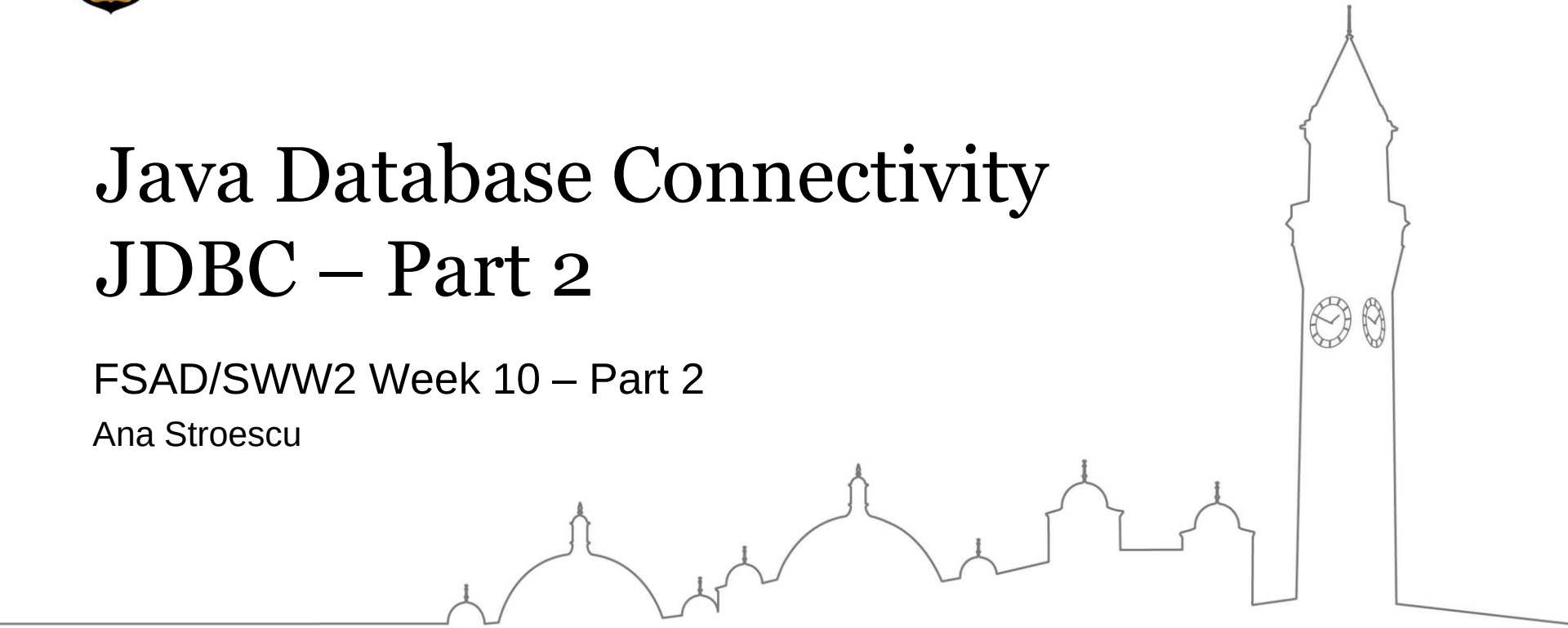
UNIVERSITY OF BIRMINGHAM

Java Database Connectivity JDBC – Part 2

FSAD/SWW2 Week 10 – Part 2

Ana Stroescu



Contents

- JDBC Examples:
  - Select
  - Insert
  - Update
  - Delete
- Working with Prepared Statements
  - Special Characters
  - SQL Injections

UNIVERSITY OF BIRMINGHAM

JDBC Examples

1. Example of SELECT Query `Select.java`

1: Import

2: Initialise driver

3: Open Connection

4: Query

5. Process results

Replace username and password with your own!

```
import java.sql.*;
public class Select {
public static void main(String[] args) {
      String USERNAME= "postgres";    String PASSWORD = "password";
      String URL= "jdbc:postgresql://localhost:5432/Music";
      try {
            Class.forName("org.postgresql.Driver");

            Connection con = DriverManager.getConnection(URL, USERNAME, PASSWORD);

            Statement stmt = con.createStatement();
            String sql = "SELECT * FROM album";
            ResultSet rs = stmt.executeQuery(sql);

            while (rs.next()) {
                  System.out.print("Artist ID: " + rs.getObject("artistid")
                  + " Title: " + rs.getObject("title") +"\n");
            }
            rs.close();    stmt.close();    con.close();
} catch (Exception e) { System.out.println(e); }} }
```

6&7: Clean-up environment

UNIVERSITY OF BIRMINGHAM

```
Artist ID: 0 Title: Diamond Dogs
Artist ID: 0 Title: Station to Station
Artist ID: 1 Title: The Beatles
Artist ID: 2 Title: The Idiot
Artist ID: 3 Title: For Your Pleasure
Artist ID: 1 Title: Revolver
Artist ID: 5 Title: Sabbath Bloody Sabbath
Artist ID: 6 Title: Welcome to Jamrock
Artist ID: 7 Title: The Top
Artist ID: 8 Title: Strange Days
Artist ID: 9 Title: Appetite for Destruction
Artist ID: 4 Title: Trans-Europe Express
Artist ID: 10 Title: Bleach
Artist ID: 0 Title: The Man Who Sold The World
Artist ID: 1 Title: Abbey Road
Artist ID: 13 Title: Hounds of Love
Artist ID: 14 Title: The Velvet Underground and Nico
Artist ID: 15 Title: The Stone Roses
Artist ID: 16 Title: Kid A
Artist ID: 17 Title: Exile On Main St
Artist ID: 18 Title: It Takes A Nation Of Millions To Hold Us Back
Artist ID: 19 Title: Led Zeppelin
Artist ID: 20 Title: The Southern Harmony and Musical Companion
Artist ID: 21 Title: Imagine

Process finished with exit code 0
```

UNIVERSITY OF BIRMINGHAM

Console output for `Select.java`

2. Example of INSERT Query

Insert.java

1: Import

```
import java.sql.*;
class Insert {
      public static void main(String args[]) {
      String USERNAME= "postgres"; String PASSWORD = "password";
      String URL= "jdbc:postgresql://localhost:5432/Music";
            try {
```

2: Initialise driver

```
Class.forName("org.postgresql.Driver");
```

3: Open Connection

```
Connection con = DriverManager.getConnection(URL, USERNAME, PASSWORD);
```

4&5: Execute command

```
Statement stmt = con.createStatement();
String sql = "INSERT INTO label VALUES ('Warner', 'New York', 'USA')";
int count = stmt.executeUpdate(sql);

System.out.print("Inserted " + count + " records successfully");
```

6&7: Clean-up environment

```
stmt.close();
con.close();
} catch (Exception e) { System.out.println(e); }}}
```

UNIVERSITY OF BIRMINGHAM

```
Music=# select * from label;
          name           |    region     | country
-------------------------+---------------+---------
 RCA                     | New York      | USA
 Apple                   | London        | UK
 Island                  | London        | UK
 Parlophone              | London        | UK
 Vertigo                 | London        | UK
 Universal               | Santa Monica  | USA
 Fiction                 | London        | UK
 Elektra                 | New York      | USA
 Geffen                  | Santa Monica  | USA
 Klink Klang             | Dusseldorf    | Germany
 Sub Pop                 | Seattle       | USA
 EMI                     | London        | UK
 Sony                    | New York      | USA
 Verve                   | Santa Monica  | USA
 Silvertone              | London        | UK
 Silvertone              | Illinois      | USA
 Rolling Stones Records  | London        | UK
 Def Jam                 | New York      | USA
 Atlantic                | Los Angeles   | USA
 Def American            | Los Angeles   | USA
 Warner                  | New York      | USA
(21 rows)
```

UNIVERSITY OF BIRMINGHAM

Table label after running **`Insert.java`**

3. Example of UPDATE Query

Update.java

1: Import

```
import java.sql.*;
class Update {
      public static void main(String args[]) {
      String USERNAME= "postgres"; String PASSWORD = "password";
      String URL= "jdbc:postgresql://localhost:5432/Music";
            try {
```

2: Initialise driver

```
            Class.forName("org.postgresql.Driver");
```

3: Open Connection

```
            Connection con = DriverManager.getConnection(URL, USERNAME, PASSWORD);
```

4&5: Execute command

```
            Statement stmt = con.createStatement();
            String sql = "UPDATE label SET name = 'BMG' WHERE name = 'Warner';
            int count = stmt.executeUpdate(sql);

            System.out.print("Updated " + count + " records successfully");
```

6&7: Clean-up environment

```
            stmt.close();
            con.close();
} catch (Exception e) { System.out.println(e); }}}
```

UNIVERSITY OF BIRMINGHAM

```
Music=# select * from label;
          name           |    region     | country
-------------------------+---------------+---------
 RCA                     | New York      | USA
 Apple                   | London        | UK
 Island                  | London        | UK
 Parlophone              | London        | UK
 Vertigo                 | London        | UK
 Universal               | Santa Monica  | USA
 Fiction                 | London        | UK
 Elektra                 | New York      | USA
 Geffen                  | Santa Monica  | USA
 Klink Klang             | Dusseldorf    | Germany
 Sub Pop                 | Seattle       | USA
 EMI                     | London        | UK
 Sony                    | New York      | USA
 Verve                   | Santa Monica  | USA
 Silvertone              | London        | UK
 Silvertone              | Illinois      | USA
 Rolling Stones Records  | London        | UK
 Def Jam                 | New York      | USA
 Atlantic                | Los Angeles   | USA
 Def American            | Los Angeles   | USA
 BMG                     | New York      | USA
(21 rows)
```

Table label after running **`Update.java`**

UNIVERSITY OF BIRMINGHAM

4. Example of DELETE Query

Delete.java

1: Import

2: Initialise driver

3: Open Connection

4&5: Execute command

6&7: Clean-up environment

```
import java.sql.*;
class Delete {
      public static void main(String args[]) {
      String USERNAME= "postgres"; String PASSWORD = "password";
      String URL= "jdbc:postgresql://localhost:5432/Music";
            try {
                  Class.forName("org.postgresql.Driver");

                  Connection con = DriverManager.getConnection(URL, USERNAME, PASSWORD);

                  Statement stmt = con.createStatement();
                  String sql = "DELETE FROM label WHERE name = 'BMG';
                  int count = stmt.executeUpdate(sql);

                  System.out.print("Deleted " + count + " records successfully");

                  stmt.close();
                  con.close();
} catch (Exception e) { System.out.println(e); }}}
```

```
Music=# select * from label;
          name           |    region     | country
-------------------------+---------------+---------
 RCA                     | New York      | USA
 Apple                   | London        | UK
 Island                  | London        | UK
 Parlophone              | London        | UK
 Vertigo                 | London        | UK
 Universal               | Santa Monica  | USA
 Fiction                 | London        | UK
 Elektra                 | New York      | USA
 Geffen                  | Santa Monica  | USA
 Klink Klang             | Dusseldorf    | Germany
 Sub Pop                 | Seattle       | USA
 EMI                     | London        | UK
 Sony                    | New York      | USA
 Verve                   | Santa Monica  | USA
 Silvertone              | London        | UK
 Silvertone              | Illinois      | USA
 Rolling Stones Records  | London        | UK
 Def Jam                 | New York      | USA
 Atlantic                | Los Angeles   | USA
 Def American            | Los Angeles   | USA
(20 rows)
```

No record with name = “BMG”

UNIVERSITY OF BIRMINGHAM

Table label after running **`Delete.java`**

Working with Prepared Statements

The advantages of `PreparedStatement` objects over `Statement` objects are:

1. The mechanism for inserting parameter values takes care of all necessary special character quoting in the correct manner.
2. They are more efficient for repetitive queries that are very similar except for some parameter values, because SQL is compiled once and executed many times, with the parameter values substituted in each execution.
3. Prevents SQL injections.

UNIVERSITY OF BIRMINGHAM

Working with Prepared Statements

- Scenario 1: Special characters

```
String sql = "INSERT INTO customer VALUES ('16', 'Marc', 'O''Reilly',
'54', 'NN168S', '3854' )";
int count = stmt.executeUpdate(sql);
```

Notice that the syntax of the SQL string passed as an argument must match the syntax of the database being used. The name *O'Reilly* is entered as O''Reilly, because the ' must be escaped.

UNIVERSITY OF BIRMINGHAM

Example of PreparedStatement

Preparedstmt.java

Parameters are specified as '?'

Parameter positions start at 1

Parameters are set using `setInt`, `setString`, etc.

The statement can be executed using `execute`, `executeUpdate` or `executeQuery`

Parameters can be cleared

Constructed with SQL statement

```
String sql = "INSERT INTO customer VALUES (?,?,?,?,?,?)";
PreparedStatement pstmt = con.prepareStatement(sql);
pstmt.clearParameters();
pstmt.setInt(1, 16);
pstmt.setString(2, "Marc");
pstmt.setString (3, "O'Reilly");
pstmt.setString (4, "54");
pstmt.setString (5, "NN168S");
pstmt.setInt (6, 3854);
int count = pstmt.executeUpdate();
System.out.print("Inserted" + count + " records successfully \n");

pstmt.clearParameters();
pstmt.setInt(1, 17);
pstmt.setString(2, "Olivia");
pstmt.setString (3, "O'Neill");
pstmt.setString (4, "456A");
pstmt.setString (5, "B59HW");
pstmt.setInt (6, 7653);
count = pstmt.executeUpdate();
System.out.print("Inserted" + count + " records successfully");

pstmt.close();
```

```
Music=# select * from customer;
 custid |   fname   |   lname   | housenum | postcode | creditcard
--------+-----------+-----------+----------+----------+------------
      0 | Terry     | David     | 14       | N164PT   |       1234
      1 | Terry     | David     | 34b      | NW16HJ   |       2343
      2 | Constance | Wilson    | 3        | MK24UJ   |       4354
      3 | Helen     | Babbage   | 4        | B705GH   |       6577
      4 | Bill      | Drummond  | 40       | OX15FH   |       8877
      5 | Ada       | Partridge | 77       | LS705DF  |       4545
      6 | William   | Stringer  | 2202c    | E172RW   |       1231
      7 | Rodney    | Silvers   | 2        | AL18RT   |       2846
      8 | Jimmy     | Osmond    | 1874     | M45FT    |       7485
      9 | Horatio   | Walters   | 14       | WS19BB   |       2332
     10 | Paul      | Cook      | 1        | NW15DP   |       8786
     11 | Steve     | Jones     | 1        | NW15DP   |       9987
     12 | Glen      | Matlock   | 1        | NW15DP   |       2865
     13 | John      | Lydon     | 1        | NW15DP   |       2227
     14 | Marc      | Feld      | 40       | N165WE   |       7803
     15 | Reginald  | Dwight    | 70       | OX25PL   |       5591
     16 | Marc      | O'Reilly  | 54       | NN168SH  |       3854
     17 | Olivia    | O'Neill   | 456A     | B59HW    |       7653
(18 rows)
```

Table customer after running **`Preparedstmt.java`**

UNIVERSITY OF BIRMINGHAM

Working with Prepared Statements

- Scenario 2: SQL Injections

```
String country = "UK'); DELETE FROM sale WHERE ('1' = '1";
String sql = "INSERT INTO artist VALUES ('23', 'Queen', '" + country + "')";
stmt.executeUpdate(sql);
```

An attacker can submit a valid SQL statement that changes the logic of the initial query used by the application. This is a software vulnerability known as SQL injection.

UNIVERSITY OF BIRMINGHAM

- The attacker can view/modify/delete sensitive data of other users or even get unauthorized access to the entire system.

In the previous example, the user successfully inserted a new entry in the artist table, but also deleted all the entries in the sale table.

```
Music=# select * from sale;
 salesref | custid | albumid | saledate
----------+--------+---------+----------
(0 rows)
```

Tables sale & artist after running **`Injection.java`**

```
Music=# select * from artist;
 artistid |          name          | countryoforigin
----------+------------------------+-----------------
        0 | David Bowie            | UK
        1 | The Beatles            | UK
        2 | Iggy Pop               | USA
        3 | Roxy Music             | UK
        4 | Kraftwerk              | Germany
        5 | Black Sabbath          | UK
        6 | Damian Marley          | Jamaica
        7 | The Cure               | UK
        8 | The Doors              | USA
        9 | Guns N' Roses          | USA
       10 | Nirvana                | USA
       11 | The Faces              | UK
       12 | Jimi Hendrix           | USA
       13 | Kate Bush              | UK
       14 | The Velvet Underground | USA
       15 | The Stone Roses        | UK
       16 | Radiohead              | UK
       17 | The Rolling Stones     | UK
       18 | Public Enemy           | USA
       19 | Led Zeppelin           | UK
       20 | The Black Crowes       | USA
       21 | John Lennon            | UK
       22 | Bon Jovi               | US
       23 | Queen                  | UK
(24 rows)
```

We can use `PreparedStatement` to prevent an SQL injection:

```
String sql = "INSERT INTO artist VALUES ('23', 'Queen’, ?)";
PreparedStatement pstmt = con.prepareStatement(sql);
pstmt.clearParameters();
pstmt.setString(1, "UK");

int count = pstmt.executeUpdate();
System.out.print("Inserted" + count + " records successfully \n");

pstmt.close();
```

UNIVERSITY OF BIRMINGHAM

References

- Oracle Tutorial: Lesson: JDBC Basics
- PostgreSQL JDBC Driver Documentation
- PostgreSQL JDBC Tutorial
- Package java.sql