

Lecture 12_Stored Procedures Part 4

DRIVER

The screenshot shows the Eclipse IDE with a project named 'jdbc-lecture-12-stored-procedures-part-4'. The 'Driver.java' file is open in the editor. The code defines a 'Driver' class with a 'main' method that connects to a database, prepares a statement, sets parameters, and displays the result set. The console output shows the results of the query, including a list of employees and their salaries, and a message about reusing the prepared statement.

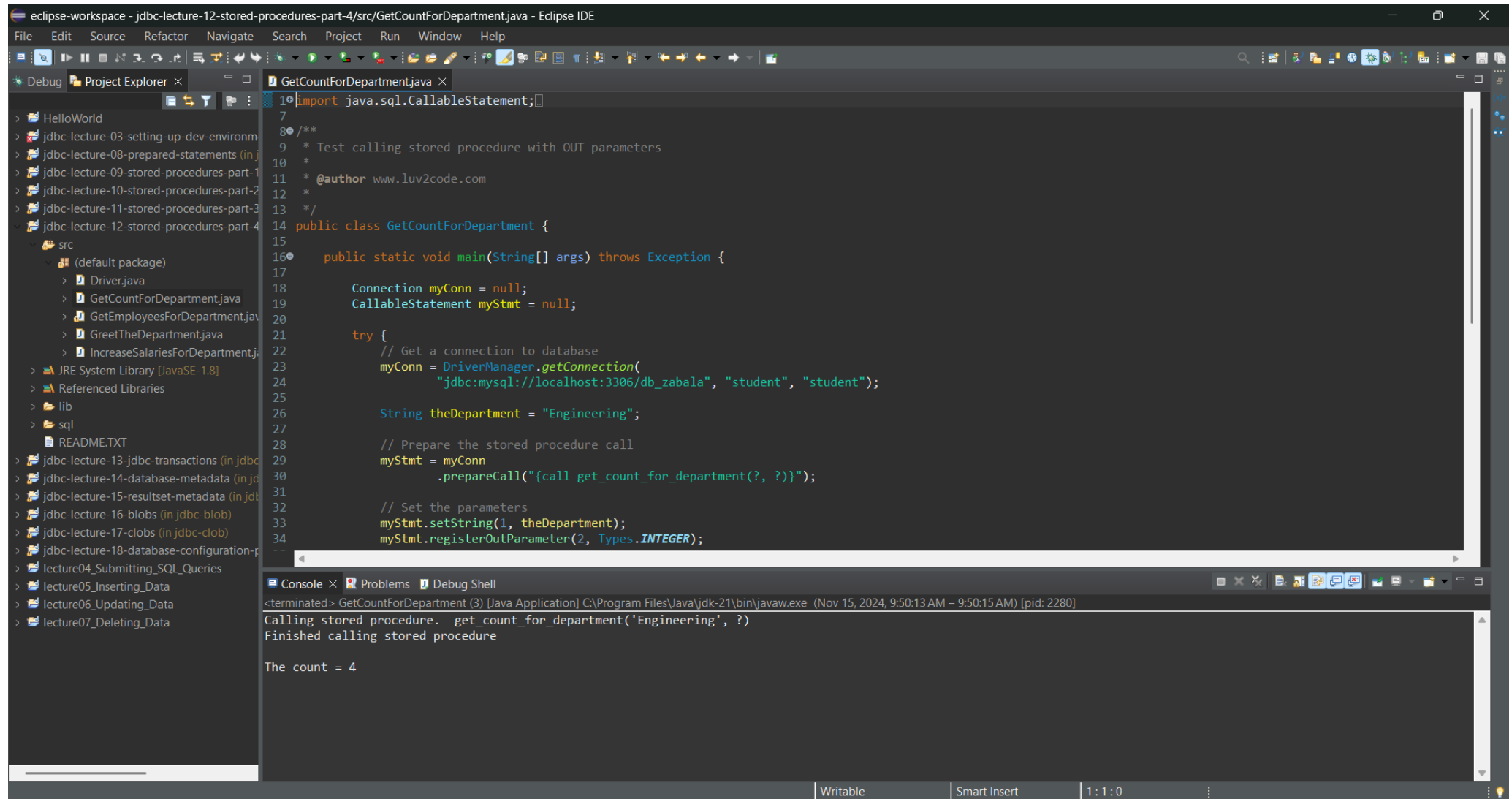
```
1
2
3 import java.sql.*;
4
5 public class Driver {
6
7     public static void main(String[] args) throws SQLException {
8
9         Connection myConn = null;
10        PreparedStatement myStmt = null;
11        ResultSet myRs = null;
12
13        try {
14            // 1. Get a connection to database
15            myConn = DriverManager.getConnection("jdbc:mysql://localhost:3306/db_zabala", "student", "student");
16
17            // 2. Prepare statement
18            myStmt = myConn.prepareStatement("select * from employees where salary > ? and department=?");
19
20            // 3. Set the parameters
21            myStmt.setDouble(1, 80000);
22            myStmt.setString(2, "Legal");
23
24            // 4. Execute SQL query
25            myRs = myStmt.executeQuery();
26
27            // 5. Display the result set
28            display(myRs);
29        }
30    }
31}
```

Console Output:

```
<terminated> Driver (3) [Java Application] C:\Program Files\Java\jdk-21\bin\javaw.exe (Nov 12, 2024, 7:27:24 PM – 7:27:25 PM) [pid: 7672]
Queue, Susan, 130000.00, Legal
Smith, Paul, 100000.00, Legal
Waters, David, 90000.00, Legal

Reuse the prepared statement: salary > 25000, department = HR
Doe, John, 55000.00, HR
Williams, David, 120000.00, HR
Adams, Carl, 50000.00, HR
Davis, John, 45000.00, HR
Wright, Eric, 33000.00, HR
```

Get Count for Department



The screenshot displays the Eclipse IDE interface. The main editor shows the file `GetCountForDepartment.java` with the following code:

```
10 import java.sql.CallableStatement;
7
8 /**
9  * Test calling stored procedure with OUT parameters
10  *
11  * @author www.luv2code.com
12  *
13  */
14 public class GetCountForDepartment {
15
16     public static void main(String[] args) throws Exception {
17
18         Connection myConn = null;
19         CallableStatement myStmt = null;
20
21         try {
22             // Get a connection to database
23             myConn = DriverManager.getConnection(
24                 "jdbc:mysql://localhost:3306/db_zabala", "student", "student");
25
26             String theDepartment = "Engineering";
27
28             // Prepare the stored procedure call
29             myStmt = myConn
30                 .prepareCall("{call get_count_for_department(?, ?)}");
31
32             // Set the parameters
33             myStmt.setString(1, theDepartment);
34             myStmt.registerOutParameter(2, Types.INTEGER);
35         } catch (SQLException e) {
36             e.printStackTrace();
37         } finally {
38             if (myConn != null) myConn.close();
39             if (myStmt != null) myStmt.close();
40         }
41     }
42 }
```

The left sidebar shows the Project Explorer with the following structure:

- src
 - (default package)
 - Driver.java
 - GetCountForDepartment.java
 - GetEmployeesForDepartment.java
 - GreetTheDepartment.java
 - IncreaseSalariesForDepartment.java
- JRE System Library [JavaSE-1.8]
- Referenced Libraries
 - lib
 - sql
- README.TXT

The bottom console window shows the following output:

```
<terminated> GetCountForDepartment (3) [Java Application] C:\Program Files\Java\jdk-21\bin\javaw.exe (Nov 15, 2024, 9:50:13 AM – 9:50:15 AM) [pid: 2280]
Calling stored procedure. get_count_for_department('Engineering', ?)
Finished calling stored procedure

The count = 4
```

Get Employees for Department

The screenshot shows the Eclipse IDE with a Java project named 'jdbc-lecture-12-stored-procedures-part-4'. The main editor displays the file 'GetEmployeesForDepartment.java'. The code is as follows:

```
1 import java.sql.*;
2
3 public class GetEmployeesForDepartment {
4
5     public static void main(String[] args) throws Exception {
6
7         Connection myConn = null;
8         CallableStatement myStmt = null;
9         ResultSet myRs = null;
10
11         try {
12             // Get a connection to database
13             myConn = DriverManager.getConnection(
14                 "jdbc:mysql://localhost:3306/db_zabala", "student", "student");
15
16             String theDepartment = "Engineering";
17
18             // Prepare the stored procedure call
19             myStmt = myConn
20                 .prepareCall("{call get_employees_for_department(?)}");
21
22             // Set the parameter
23             myStmt.setString(1, theDepartment);
24
25             // Call stored procedure
26             System.out.println("Calling stored procedure.  get_employees_for_department('" + theDepartment + "')");
27             myStmt.execute();
28             System.out.println("Finished calling stored procedure.\n");
29         }
30     }
31 }
```

The Project Explorer on the left shows the project structure, including a 'src' folder with files like 'Driver.java', 'GetCountForDepartment.java', 'GetEmployeesForDepartment.java', 'GreetTheDepartment.java', and 'IncreaseSalariesForDepartment.java'. It also shows the 'JRE System Library [JavaSE-1.8]' and 'Referenced Libraries'.

The Console at the bottom shows the output of the program:

```
<terminated> GetEmployeesForDepartment (3) [Java Application] C:\Program Files\Java\jdk-21\bin\javaw.exe (Nov 12, 2024, 7:29:34 PM - 7:29:35 PM) [pid: 18060]
Calling stored procedure.  get_employees_for_department('Engineering')
Finished calling stored procedure.
Public, Mary, Engineering, 105000.00
Johnson, Lisa, Engineering, 80000.00
Brown, Bill, Engineering, 80000.00
Fowler, Mary, Engineering, 95000.00
```

The status bar at the bottom indicates 'Writable', 'Smart Insert', and the time '16:50:405'.

Greet the Department

The screenshot shows the Eclipse IDE interface. The main editor displays the `GreetTheDepartment.java` file. The code is as follows:

```
1 import java.sql.*;
2
3 /**
4  * Test calling stored procedure with INOUT parameters
5  *
6  * @author www.luv2code.com
7  */
8
9 public class GreetTheDepartment {
10
11     public static void main(String[] args) throws Exception {
12
13         Connection myConn = null;
14         CallableStatement myStmt = null;
15
16         try {
17             // Get a connection to database
18             myConn = DriverManager.getConnection(
19                 "jdbc:mysql://localhost:3306/db_zabala", "student", "student");
20
21             String theDepartment = "Engineering";
22
23             // Prepare the stored procedure call
24             myStmt = myConn
25                 .prepareCall("{call greet_the_department(?)}");
26
27             // Set the parameters
28             myStmt.registerOutParameter(1, Types.VARCHAR);
29             myStmt.setString(1, theDepartment);
30         } catch (SQLException e) {
31             e.printStackTrace();
32         } finally {
33             if (myConn != null) myConn.close();
34             if (myStmt != null) myStmt.close();
35         }
36     }
37 }
```

The left sidebar shows the Project Explorer with the following structure:

- src
 - (default package)
 - Driver.java
 - GetCountForDepartment.java
 - GetEmployeesForDepartment.java
 - GreetTheDepartment.java
 - IncreaseSalariesForDepartment.java
- JRE System Library [JavaSE-1.8]
- Referenced Libraries
- lib
- sql
 - table-setup.sql
- README.TXT

The bottom console shows the execution output:

```
<terminated> GreetTheDepartment (3) [Java Application] C:\Program Files\Java\jdk-21\bin\javaw.exe (Nov 12, 2024, 7:30:30 PM - 7:30:31 PM) [pid: 12060]
Calling stored procedure.  greet_the_department('Engineering')
Finished calling stored procedure
The result = Hello to the awesome Engineering team!
```

The status bar at the bottom indicates the current state: Writable, Smart Insert, 19:59:404.

Increase Salaries for Department

The screenshot shows the Eclipse IDE with a Java project named 'jdbc-lecture-12-stored-procedures-part-4'. The main editor displays the file 'IncreaseSalariesForDepartment.java'. The code imports 'java.sql.*' and defines a 'public class IncreaseSalariesForDepartment' with a 'main' method. The 'main' method establishes a database connection, sets the department to 'Engineering', and calls a stored procedure to increase salaries by 10,000. The console output shows the salaries before the update and the successful execution of the stored procedure call.

```
1 import java.sql.*;
2
3 /**
4  * Test calling stored procedure with IN parameters
5  *
6  * @author www.luv2code.com
7  */
8
9 public class IncreaseSalariesForDepartment {
10
11     public static void main(String[] args) throws Exception {
12
13         Connection myConn = null;
14         CallableStatement myStmt = null;
15
16         try {
17             // Get a connection to database
18             myConn = DriverManager.getConnection(
19                 "jdbc:mysql://localhost:3306/db_zabala", "student", "student");
20
21             String theDepartment = "Engineering";
22             int theIncreaseAmount = 10000;
23
24             // Show salaries BEFORE
25             System.out.println("Salaries BEFORE\n");
26             showSalaries(myConn, theDepartment);
27
28             // Prepare the stored procedure call
29             myStmt = myConn
30         }
31     }
32 }
```

Salaries BEFORE

Public	Mary	Engineering	105000.00
Johnson	Lisa	Engineering	80000.00
Brown	Bill	Engineering	80000.00
Fowler	Mary	Engineering	95000.00

Calling stored procedure. increase_salaries_for_department('Engineering', 10000)
Finished calling stored procedure