Universidad de Chile
Facultad de Ciencias Físicas y Matemáticas
Departamento de Ciencias de la Computación

Bases de Datos

Laboratorio 6

Acceso Programático

Integrantes: Jean Cherubini
Claudio Urbina
Profesores: Aidan Hogan
Auxiliares: Sebastián Ferrada A.
Ayudantes: Henry Rosales M.
Marco Caballero G.
Pablo Miranda Á.

Fecha de entrega: 7 de mayo de 2018
Santiago, Chile

P0

El código utilizado para buscar los salarios mensuales por apellido paterno se encuentra en el código de abajo. En este caso se busca el apellido “Mackinnon”.

```
SELECT nombres, apellido_p, total
FROM uchile.transparencia
WHERE apellido_p = 'Mackinnon'
ORDER BY total ASC;
```

El código utilizado para encontrar nuestras notas del ramo se presenta en el código de abajo:

```
SELECT nombre,nota
FROM nota.BdD2018
WHERE nombre = 'Cherubini Fouilloux, Jean Franco'
OR nombre= 'Urbina Lara, Claudio Isaias';
```

P1

No se pide adjuntar ninguna respuesta.

P2

No es seguro contra inyecciones pues fue posible colocar notas 7 para ambos utiizando:

```
';
UPDATE nota.BdD2018
SET nota=7
WHERE nombre='Urbina Lara, Claudio Isaias'
OR nombre='Cherubini Fouilloux, Jean Franco';
SELECT total
FROM uchile.transparencia
WHERE apellido_p='Mackinnon
```

Esto claramente arroja una excepción pues no está programado para entregar 2 solicitudes, pero esto no importa en demasía pues el ataque es exitoso.
Además de esto, el código presentado arribse ve extraño , pero esto es debido a la asimetría intencional causada por cerrar con una comilla la consulta default del programa **ApellidoApp.java**.

P3

Para poder consultar nuestras notas se utilizó el código:

```
' UNION SELECT p.nombre, p.nombre, p.nombre, p.nota, p.nota, p.nota
FROM nota.BdD2018 p
WHERE p.nombre = 'Urbina Lara, Claudio Isaias'
OR p.nombre='Cherubini Fouilloux, Jean Franco'
UNION SELECT nombre, nombre, nombre, nota, nota, nota
FROM nota.BdD2018
WHERE nombre = 'Claudio Urbina'
OR nombre='Jean Cherubini
```

P4

No es posible ejecutar los ataques anteriores, no surten ningun tipo de efecto, solo se entregan resultados vacíos.

P5

El archivo es entregado aparte, pero se adjunta aquí de todas formas.

```
1  package org.uchile;
2
3  import java.io.BufferedReader;
4  import java.io.IOException;
5  import java.io.InputStreamReader;
6  import java.sql.Connection;
7  import java.sql.DriverManager;
8  import java.sql.PreparedStatement;
9  import java.sql.ResultSet;
10 import java.sql.ResultSetMetaData;
11 import java.sql.SQLException;
12 import java.util.Properties;
13
14 public class NotaAppSegura {
15     private static final String HOST = "cc3201.dcc.uchile.cl";
16     private static final String PORT = "5432"; // o 5440
17     private static final String DATABASE = "cc3201";
18     private static final String CONNECTION_URL =
       "jdbc:postgresql://"+HOST+":"+PORT+"/"+DATABASE;
19     private static final String USERNAME = "cc3201";
20     private static final String PASSWORD = "je_<3_cc3201";
21     private static final String SSL = "true";
22
23    private static final String KILL = "-k";
24
25
26    public static void main(String[] args) throws SQLException, IOException{
27       String url = CONNECTION_URL;
28
29       Properties props = new Properties();
30       props.setProperty("user",USERNAME);
31       props.setProperty("password",PASSWORD);
```

```
32      props.setProperty("ssl",SSL);
33
34      // la siguiente propiedad es para deshabilitar la validación de
35      // certificados en SSL (normalmente, no se recomienda, pero en el lab,
36      // será complejo instalar un certificado en cada truststore)
37      props.setProperty("sslfactory","org.postgresql.ssl.NonValidatingFactory");
38      Connection conn = DriverManager.getConnection(url, props);
39
40      // esta vez vamos a preparar el "statement" antes
41      // luego reemplezaremos los '?' con los valores de entrada
42      PreparedStatement pSt = conn.prepareStatement("SELECT nombre,nota*100/7 as Nota,
    comentario FROM nota.BdD2018 WHERE ID=? ");
43      PreparedStatement pSt2 = conn.prepareStatement("update nota.BdD2018 set
    comentario=? where id=?");
44      BufferedReader br = new BufferedReader(new InputStreamReader(System.in,"utf-8"));
45
46      try{
47         while(true){
48            System.out.println("Ingrese un Id y pulse [Intro] (o '"+KILL+"' para
    matar):");
49
50            String input = br.readLine().trim();
51            if(input.equals(KILL)) break;
52            pSt.setInt(1, Integer.parseInt(input));
53
54            // executar una consulta
55            System.out.println("=== Ejecutando la consulta: "+pSt+" ===");
56            ResultSet rs = pSt.executeQuery();
57
58            System.out.println("=== Resultados ===");
59
60            System.out.println("\n-- Esquema --");
61            ResultSetMetaData rsm = rs.getMetaData();
62            for(int i=1; i<=rsm.getColumnCount(); i++){ // las columnas empiezan con 1
63               if(i!=1)
64                  System.out.print("\t");
65               System.out.print(rsm.getColumnLabel(i)+":"+rsm.getColumnTypeName(i));
66            }
67
68            System.out.println("\n\n-- Datos --");
69
70            while (rs.next()){ // mover el cursor a la proxima posición (y devolver true
    si hay una tupla más)
71               System.out.println();
72               for(int i=1; i<=rsm.getColumnCount(); i++){ // las columnas empiezan con 1
73                  if(i!=1)
74                     System.out.print("\t");
75                  System.out.print(rs.getString(i));
76               }
77               if(rs != null){
78                  System.out.println("\nDesea modificar el comentario? [y/n]:");
79                  String deseo = br.readLine().trim();
```

```
80                  if(deseo.equals("y")){
81                     System.out.println("\nIngrese un comentario si lo desea y pulse
    [Intro] (o -k para cancelar):");
82                     String input2 = br.readLine().trim();
83                     if(input2.equals(KILL)) break;
84                     pSt2.setString(1, input2);
85                     pSt2.setInt(2, Integer.parseInt(input));
86                     System.out.println("=== Ejecutando la consulta: "+pSt2+" ===");
87                     pSt2.execute();
88                  }else{
89                     if(deseo.equals("n")){
90                        System.out.println("\nOk.");
91                     }else{
92                        System.out.println("\nComando Inválido.");
93                     }
94                  }
95
96               }
97            }
98            System.out.println("\n==============================\n\n\n");
99            rs.close();
100        }
101     } catch(Exception e){
102        e.printStackTrace();
103     } finally {
104        try{
105           pSt.close();
106           br.close();
107        } finally {
108           conn.close();
109           System.out.println("////////////////////////// Terminado.");
110        }
111     }
112  }
113 }
```