbc:mysql://localhost:3306/ejercicio1";
62         String user = "root";
63         String password = "root";
64         conn1 = DriverManager.getConnection(url1, user, password);
65
66         Statement stm = conn1.createStatement();
67         stm.executeUpdate("INSERT INTO CLIENTES (DNI, APELLIDOS, CP)
68             + "(78901234X", "NADALES", "44126"),
69             + "(89012345E", "HOTAS", null),
70             + "(56789012B", "SAMPER", "29730"),
71             + "(09876543K", "LAMQUIZ", null);");
72
73         stm.executeUpdate("DELETE FROM CLIENTES WHERE DNI = '09876543K';");
74
75         System.out.println("Filas insertadas correctamente");
76     } catch (SQLException ex) {
77         System.out.println("Error al conectar a la base de datos");
78     } catch (Exception e) {
79         e.printStackTrace(System.err);
80     }
81 }
82
83

```

eclipse-workspace - Tema3\_Ejercicio1/src/Tema3\_Ejercicio1/Conexion.java - Eclipse IDE

```

File Edit Source Refactor Navigate Project Run Window Help
Package Explorer X MainJava ConexionJava X Outline X
Practica_Tema_3 Tema3_Ejercicio1 Tema3_Ejercicio1
src Tema3_Ejercicio1
MainJava ConexionJava MainJava ConexionJava
module-info.java cerrarconexion
IRE System Library [RE] insertardatos
Referenced Libraries
41 }
42 }
43 public void creartabla() {
44     try {
45         String url1 = "jdbc:mysql://localhost:3306/ejercicio1",
46             + "String user = "root";
47             + "String password = "root";
48             conn1 = DriverManager.getConnection(url1, user, password);
49
50         Statement stm = conn1.createStatement();
51         stm.execute("CREATE TABLE CLIENTES (DNI CHAR(9) NOT NULL,
52             + "APELLOS VARCHAR(32) NOT NULL, CP CHAR(5),
53             + " PRIMARY KEY (DNI');");
54
55         System.out.println("Tabla creado correctamente");
56     } catch (SQLException ex) {
57         System.out.println("Error al conectar a la base de datos");
58     } catch (Exception e) {
59         e.printStackTrace(System.err);
60     }
61 }
62
63 public void insertardatos() {
64     try {
65         String url1 = "jdbc:mysql://localhost:3306/ejercicio1";
66         String user = "root";
67         String password = "root";
68         conn1 = DriverManager.getConnection(url1, user, password);
69
70         Statement stm = conn1.createStatement();
71         stm.executeUpdate("INSERT INTO CLIENTES (DNI, APELLIDOS, CP)
72             + "(78901234X", "NADALES", "44126"),
73             + "(89012345E", "HOTAS", null),
74             + "(56789012B", "SAMPER", "29730"),
75             + "(09876543K", "LAMQUIZ", null);");
76
77         stm.executeUpdate("DELETE FROM CLIENTES WHERE DNI = '09876543K';");
78
79         System.out.println("Filas insertadas correctamente");
80     } catch (SQLException ex) {
81         System.out.println("Error al conectar a la base de datos");
82     } catch (Exception e) {
83         e.printStackTrace(System.err);
84     }
85 }
86
87

```

The screenshot shows the Eclipse IDE interface with the following details:

- Title Bar:** eclipse-workspace - Tema3\_Ejercicio1/src/Tema3\_Ejercicio1/Main.java - Eclipse IDE
- Menu Bar:** File Edit Source Refactor Navigate Search Project Run Window Help
- Toolbars:** Standard, Java, Java Editor, Java Outline, Java Problems, Java Javadoc, Java Declaration, Java Coverage.
- Left Sidebar:** Package Explorer (Practica\_Tema\_3, Tema3\_Ejercicio1, src folder containing Tema3\_Ejercicio1.java, Conexion.java, Main.java, module-info.java, JRE System Library [JRE], Referenced Libraries).
- Central Area:** Editor tab for Main.java (Content Type: Java). The code is as follows:

```
1 package Tema3_Ejercicio1;
2
3 public class Main {
4
5     public static void main(String[] args) {
6         Conexion constructor = new Conexion();
7
8         constructor.conexion();
9         constructor.creatatabla();
10        constructor.insertardatos();
11    }
12 }
```

- Right Sidebar:** Outline (Tema3\_Ejercicio1, Main, main(String[]))
- Bottom Status Bar:** Writable, Smart Insert, 1:9:8, 17:15, 29/10/2022

The screenshot shows the Eclipse IDE interface with the following details:

- Title Bar:** eclipse-workspace - Tema3\_Ejercicio1/src/Tema3\_Ejercicio1/Main.java - Eclipse IDE
- Menu Bar:** File Edit Source Refactor Navigate Search Project Run Window Help
- Toolbars:** Standard, Java, Java Editor, Java Outline, Java Problems, Java Javadoc, Java Declaration, Java Coverage.
- Left Sidebar:** Package Explorer (Practica\_Tema\_3, Tema3\_Ejercicio1, src folder containing Tema3\_Ejercicio1.java, Conexion.java, Main.java, module-info.java, JRE System Library [JRE], Referenced Libraries).
- Central Area:** Editor tab for Main.java (Content Type: Java). The code is identical to the previous screenshot.
- Bottom Area:** Console tab (Content Type: Text). The output is:

```
Conectado a ejercicio 1
Tabla creado correctamente
Filas insertadas correctamente
```

- Bottom Status Bar:** 17:17, 29/10/2022

The screenshot shows the Eclipse IDE interface. In the center, the code editor displays a Main.java file with the following content:

```
package Tema3_Ejercicio1;
public class Main {
    public static void main(String[] args) {
        Conexion constructor = new Conexion();
        constructor.conexion();
        constructor.crearTabla();
        constructor.insertardatos();
        constructor.cerrarconexion();
    }
}
```

The output window below the editor shows the results of the program's execution:

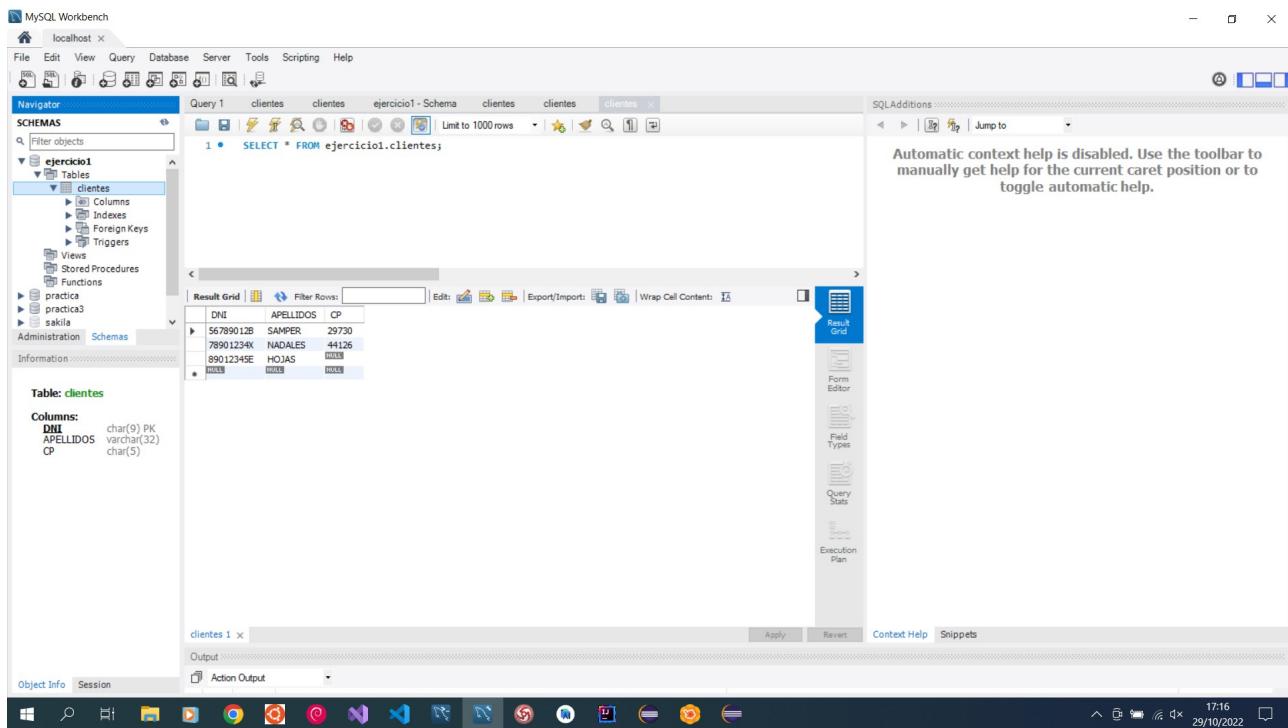
```
Conectado a ejercicio 1
Tabla creado correctamente
Filas insertadas correctamente
```

The screenshot shows the MySQL Workbench interface. The left sidebar displays the Navigator and Schemas sections, with the schema 'ejercicio1' selected. The central area shows a query editor with the following SQL statement:

```
1 • SELECT * FROM ejercicio1.clientes;
```

The result grid displays the data from the 'clientes' table:

DNI	APELLIDOS	CP
09876543K	LAMIQUEZ	29730
56789012B	SAMPER	29730
78901234X	NADALES	44126
89012345E	HOJAS	NULL
NULL	NULL	NULL



b) El código postal (columna CP) está definido con tipo CHAR(5) en la tabla CLIENTES, pero es siempre un número entero. ¿Se podría utilizar getInt() en lugar de getString() para recuperar su valor? Cambia el programa y verifica tu hipótesis, o justifica los resultados si no son los que esperabas.

Se puede realizar y funciona pero no debidamente, dado que su valor real es un Char; un carácter puede tener valor null mientras que un Integer no hay valor null y cuando el char es null saca el valor Integer “0”.

```

public void consultasclientes() {
    try {
        String url = "jdbc:mysql://localhost:3306/ejercicio1";
        String user = "root";
        String password = "root";
        conn1 = DriverManager.getConnection(url, user, password);
        // PreparedStatement stmt = conn1.prepareStatement("");
        Statement stm = conn1.createStatement();
        ResultSet rs = stm.executeQuery("SELECT * FROM CLIENTES");
        int i = 1;
        while (rs.next()) {
            System.out.println("(" + (i++) + ")");
            System.out.println("DNI: " + rs.getString("DNI"));
            System.out.println("APELLIDOS: " + rs.getString("APELLIDOS"));
            System.out.println("CP: " + rs.getChar("CP"));
        }
    } catch (SQLException ex) {
        System.out.println("Error al conectar a la base de datos");
    } catch (Exception e) {
        e.printStackTrace(System.err);
    }
}

```

Output:

```

DNI: 78901234X
DNI: NADALES
DNI: 44126
[3]
DNI: 89012345E
DNI: HOJAS
DNI: 0

```

### 3.5 – SENTENCIAS PREPARADAS, TRANSACCIONES Y CLAVES AUTOGENERADAS

a) Escribe un programa que muestre los datos de varios clientes, o todos, de la tabla CLIENTES1. El programa debe utilizar una sentencia preparada para la consulta SELECT \* FROM CLIENTES1 WHERE DNI=? Debe realizarse una consulta para cada cliente, especificando su DNI, y obtener los datos del ResultSet resultante, que solo tendrá una fila, al ser el acceso por clave primaria.

The screenshot shows the Eclipse IDE interface with the following details:

- Project Explorer:** Shows the project structure with packages like Práctica\_Tema\_3 and Tema3\_Ejercicio1, and source files like Conexion.java and Main.java.
- Main.java:** Contains Java code to establish a database connection and execute a query to print client information. The code uses prepared statements and handles exceptions.
- Console Output:** Displays the output of the program, which prints two rows of client data (DNI, APELLIDOS, CP) for two different clients.
- Bottom Status Bar:** Shows the date (29/10/2022), time (18:06), and file path (C:\Users\david.p2\pool\plugins\org.eclipse.jdt.openjdk.hotspot.jre.full.win32.x86\_64\_17.0.4.v20221004-1257\jre\bin\javaw.exe).

The screenshot shows the Eclipse IDE interface with the following details:

- Project Explorer:** Shows the project structure with packages like Práctica\_Tema\_3 and Tema3\_Ejercicio1, and source files like Conexion.java and Main.java.
- Main.java:** Contains Java code to establish a database connection and execute queries to print client information for specific DNI values ('78901234X', '89012345E', and '56789012B'). The code uses prepared statements and handles exceptions.
- Console Output:** Displays the output of the program, which prints client data for three specific clients (DNI, APELLIDOS, CP) corresponding to the specified DNI values.
- Bottom Status Bar:** Shows the date (29/10/2022), time (20:30), and file path (C:\Users\david.p2\pool\plugins\org.eclipse.jdt.openjdk.hotspot.jre.full.win32.x86\_64\_17.0.4.v20221004-1257\jre\bin\javaw.exe).

The screenshot shows the Eclipse IDE interface with the following details:

- Project Explorer:** Shows the project structure with files like Main.java and Conexion.java.
- Code Editor:** Displays Java code for connecting to a database and executing queries to print client information (DNI, APELLIDOS, CP).
- Console Output:** Shows the execution results:

```
Conectado a ejercicio 1
DNI: 78901234X
APELLOIDS: NADALES
CP: 44126

DNI: 89012345E
APELLOIDS: HOJAS
CP: null
```
- Bottom Status Bar:** Shows the date (29/10/2022), time (20:31), and file path (C:\Users\David\P...\Tema3\_Ejercicio1\src\Tema3\_Ejercicio1\Conexion.java).

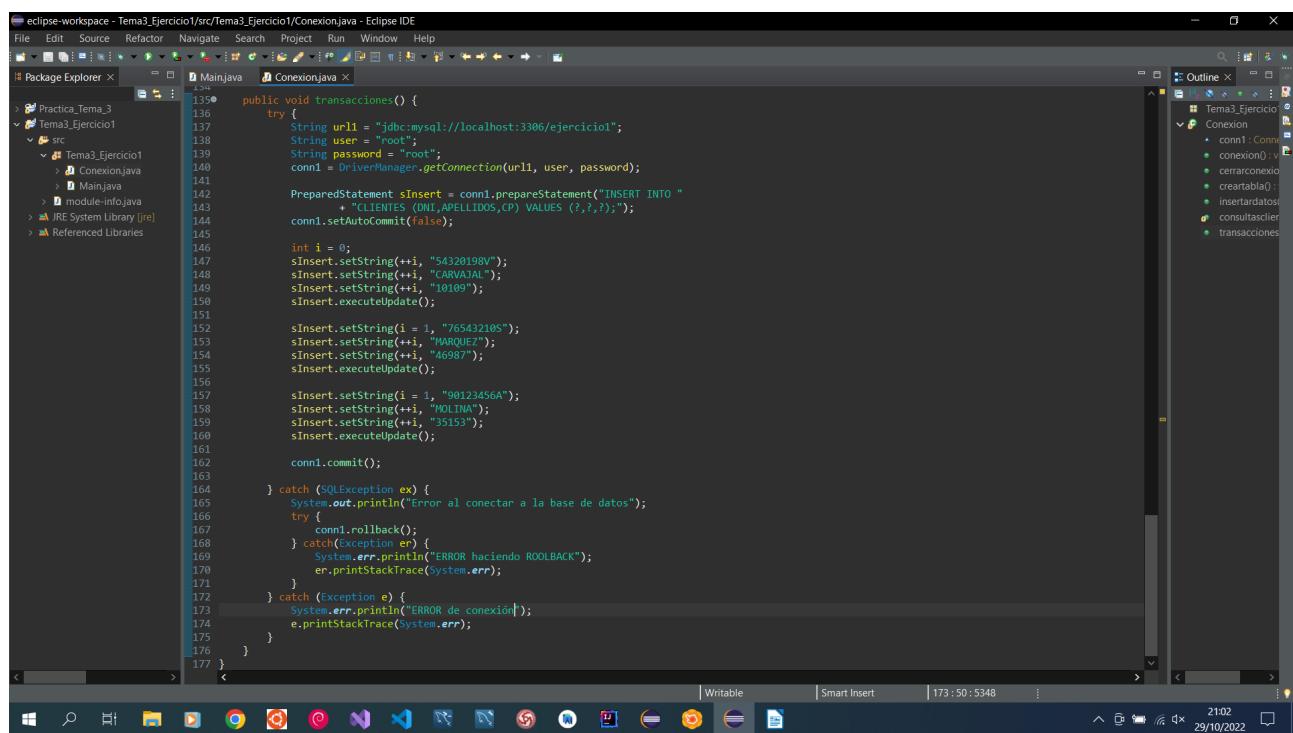
The screenshot shows the Eclipse IDE interface with the following details:

- Project Explorer:** Shows the project structure with files like Main.java and Conexion.java.
- Code Editor:** Displays Java code for connecting to a database and executing queries to print client information (DNI, APELLIDOS, CP). The code is identical to the one in the first screenshot.
- Console Output:** Shows the execution results:

```
Conectado a ejercicio 1
DNI: 78901234X
APELLOIDS: NADALES
CP: 44126

DNI: 89012345E
APELLOIDS: HOJAS
CP: null
```
- Bottom Status Bar:** Shows the date (29/10/2022), time (20:31), and file path (C:\Users\David\P...\Tema3\_Ejercicio1\src\Tema3\_Ejercicio1\Conexion.java).

b) Comprueba las diferencias entre el programa anterior, en el que se agrupan todas la inserciones de registros con transacciones, y otro programa igual pero sin transacciones. Primero hay que hacer una copia del programa y eliminar el código que gestiona las transacciones (setAutoCommit, commit y rollback). Después se trata de probar ambos programas con un mismo conjunto de datos inicial en la tabla CLIENTES1, para comprobar la manera distinta en que se comportan. El conjunto de datos inicial y el programa podrían ser tales que las dos primeras inserciones se realizaran sin problemas, pero la última no, por haber ya en la tabla un cliente con ese DNI. El programa con transacciones no insertaría ningún registro. El programa sin transacciones insertará registros hasta que se produjera un error. La creación del conjunto de datos inicial debe automatizarse. Se puede hacer mediante una secuencia de sentencias SQL, que se pueden guardar en un fichero de texto, y ejecutar con el intérprete de SQL, o mediante un programa. En esas secuencias podrían borrarse todos los contenidos de la tabla con una sentencia DELETE y añadirse varias filas con sentencias INSERT.



The screenshot shows the Eclipse IDE interface with the following details:

- Project Explorer:** Shows the project structure with packages like Práctica\_Tema\_3, Tema3\_Ejercicio1, and Tema3\_Ejercicio1/src. Under Tema3\_Ejercicio1/src, there are files Mainjava and Conexion.java.
- Code Editor:** Displays the content of the Conexion.java file. The code is a Java class named Conexión with a public void method named transacciones(). The method establishes a connection to a MySQL database at localhost:3306, sets the user to 'root', and the password to 'root'. It then prepares an INSERT statement for the 'CLIENTES' table with columns (DNI, APELLIDOS, CP). The loop starts with i=0 and increments by 1 for each row. The values for DNI, APELLIDOS, and CP are set sequentially: 54320198V, CARVATIL, 10109; 76543210S, MARQUEZ, 46987; 90123456A, MOLINA, 35153. Finally, the conn1.commit() method is called to commit the transaction.
- Outline View:** Shows the class structure with methods like conexión(), cerrarconexion(), creartabla0(), insertardatos(), consultascleric(), and transacciones().
- Bottom Status Bar:** Shows the current time as 17:50:5348 and the date as 29/10/2022.

```

eclipse-workspace - Tema3_Ejercicio1/src/Tema3_Ejercicio1/Conexion.java - Eclipse IDE
File Edit Source Refactor Navigate Project Run Window Help
Package Explorer X MainJava Conexion.java X e.printStackTrace(System.err);
...
134 }
135 public void transacciones() {
136     try {
137         String url = "jdbc:mysql://localhost:3306/ejercicio1";
138         String user = "root";
139         String password = "root";
140         conn1 = DriverManager.getConnection(url1, user, password);
141
142         PreparedStatement sInsert = conn1.prepareStatement("INSERT INTO "
143             + "CLIENTES (DNI, APELLIDOS, CP) VALUES (?, ?, ?);");
144         conn1.setAutoCommit(false);
145
146         int i = 0;
147         sInsert.setString(i++, "54320198V");
148         sInsert.setString(i++, "CARVAJAL");
149         sInsert.setString(i++, "10109");
150         sInsert.executeUpdate();
151
152         sInsert.setString(i + 1, "76543210S");
153         sInsert.setString(i + 1, "MARQUEZ");
154         sInsert.setString(i + 1, "46987");
155         sInsert.executeUpdate();
156
157         sInsert.setString(i + 1, "90123456A");
158         sInsert.setString(i + 1, "VOLINA");
159         sInsert.setString(i + 1, "35153");
160         sInsert.executeUpdate();
161
162         conn1.commit();
163     } catch (SQLException ex) {
164         System.out.println("Error al conectar a la base de datos");
165         try {
166             conn1.rollback();
167         } catch (Exception er) {
168             System.err.println("ERROR haciendo ROLBACK");
169             er.printStackTrace(System.err);
170         }
171     } catch (Exception e) {
172         System.err.println("ERROR de conexión");
173         e.printStackTrace(System.err);
174     }
    
```

Conectado a ejercicio 1  
DNI: 78901234X  
APELIDOS: NADALES  
CP: 44126  
  
DNI: 89012345E  
APELIDOS: HOJAS  
CP: null  
  
DNI: 56789012B  
APELIDOS: SAMPER  
CP: 29730

Error al conectar a la base de datos

```

eclipse-workspace - Tema3_Ejercicio1/src/Tema3_Ejercicio1/Conexion.java - Eclipse IDE
File Edit Source Refactor Navigate Project Run Window Help
Package Explorer X MainJava Conexion.java X e.printStackTrace(System.err);
...
134 }
135 public void transacciones() {
136     try {
137         String url = "jdbc:mysql://localhost:3306/ejercicio1";
138         String user = "root";
139         String password = "root";
140         conn1 = DriverManager.getConnection(url1, user, password);
141
142         PreparedStatement sInsert = conn1.prepareStatement("INSERT INTO "
143             + "CLIENTES (DNI, APELLIDOS, CP) VALUES (?, ?, ?);");
144         conn1.setAutoCommit(false);
145
146         int i = 0;
147         sInsert.setString(i++, "54320198V");
148         sInsert.setString(i++, "CARVAJAL");
149         sInsert.setString(i++, "10109");
150         sInsert.executeUpdate();
151
152         sInsert.setString(i + 1, "76543210S");
153         sInsert.setString(i + 1, "MARQUEZ");
154         sInsert.setString(i + 1, "46987");
155         sInsert.executeUpdate();
156
157         sInsert.setString(i + 1, "90123456A");
158         sInsert.setString(i + 1, "VOLINA");
159         sInsert.setString(i + 1, "35153");
160         sInsert.executeUpdate();
161
162         conn1.commit();
163     } catch (SQLException ex) {
164         System.out.println("Error al conectar a la base de datos");
165         try {
166             conn1.rollback();
167         } catch (Exception er) {
168             System.err.println("ERROR haciendo ROLBACK");
169             er.printStackTrace(System.err);
170         }
171     } catch (Exception e) {
172         System.err.println("ERROR de conexión");
173         e.printStackTrace(System.err);
174     }
    
```

Conectado a ejercicio 1  
DNI: 78901234X  
APELIDOS: NADALES  
CP: 44126  
  
DNI: 89012345E  
APELIDOS: HOJAS  
CP: null  
  
DNI: 56789012B  
APELIDOS: SAMPER  
CP: 29730

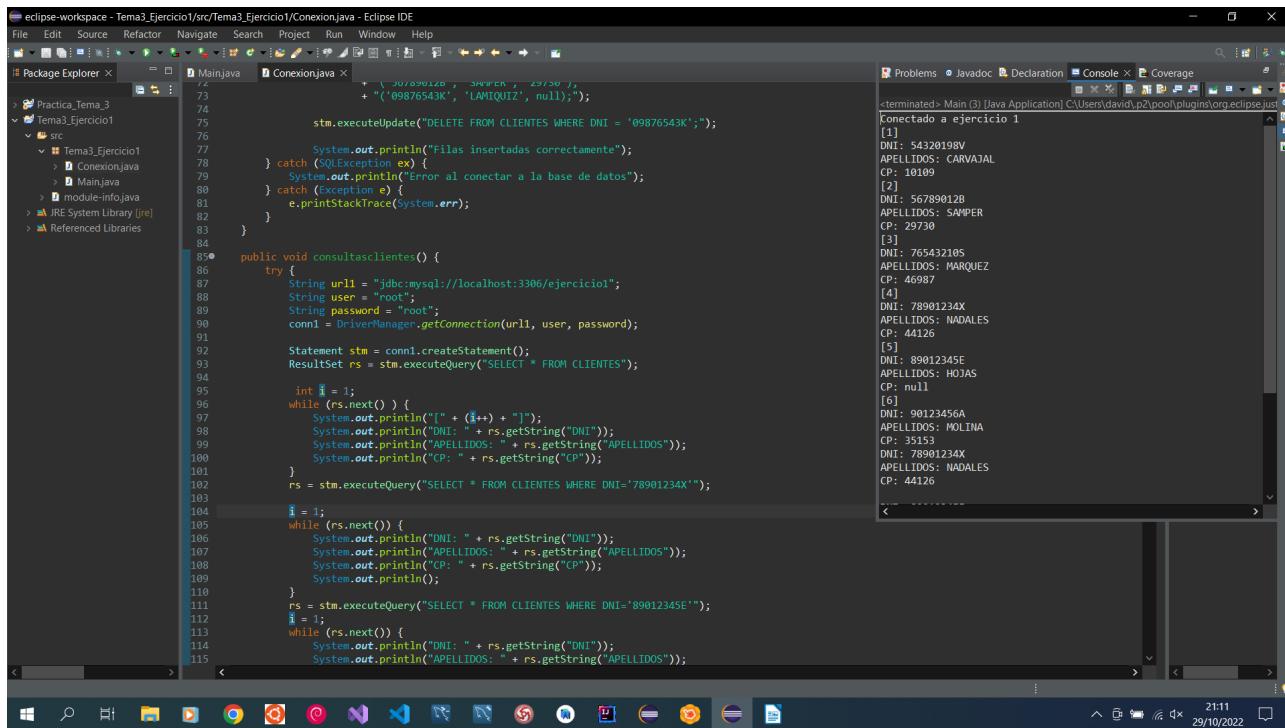
Error al conectar a la base de datos  
Error al conectar a la base de datos

The screenshot shows the Eclipse IDE interface with the 'Conexión.java' file open in the central editor. The code implements a class named 'Conexion' with methods for creating a connection, inserting data into a 'CLIENTES' table, and performing various database operations like creating tables and closing connections. The code uses JDBC to interact with a MySQL database.

```
1 package Tema3_Ejercicio1;
2
3 import java.sql.Connection;
4 import java.sql.DriverManager;
5 import java.sql.PreparedStatement;
6 import java.sql.SQLException;
7 import java.sql.Statement;
8
9 public class Conexion {
10     Connection conn1;
11
12     public void crearTabla() {
13         try {
14             Statement sCreate = conn1.createStatement();
15             sCreate.executeUpdate("CREATE TABLE CLIENTES(DNI, APELLIDOS, CP) VALUES ('', '' , '')");
16         } catch (SQLException ex) {
17             System.out.println("Error al conectar a la base de datos");
18         }
19     }
20
21     public void insertarDatos() {
22         try {
23             Statement sInsert = conn1.createStatement();
24             sInsert.executeUpdate("INSERT INTO CLIENTES(DNI, APELLIDOS, CP) VALUES ('54320198V', 'CARVAJAL', '10109')");
25         } catch (SQLException ex) {
26             System.out.println("Error de conexión");
27         }
28     }
29
30     public void consultarClientes() {
31         try {
32             Statement sSelect = conn1.createStatement();
33             String resultado = sSelect.executeQuery("SELECT * FROM CLIENTES").getResultSet().toString();
34             System.out.println(resultado);
35         } catch (SQLException ex) {
36             System.out.println("Error de conexión");
37         }
38     }
39
40     public void cerrarConexion() {
41         try {
42             conn1.close();
43         } catch (SQLException ex) {
44             System.out.println("Error de conexión");
45         }
46     }
47 }
```

The screenshot shows the Eclipse IDE interface with the 'Main.java' file open in the central editor. The code defines a 'Main' class with a static 'main' method. This method creates an instance of the 'Conexion' class, performs the same sequence of database operations as the previous code (creating a table, inserting data, querying it, and closing the connection), and then exits the application.

```
1 package Tema3_Ejercicio1;
2
3 public class Main {
4
5     public static void main(String[] args) {
6         Conexion constructor = new Conexion();
7
8         constructor.crearTabla();
9         //constructor.creartabla();
10        //constructor.insertardatos();
11        constructor.consultasclientes();
12        constructor.sintransacciones();
13        constructor.transacciones();
14        //constructor.cerrarconexion();
15    }
16 }
```



The screenshot shows the Eclipse IDE interface. On the left, the Package Explorer view displays a project structure with a file named 'ConexionJava.java'. The code in this file is as follows:

```

eclipse-workspace - Tema3_Ejercicio1/src/Tema3_Ejercicio1/ConexionJava - Eclipse IDE
File Edit Source Refactor Navigate Search Project Run Window Help
Package Explorer X MainJava ConexionJava X
Practica_Tema_3
Tema3_Ejercicio1
src
Tema3_Ejercicio1
MainJava
module-info.java
IREE System Library [ire]
Referenced Libraries
73     + " + rs.getString("APPELLIDOS") + ", " + rs.getString("CP") + ");"
74     + "('09876543K', 'LAMIQUEZ', null);");
75     statement.executeUpdate("DELETE FROM CLIENTES WHERE DNI = '09876543K');");
76
77     System.out.println("filas insertadas correctamente");
78 } catch (SQLException ex) {
79     System.out.println("Error al conectar a la base de datos");
80 } catch (Exception e) {
81     e.printStackTrace(System.err);
82 }
83 }

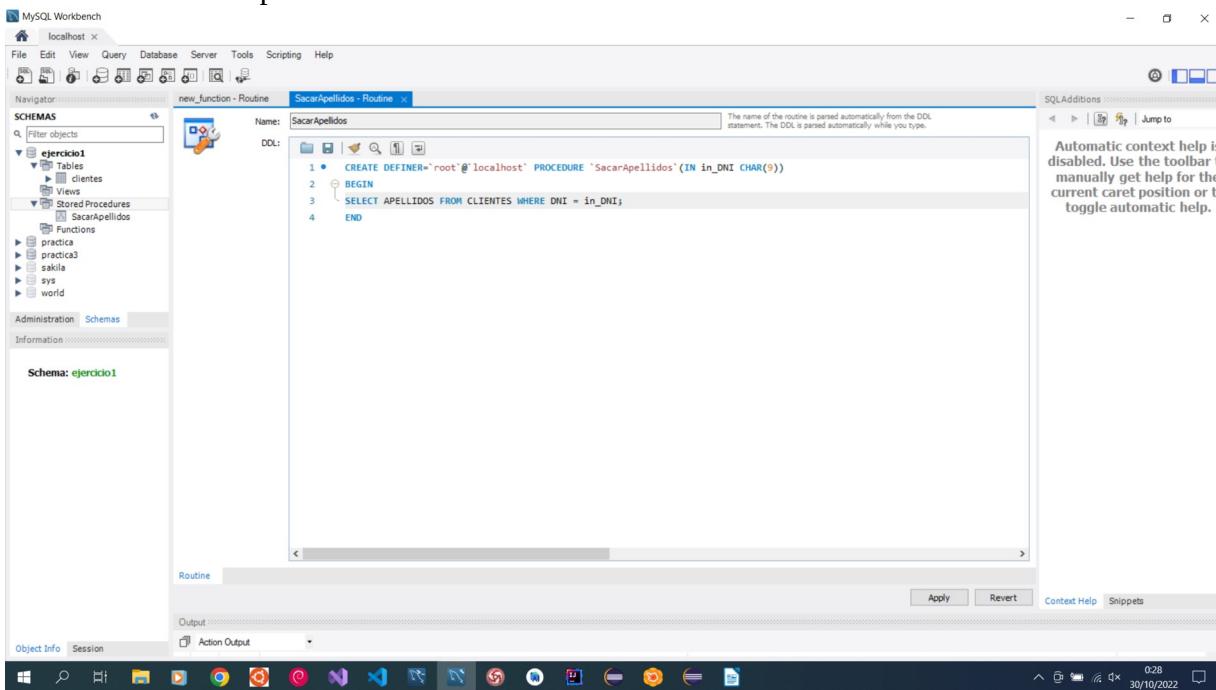
85 public void consultasclientes() {
86     try {
87         String url = "jdbc:mysql://localhost:3306/ejercicio1";
88         String user = "root";
89         String password = "root";
90         conn1 = DriverManager.getConnection(url, user, password);
91
92         Statement stm = conn1.createStatement();
93         ResultSet rs = stm.executeQuery("SELECT * FROM CLIENTES");
94
95         int i = 1;
96         while (rs.next()) {
97             System.out.println("[" + (i++) + "]");
98             System.out.println("DNI: " + rs.getString("DNI"));
99             System.out.println("APELLIDOS: " + rs.getString("APELLIDOS"));
100            System.out.println("CP: " + rs.getString("CP"));
101        }
102        rs = stm.executeQuery("SELECT * FROM CLIENTES WHERE DNI='78901234X'");
103
104        i = 1;
105        while (rs.next()) {
106            System.out.println("[" + rs.getString("DNI"));
107            System.out.println("APELLIDOS: " + rs.getString("APELLIDOS"));
108            System.out.println("CP: " + rs.getString("CP"));
109            System.out.println("]");
110        }
111        rs = stm.executeQuery("SELECT * FROM CLIENTES WHERE DNI='89012345E'");
112        i = 1;
113        while (rs.next()) {
114            System.out.println("[" + rs.getString("DNI"));
115            System.out.println("APELLIDOS: " + rs.getString("APELLIDOS"));
116        }
    
```

The right side of the interface shows the 'Console' tab with the output of the Java application, which lists several client records with their DNI, Apellidos, and CP values.

### 3.6 – PROCEDIMIENTOS Y FUNCIONES ALMACENADAS

a) Crea una función almacenada con MySQL a la que se le pase el DNI de un cliente y que devuelva sus apellidos. Crea un programa en Java que realice una llamada a ella utilizando JDBC y escriba el valor devuelto por la función. Será necesario consultar documentación o investigar cómo crear funciones almacenadas en la base de datos utilizada, pero se hará de manera muy similar a un procedimiento almacenado.

- Creando un procedimiento:



The screenshot shows the MySQL Workbench interface. In the 'Navigator' pane, under the 'ejercicio1' schema, there is a 'Stored Procedures' folder containing a procedure named 'SacarApellidos'. The 'Routine' tab shows the SQL code for the procedure:

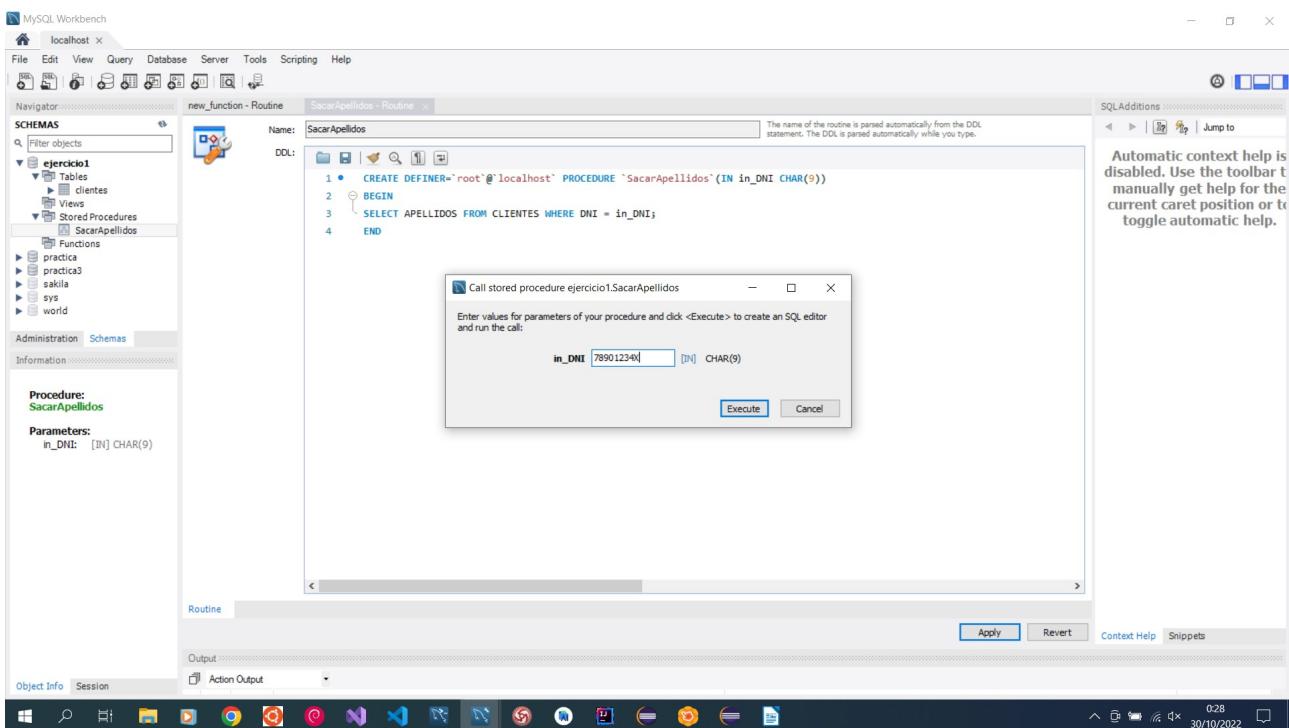
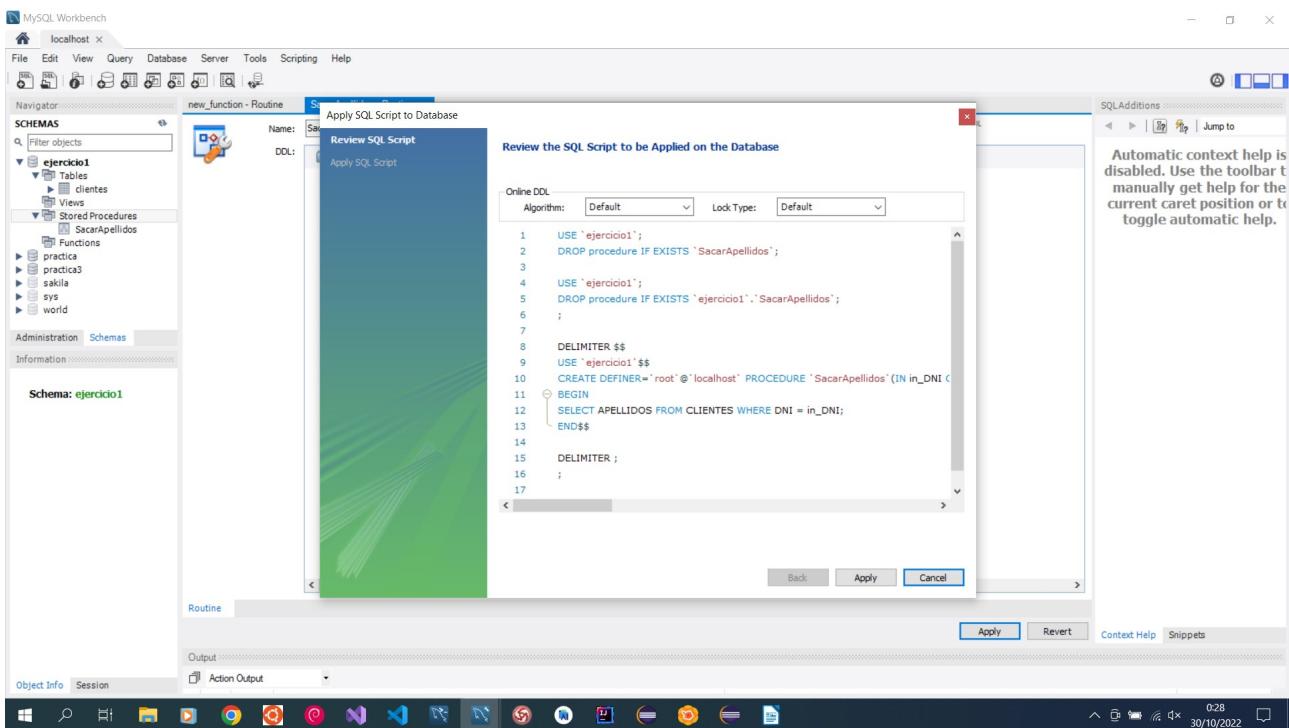
```

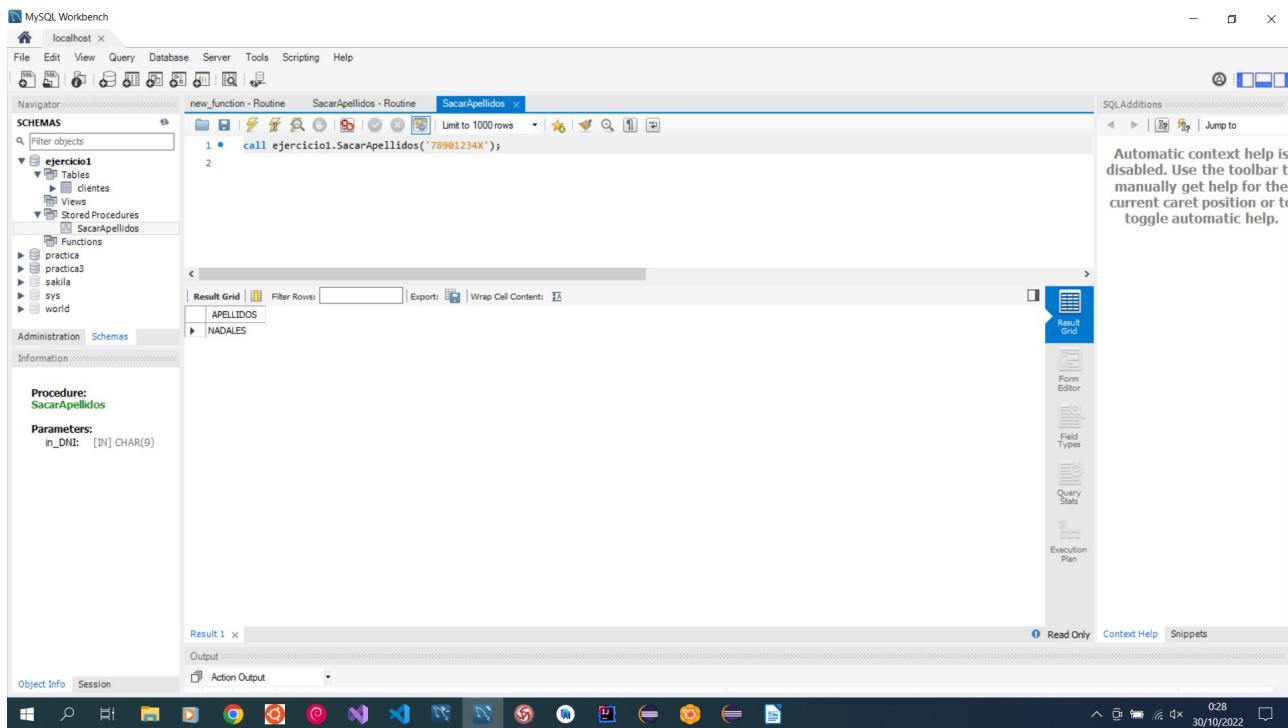
CREATE DEFINER='root'@'localhost' PROCEDURE `SacarApellidos`(IN in_DNI CHAR(9))
BEGIN
    SELECT APPELLIDOS FROM CLIENTES WHERE DNI = in_DNI;
END
    
```

The 'Output' pane at the bottom shows the results of a test query:

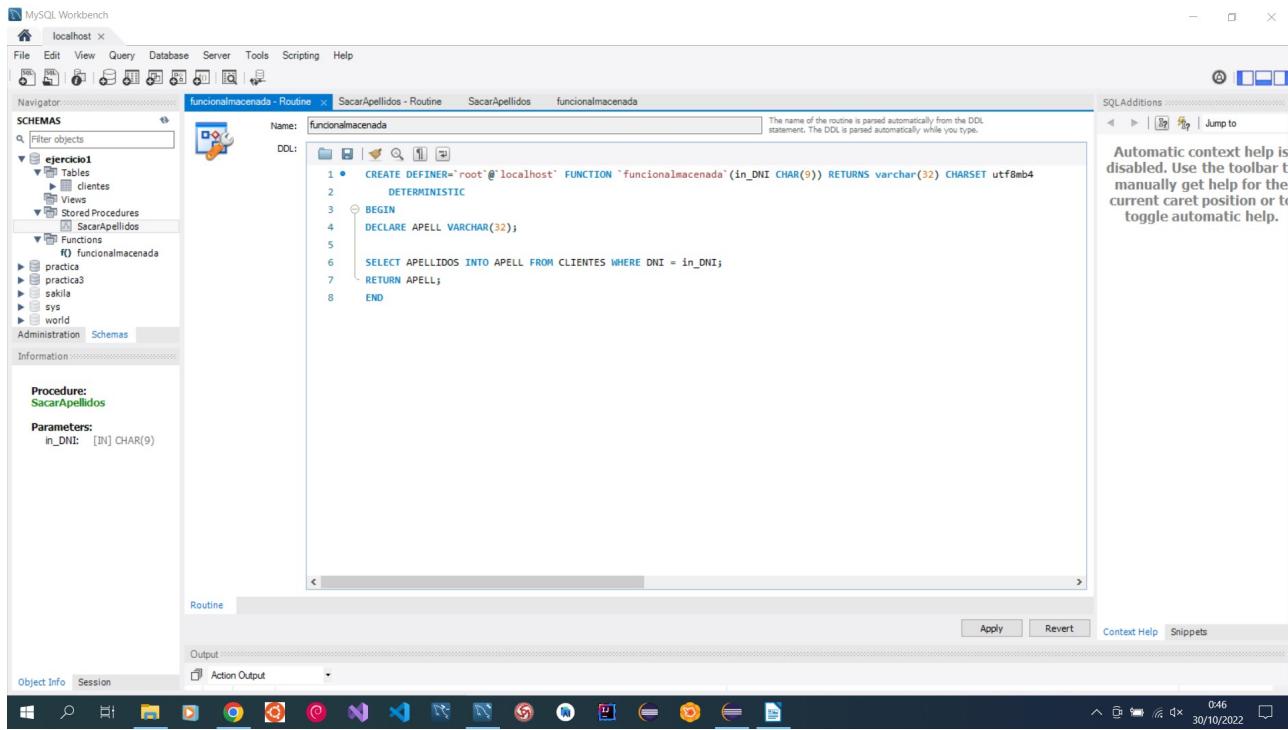
```

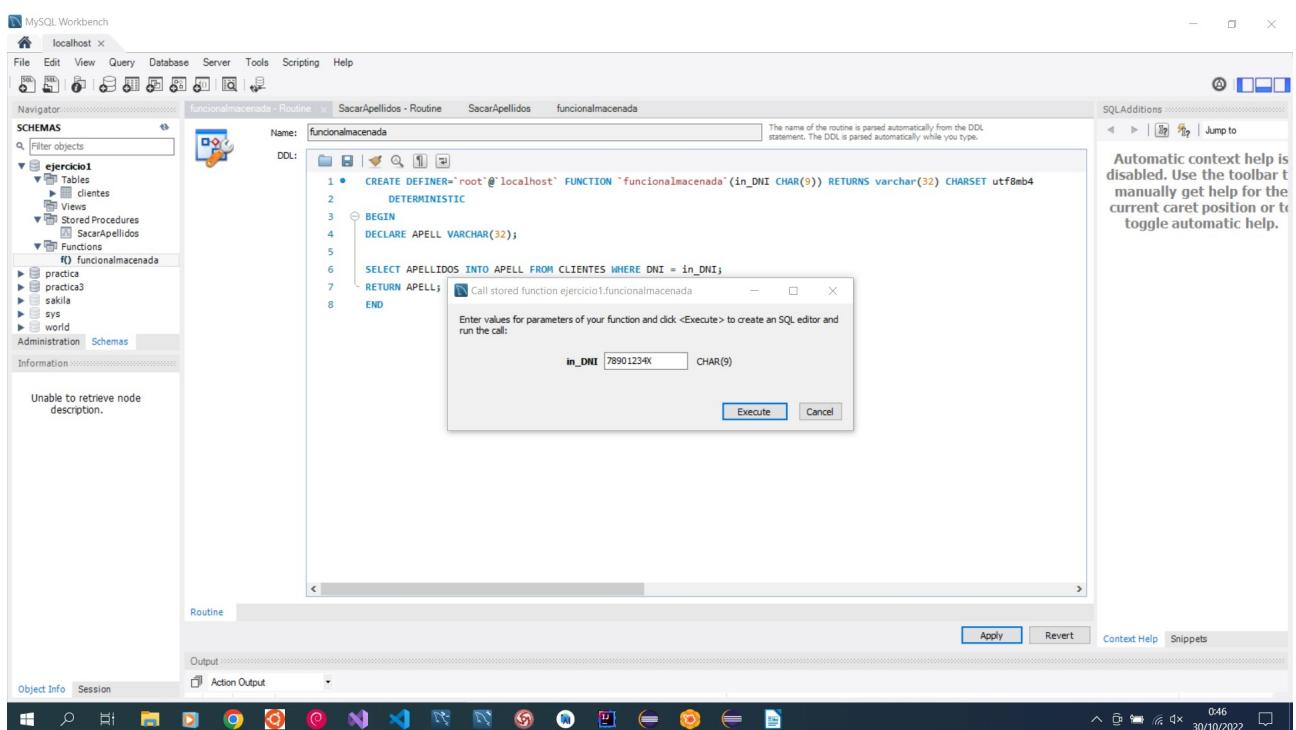
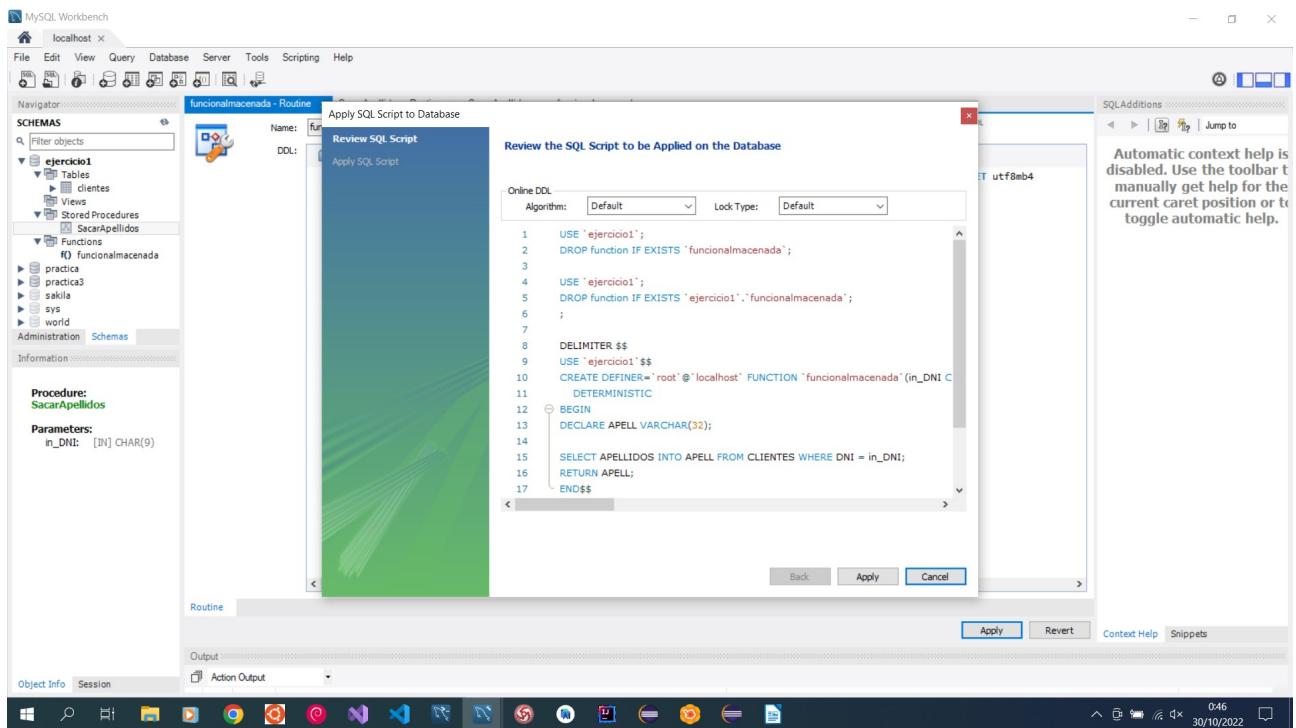
+-----+
| APPELLIDOS |
+-----+
| LAMIQUEZ   |
+-----+
    
```

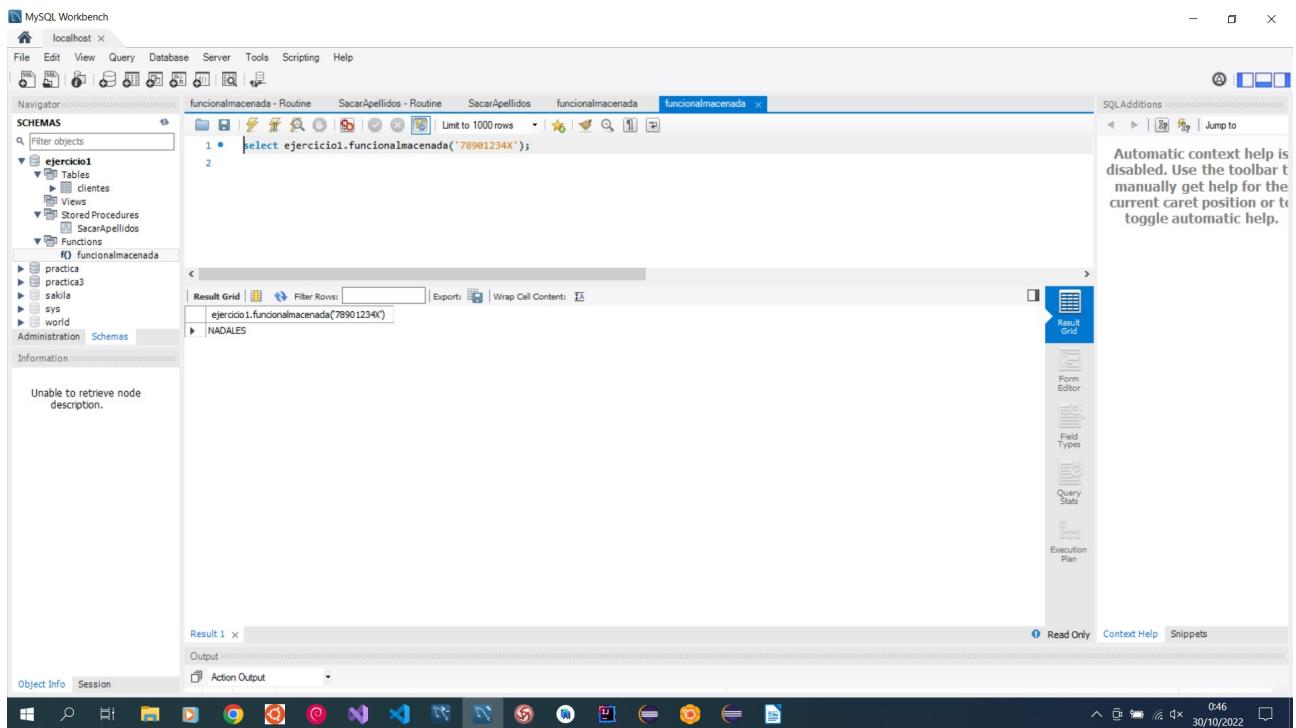




- Creando una función:







- Llamada a la función creada:

```

eclipse-workspace - Tema3_Ejercicio1/src/Tema3_Ejercicio1/Conexion.java - Eclipse IDE
File Edit Source Refactor Navigate Search Project Run Window Help
File Edit View Insert Database Tools Help
Mainjava D Conexion.java
194     sInsert.setString(i++, "78901234X");
195     sInsert.executeUpdate();
196
197     sInsert.setString(i = 1, "765432105");
198     sInsert.setString(i++, "MARQUEZ");
199     sInsert.setString(i++, "46987");
200     sInsert.executeUpdate();
201
202     sInsert.setString(i = 1, "90123456A");
203     sInsert.setString(i++, "MOLINA");
204     sInsert.setString(i++, "35153");
205     sInsert.executeUpdate();
206
207 } catch (SQLException ex) {
208     System.out.println("Error al conectar a la base de datos");
209 } catch (Exception e) {
210     System.err.println("ERROR de conexión");
211     e.printStackTrace(System.err);
212 }
213
214 public void funcionsacarapellidos() {
215     try {
216         String url = "jdbc:mysql://localhost:3306/ejercicio1";
217         String user = "root";
218         String password = "root";
219         conn1 = DriverManager.getConnection(url, user, password);
220
221         CallableStatement calstm = conn1.prepareCall("{? = call funcionalmacenada(?)}");
222
223         calstm.registerOutParameter(1, Types.VARCHAR);
224         calstm.setString(2, "78901234X");
225         calstm.execute();
226         String resultado = calstm.getString(1);
227         System.out.println(resultado);
228
229     } catch (SQLException ex) {
230         System.out.println("Error al conectar a la base de datos");
231     } catch (Exception e) {
232         System.err.println("ERROR de conexión");
233         e.printStackTrace(System.err);
234     }
235 }
236

```

The screenshot shows the Eclipse IDE interface with the following details:

- Project Explorer:** Shows the project structure with files like Main.java and Conexion.java.
- Code Editor:** Displays Java code for connecting to a MySQL database and executing a stored procedure named `funcionalmacenada`.
- Console:** Shows the output of the executed code, indicating a successful connection and the name "NADALES".
- Bottom:** Shows the Windows taskbar with various application icons.

```

eclipse-workspace - Tema3_Ejercicio1/src/Tema3_Ejercicio1/Conexion.java - Eclipse IDE
File Edit Source Refactor Navigate Search Project Run Window Help
Package Explorer X Main.java Conexion.java X
1>> 194 sInsert.setString(+i, "NADALES");
195 sInsert.executeUpdate();
196
197 sInsert.setString(i = 1, "765432105");
198 sInsert.setString(+i, "MARQUEZ");
199 sInsert.setString(+i, "46987");
200 sInsert.executeUpdate();
201
202 sInsert.setString(i = 1, "90123456A");
203 sInsert.setString(+i, "MOLINA");
204 sInsert.setString(+i, "35153");
205 sInsert.executeUpdate();
206
207 } catch (SQLException ex) {
208     System.out.println("Error al conectar a la base de datos");
209 } catch (Exception e) {
210     System.err.println("ERROR de conexión");
211     e.printStackTrace(System.err);
212 }
213
214 public void funcionesacarapellidos() {
215     try {
216         String url1 = "jdbc:mysql://localhost:3306/ejercicio1";
217         String user = "root";
218         String password = "root";
219         conn1 = DriverManager.getConnection(url1, user, password);
220
221         CallableStatement calstm = conn1.prepareCall("{? = call funcionalmacenada(?)}");
222
223         calstm.registerOutParameter(1, Types.VARCHAR);
224         calstm.setString(2, "78901234X");
225         calstm.execute();
226         String resultado = calstm.getString(1);
227         System.out.println(resultado);
228
229     } catch (SQLException ex) {
230         System.out.println("Error al conectar a la base de datos");
231     } catch (Exception e) {
232         System.err.println("ERROR de conexión");
233         e.printStackTrace(System.err);
234     }
235 }
236

```

- Llamada al procedure creado:

The screenshot shows the Eclipse IDE interface with the following details:

- Project Explorer:** Shows the project structure with files like Main.java and Conexion.java.
- Code Editor:** Displays Java code for connecting to a MySQL database and executing a stored procedure named `proceduresacarapellidos()`.
- Outline View:** Shows the class structure with methods like `conn1`, `conexion()`, `cerrarconexion()`, `creatable()`, `insertardatos()`, `consultasclientes()`, `transacciones()`, `sintransacciones()`, and `proceduresacarapellidos()`.
- Bottom:** Shows the Windows taskbar with various application icons.

```

eclipse-workspace - Tema3_Ejercicio1/src/Tema3_Ejercicio1/Conexion.java - Eclipse IDE
File Edit Source Refactor Navigate Search Project Run Window Help
Package Explorer X Main.java Conexion.java X Outline X
223 calstm.registerOutParameter(1, Types.VARCHAR);
224 calstm.setString(2, "78901234X");
225 calstm.execute();
226 String resultado = calstm.getString(1);
227 System.out.println(resultado);
228
229 } catch (SQLException ex) {
230     System.out.println("Error al conectar a la base de datos");
231 } catch (Exception e) {
232     System.err.println("ERROR de conexión");
233     e.printStackTrace(System.err);
234 }
235
236 public void proceduresacarapellidos() {
237     try {
238         String url1 = "jdbc:mysql://localhost:3306/ejercicio1";
239         String user = "root";
240         String password = "root";
241         conn1 = DriverManager.getConnection(url1, user, password);
242
243         CallableStatement calstm = conn1.prepareCall("{call sacarApellidos(?)}");
244
245         calstm.setString(1, "78901234X");
246         calstm.execute();
247
248         ResultSet rs = calstm.getResultSet();
249
250         int ncli = 0;
251         while (rs.next()) {
252             System.out.println("[" + (++ncli) + "]");
253             System.out.println("APELIDOS: " + rs.getString("APELIDOS"));
254         }
255
256     } catch (SQLException ex) {
257         System.out.println("Error al conectar a la base de datos");
258     } catch (Exception e) {
259         System.err.println("ERROR de conexión");
260         e.printStackTrace(System.err);
261     }
262 }
263
264

```

The screenshot shows the Eclipse IDE interface with the following details:

- Project Explorer:** Shows the project structure with files like Main.java, Conexion.java, and module-info.java.
- Code Editor:** Displays the Java code for `Conexion.java`. The code connects to a MySQL database and executes a stored procedure to retrieve last names.
- Console:** Shows the output of the executed code, indicating a successful connection and the retrieved result: "APELIDOS: NADALES".
- Bottom Bar:** Shows the Windows taskbar with various application icons and the system clock.

```
223     calstm.registerOutParameter(1, Types.VARCHAR);
224     calstm.setString(2, "78901234X");
225     calstm.execute();
226     String resultado = calstm.getString(1);
227     System.out.println(resultado);
228
229 } catch (SQLException ex) {
230     System.out.println("Error al conectar a la base de datos");
231 } catch (Exception e) {
232     System.err.println("ERROR de conexión");
233     e.printStackTrace(System.err);
234 }
235
236 public void proceduresacarApellidos() {
237     try {
238         String url1 = "jdbc:mysql://localhost:3306/ejercicio1";
239         String user = "root";
240         String password = "Root";
241         conn1 = DriverManager.getConnection(url1, user, password);
242
243         CallableStatement calstm = conn1.prepareCall("{call sacarApellidos(?)}");
244
245         calstm.setString(1, "78901234X");
246         calstm.execute();
247
248         ResultSet rs = calstm.getResultSet();
249
250         int nCli = 0;
251         while (rs.next()) {
252             System.out.println("[" + (++nCli) + "]");
253             System.out.println("APELIDOS: " + rs.getString("APELIDOS"));
254         }
255
256     } catch (SQLException ex) {
257         System.out.println("Error al conectar a la base de datos");
258     } catch (Exception e) {
259         System.err.println("ERROR de conexión");
260         e.printStackTrace(System.err);
261     }
262 }
263
264 }
```