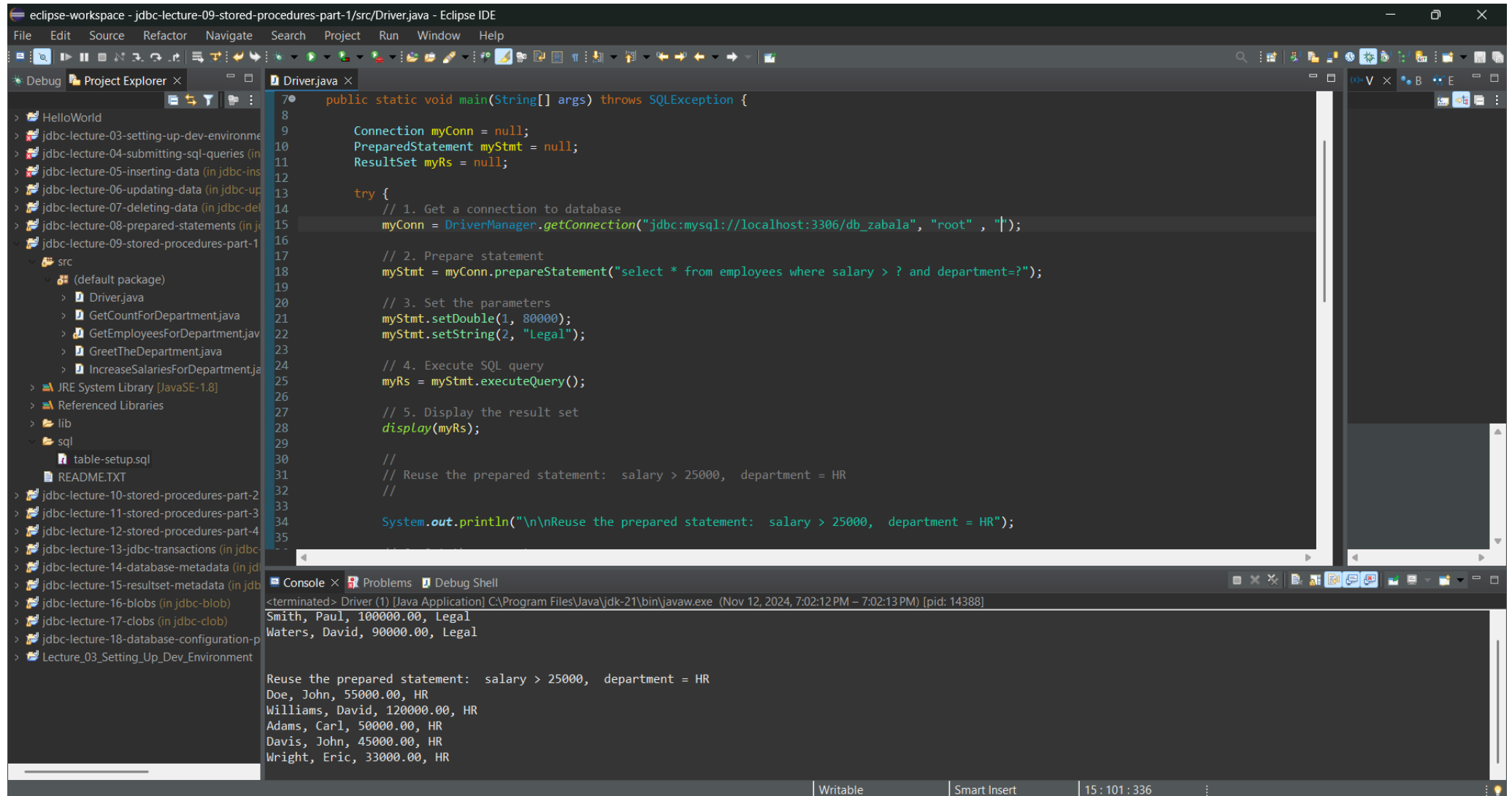


# Lecture 9\_Stored Procedures Part 1

## DRIVER



```
eclipse-workspace - jdbc-lecture-09-stored-procedures-part-1/src/Driver.java - Eclipse IDE
File Edit Source Refactor Navigate Search Project Run Window Help

Debug Project Explorer Driver.java

> HelloWorld
> jdbc-lecture-03-setting-up-dev-environment
> jdbc-lecture-04-submitting-sql-queries (in jdbc-lecture-04-submitting-sql-queries)
> jdbc-lecture-05-inserting-data (in jdbc-lecture-05-inserting-data)
> jdbc-lecture-06-updating-data (in jdbc-lecture-06-updating-data)
> jdbc-lecture-07-deleting-data (in jdbc-lecture-07-deleting-data)
> jdbc-lecture-08-prepared-statements (in jdbc-lecture-08-prepared-statements)
> jdbc-lecture-09-stored-procedures-part-1
  > 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
  > jdbc-lecture-10-stored-procedures-part-2
  > jdbc-lecture-11-stored-procedures-part-3
  > jdbc-lecture-12-stored-procedures-part-4
  > jdbc-lecture-13-jdbc-transactions (in jdbc-lecture-13-jdbc-transactions)
  > jdbc-lecture-14-database-metadata (in jdbc-lecture-14-database-metadata)
  > jdbc-lecture-15-resultset-metadata (in jdbc-lecture-15-resultset-metadata)
  > jdbc-lecture-16-blobs (in jdbc-lecture-16-blobs)
  > jdbc-lecture-17-clobs (in jdbc-lecture-17-clobs)
  > jdbc-lecture-18-database-configuration-properties (in jdbc-lecture-18-database-configuration-properties)
  > Lecture_03_Setting_Up_Dev_Environment

70 public static void main(String[] args) throws SQLException {
71
72     Connection myConn = null;
73     PreparedStatement myStmt = null;
74     ResultSet myRs = null;
75
76     try {
77         // 1. Get a connection to database
78         myConn = DriverManager.getConnection("jdbc:mysql://localhost:3306/db_zabala", "root", "|");
79
80         // 2. Prepare statement
81         myStmt = myConn.prepareStatement("select * from employees where salary > ? and department=?");
82
83         // 3. Set the parameters
84         myStmt.setDouble(1, 80000);
85         myStmt.setString(2, "Legal");
86
87         // 4. Execute SQL query
88         myRs = myStmt.executeQuery();
89
90         // 5. Display the result set
91         display(myRs);
92
93         //
94         // Reuse the prepared statement: salary > 25000, department = HR
95         //
96
97         System.out.println("\n\nReuse the prepared statement: salary > 25000, department = HR");
98     }
99 }

Console Problems Debug Shell
<terminated> Driver (1) [Java Application] C:\Program Files\Java\jdk-21\bin\javaw.exe (Nov 12, 2024, 7:02:12 PM - 7:02:13 PM) [pid: 14388]
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 shows the Eclipse IDE with a project named 'jdbc-lecture-09-stored-procedures-part-1'. The main editor displays the file 'GetCountForDepartment.java'. The code is as follows:

```
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
36              // Call stored procedure
37              System.out.println("Calling stored procedure.  get_count_for_department('" + theDepartment + "', ?)");
38              myStmt.execute();
39              System.out.println("Finished calling stored procedure");
40          }
41      }
42  }
```

The Console window at the bottom shows the output of the application:

```
<terminated> GetCountForDepartment [Java Application] C:\Program Files\Java\jdk-21\bin\javaw.exe (Nov 12, 2024, 7:05:57 PM – 7:05:58 PM) [pid: 18376]
Calling stored procedure.  get_count_for_department('Engineering', ?)
Finished calling stored procedure

The count = 4
```

The status bar at the bottom indicates 'Writable', 'Smart Insert', and the time '24 : 81 : 582'.

# Get Employees for Department

The screenshot shows the Eclipse IDE with the following components:

- Project Explorer:** Displays a project named 'jdbc-lecture-09-stored-procedures-part-1' with a source folder 'src' containing several Java files, including 'GetEmployeesForDepartment.java'.
- Editor:** Shows the code for 'GetEmployeesForDepartment.java'. The code imports 'java.sql.\*', defines a 'main' method, and uses JDBC to connect to a database, call a stored procedure, and print the results.
- Console:** Shows the output of the application, including the message 'Calling stored procedure. get\_employees\_for\_department('Engineering')' and the resulting list of employees.

```
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 }
```

Console Output:

```
<terminated> GetEmployeesForDepartment [Java Application] C:\Program Files\Java\jdk-21\bin\javaw.exe (Nov 12, 2024, 7:06:28 PM - 7:06:29 PM) [pid: 6804]
Calling stored procedure. get_employees_for_department('Engineering')
Finished calling stored procedure.

Public, Mary, Engineering, 75000.00
Johnson, Lisa, Engineering, 50000.00
Brown, Bill, Engineering, 50000.00
Fowler, Mary, Engineering, 65000.00
```

# Greet the Department

The screenshot shows the Eclipse IDE with a project named 'jdbc-lecture-09-stored-procedures-part-1'. The main editor displays the file 'GreetTheDepartment.java'. 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
31             // Execute the stored procedure call
32             myStmt.execute();
33
34             // Print the result
35             String result = myStmt.getString(1);
36             System.out.println("The result = " + result);
37
38             // Close the connection
39             myConn.close();
40
41         } catch (SQLException e) {
42             e.printStackTrace();
43         }
44     }
45 }
```

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 output of the application:

```
<terminated> GreetTheDepartment [Java Application] C:\Program Files\Java\jdk-21\bin\javaw.exe (Nov 12, 2024, 7:07:03 PM - 7:07:04 PM) [pid: 9888]
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 'Writable', 'Smart Insert', and '19 : 59 : 404'.

# Increase Salaries for Department

The screenshot shows the Eclipse IDE with a Java project named 'jdbc-lecture-09-stored-procedures-part-1'. The main editor displays the file 'IncreaseSalariesForDepartment.java'. The code imports 'java.sql.\*' and includes a Javadoc comment describing the test. The 'main' method uses 'DriverManager.getConnection' to connect to a MySQL database at 'localhost:3306/db\_zabala' with 'student' credentials. It sets 'theDepartment' to 'Engineering' and 'theIncreaseAmount' to 10000. It then calls 'showSalaries' and 'prepare' methods. The console output shows the execution of the stored procedure and the resulting salary list.

```
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         } catch (SQLException e) {
32             e.printStackTrace();
33         } finally {
34             if (myConn != null) {
35                 myConn.close();
36             }
37             if (myStmt != null) {
38                 myStmt.close();
39             }
40         }
41     }
42 }
```

Console Output:

```
<terminated> IncreaseSalariesForDepartment [Java Application] C:\Program Files\Java\jdk-21\bin\javaw.exe (Nov 12, 2024, 7:07:37 PM - 7:07:38 PM) [pid: 11304]
Calling stored procedure.  increase_salaries_for_department('Engineering', 10000)
Finished calling stored procedure

Salaries AFTER

Public, Mary, Engineering, 85000.00
Johnson, Lisa, Engineering, 60000.00
Brown, Bill, Engineering, 60000.00
Fowler, Mary, Engineering, 75000.00
```