

Lesson 01 Demo 08

Extracting Database Information Using Metadata

Objective: To extract information about the database in a Java program using metadata for understanding the database structure and schema. This approach allows for dynamic retrieval of database details such as tables, columns, and data types.

Tool required: Eclipse IDE

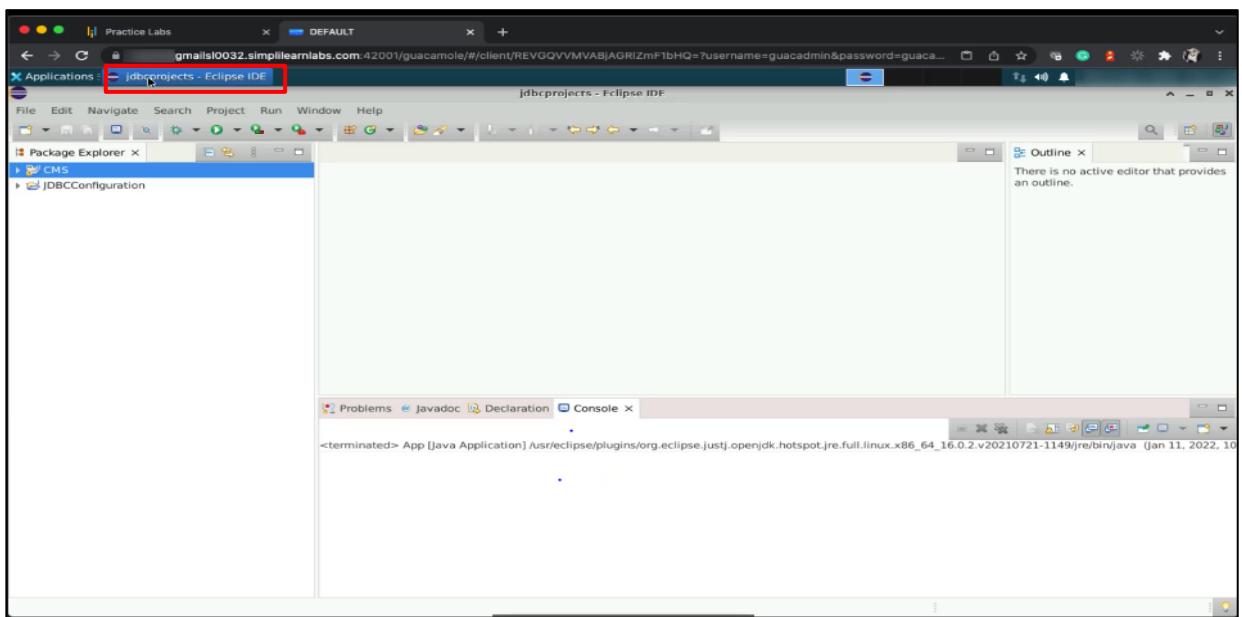
Prerequisites: None

Steps to be followed:

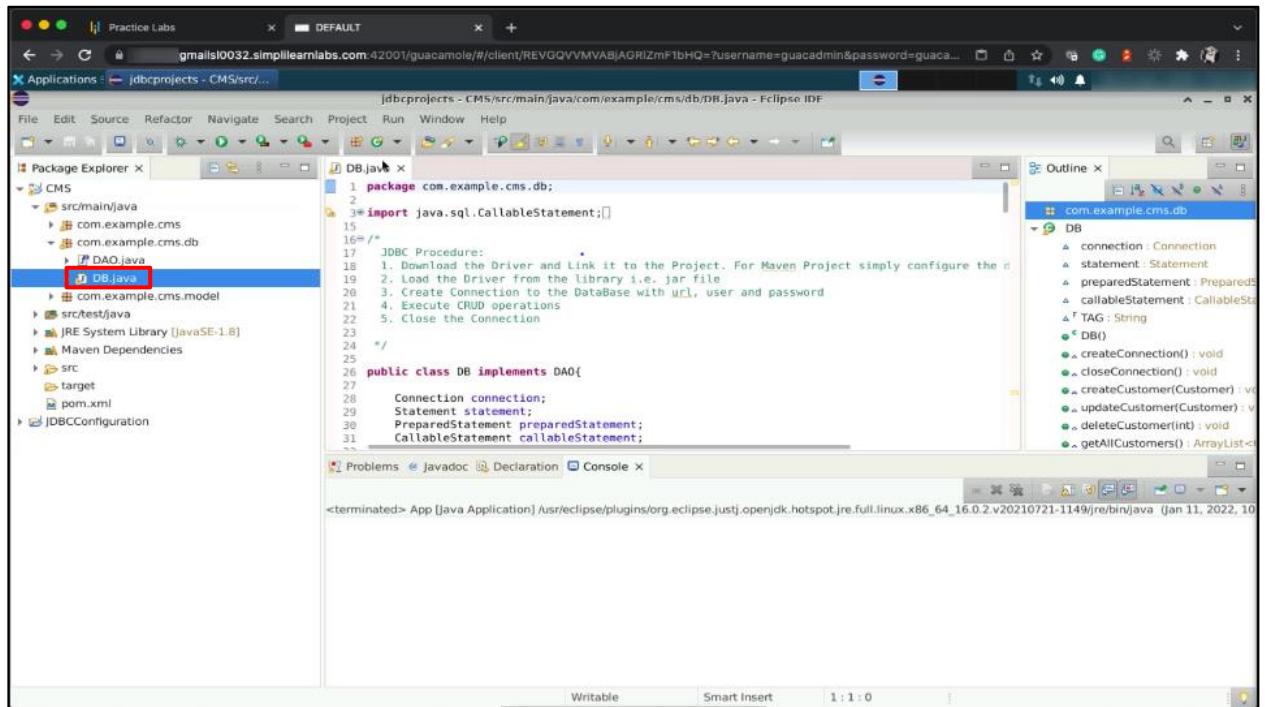
1. Extract details about the database and list of tables
2. Extract Metadata containing the list of tables

Step 1: Extract details about the database and list of tables

1.1 Open Eclipse IDE



1.2 Open CMS > src/main/java > com.example.cms.db > DB.java file

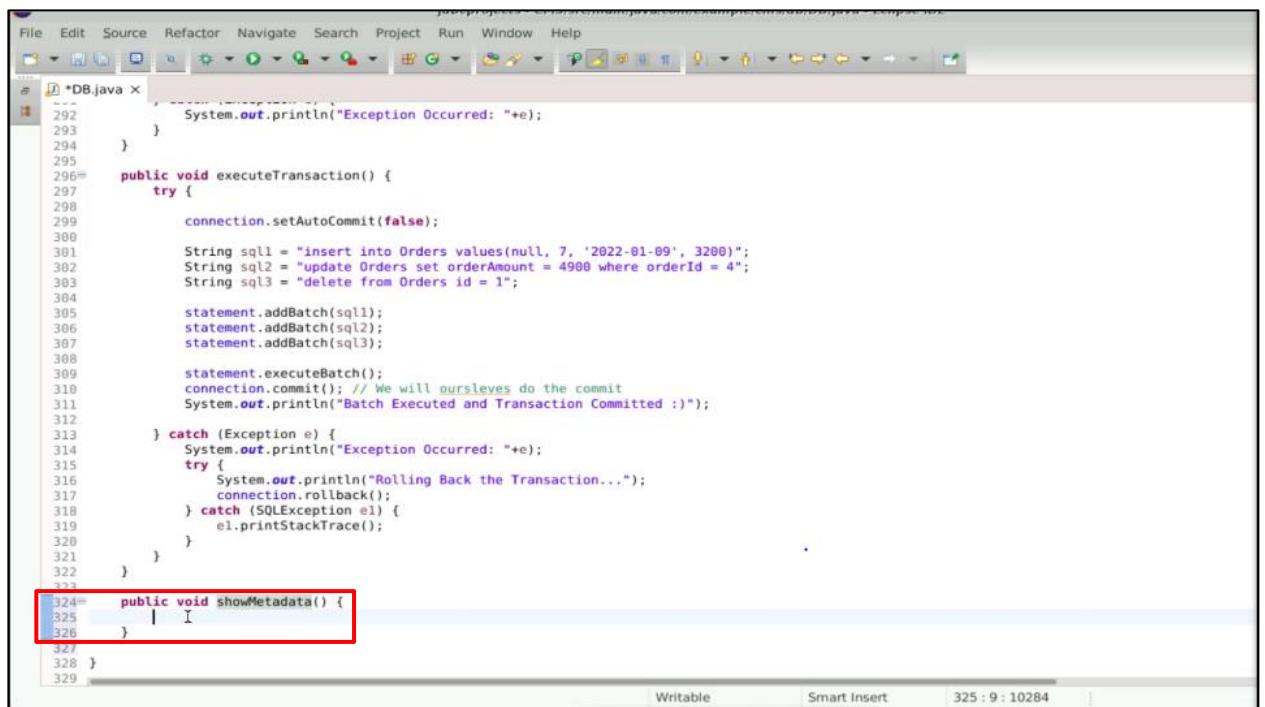


```

1 package com.example.cms.db;
2
3 import java.sql.CallableStatement;
4
5 /**
6 * JDBC Procedure:
7 * 1. Download the Driver and Link it to the Project. For Maven Project simply configure the c
8 * 2. Load the Driver from the library i.e. jar file
9 * 3. Create Connection to the DataBase with url, user and password
10 * 4. Execute CRUD operations
11 * 5. Close the Connection
12 */
13
14 public class DB implements DAO{
15     Connection connection;
16     Statement statement;
17     PreparedStatement preparedStatement;
18     CallableStatement callableStatement;
19 }

```

1.3 Create a new method showMetadata()



```

1 package com.example.cms.db;
2
3 import java.sql.Connection;
4 import java.sql.PreparedStatement;
5 import java.sql.ResultSet;
6 import java.sql.SQLException;
7 import java.sql.Statement;
8 import java.util.ArrayList;
9 import java.util.List;
10
11 public class DB implements DAO{
12     Connection connection;
13     Statement statement;
14     PreparedStatement preparedStatement;
15     CallableStatement callableStatement;
16
17     public void executeTransaction() {
18         try {
19             connection.setAutoCommit(false);
20
21             String sql1 = "insert into Orders values(null, 7, '2022-01-09', 3200)";
22             String sql2 = "update Orders set orderAmount = 4900 where orderId = 4";
23             String sql3 = "delete from Orders id = 1";
24
25             statement.addBatch(sql1);
26             statement.addBatch(sql2);
27             statement.addBatch(sql3);
28
29             statement.executeBatch();
30             connection.commit(); // We will ourselves do the commit
31             System.out.println("Batch Executed and Transaction Committed :)");
32
33         } catch (Exception e) {
34             System.out.println("Exception Occurred: "+e);
35             try {
36                 System.out.println("Rolling Back the Transaction...");
37                 connection.rollback();
38             } catch (SQLException e1) {
39                 e1.printStackTrace();
40             }
41         }
42     }
43
44     public void showMetadata() {
45     }
46 }

```

1.4 Write the **try-catch** block to work on the object **DatabaseMetaData** object which will provide the database information by using the **getMetaData** method

The screenshot shows the Eclipse IDE interface with the title bar "Applications - jdbcprojects - CMS/src/..." and "jdbcprojects - CMS/src/main/java/com/example/cms/db/DB.java - Eclipse IDE". The menu bar includes File, Edit, Source, Refactor, Navigate, Search, Project, Run, Window, Help. The toolbar has icons for New, Open, Save, Cut, Copy, Paste, Find, etc. The code editor displays Java code for a class named DB.java. The code handles database transactions and metadata retrieval. A red box highlights the section from line 325 to line 336, which contains the method showMetadata().

```
File Edit Source Refactor Navigate Search Project Run Window Help

# DB.java x
~ 299
300     connection.setAutoCommit(false);
301
302     String sql1 = "insert into Orders values(null, 7, '2022-01-09', 3200)";
303     String sql2 = "update Orders set orderAmount = 4900 where orderId = 4";
304     String sql3 = "delete from Orders id = 1";
305
306     statement.addBatch(sql1);
307     statement.addBatch(sql2);
308     statement.addBatch(sql3);
309
310     statement.executeBatch();
311     connection.commit(); // We will ourselves do the commit
312     System.out.println("Batch Executed and Transaction Committed :)");
313
314 } catch (Exception e) {
315     System.out.println("Exception Occurred: "+e);
316     try {
317         System.out.println("Rolling Back the Transaction...");
318         connection.rollback();
319     } catch (SQLException e1) {
320         e1.printStackTrace();
321     }
322 }
323 }

325= public void showMetadata() {
326     try {
327
328         DatabaseMetaData dbMetaData = connection.getMetaData();
329         System.out.println("Product Name: "+dbMetaData.getDatabaseProductName());
330         System.out.println("Version: "+dbMetaData.getDatabaseProductVersion()+" Minor: "+dbMetaData.getDatabaseMinorVersion()+" Major: "+dbMetaData.getDatabaseMajorVersion());
331         System.out.println("Driver Version: "+dbMetaData.getDriverVersion());
332         System.out.println("Driver Name: "+dbMetaData.getDriverName());
333         System.out.println("UserName: "+dbMetaData.getUserName());
334
335     } catch (Exception e) {
336         System.out.println("Exception Occurred: "+e);
337     }
338 }
```

1.5 Run the **showMetadata** method from **App.java** by adding the below line in the **App.java** file:

```
db.showMetadata()
```

```
32 //db.deleteCustomer(3);
33
34 //System.out.println();
35
36 ArrayList<Customer> customers = db.getAllCustomers();
37 customers.forEach( cRef -> System.out.println(cRef));
38
39 db.closeConnection();/*
40
41 //Scanner scanner = new Scanner(System.in);
42 //System.out.println("Enter Name: ");
43 //String name = scanner.nextLine();
44
45 //System.out.println("Enter Password: ");
46 //String password = scanner.nextLine();
47
48 //System.out.println("Enter Customer ID:");
49 //int cid = scanner.nextInt();
50
51 //scanner.close();
52
53
54
55 DB db = new DB();
56 db.createConnection();
57 //db.executeProcedure(name, password);
58 //db.executeProcedure(cid);
59
60
61 //db.executeSQLStatementsInBatch();
62
63 //db.beginTransaction();
64 db.showMetadata();
65
66 db.closeConnection();
67
68 }
69 }
```

1.6 Run the code. In the output, see all the attributes, like Product Name and Version printed using the try-catch block.

```
<terminated> App [Java Application] /usr/eclipse/plugins/org.eclipse.jstj.openjdk.hotspot.jre.full.linux.x86_64_16.0.2.v20210721-1149/re/bin/java (Jan 11, 2022, 10:34:42 AM - [DB] Driver Loaded [DB] Connection Created
Product Name: MySQL
Version: 8.0.27-Ubuntu0.20.04.1 Minor: 0 Major: 8
Driver Version: mysql-connector-java-8.0.27 (Revision: e920b979015ae7117d60d72bcc8f077a839cd791)
Driver Name:MySQL Connector/J
UserName: john@localhost
[DB] Connection Closed. Close Status: true
```

1.7 Log in into **mysql** by opening the terminal

```
erishant@gmail@ip-172-31-17-157:~$ mysql -u john -p
Enter password:
```

1.8 Type the command **use estore;** to change the database

```
erishant@gmail@ip-172-31-17-157:~$ mysql -u john -p
Enter password:
Welcome to the MySQL monitor. Commands end with ; or \g.
Your MySQL connection id is 20
Server version: 8.0.27-Ubuntu0.20.04.1 (Ubuntu)

Copyright (c) 2000, 2021, Oracle and/or its affiliates.

Oracle is a registered trademark of Oracle Corporation and/or its
affiliates. Other names may be trademarks of their respective
owners.

Type 'help;' or '\h' for help. Type '\c' to clear the current input statement.

mysql> use estore;
```

1.9 Write the **show tables;** command to list the tables in the **estore** database

Terminal Output:

```

erishant@gmail@ip-172-31-17-157:~$ mysql -u john -p
Enter password:
Welcome to the MySQL monitor.  Commands end with ; or \g.
Your MySQL connection id is 20
Server version: 8.0.27-0ubuntu0.20.04.1 (Ubuntu)

Copyright (c) 2000, 2021, Oracle and/or its affiliates.

Oracle is a registered trademark of Oracle Corporation and/or its
affiliates. Other names may be trademarks of their respective
owners.

Type 'help;' or '\h' for help. Type '\c' to clear the current input statement.

mysql> use estore;
Reading table information for completion of table and column names
You can turn off this feature to get a quicker startup with -A

Database changed
mysql> show tables;

```

Eclipse IDE (right side):

```

getAllCustomers();
ut.println(cRef);

tem.in);
);

d: ");
e());
r ID:");

```

<terminated> App [Java Application] /usr/eclipse/plugins/org.eclipse.jstj.openjdk.hotspot.jre.full.linux.x86_64_16.0.2.v
[DB] Driver Loaded
[DB] Connection Created
Product Name: MySQL
Version: 8.0.27-0ubuntu0.20.04.1 Minor: 0 Major: 8

Terminal Output:

```

erishant@gmail@ip-172-31-17-157:~$ 
File Edit View Search Terminal Help

Oracle is a registered trademark of Oracle Corporation and/or its
affiliates. Other names may be trademarks of their respective
owners.

Type 'help;' or '\h' for help. Type '\c' to clear the current input statement.

mysql> use estore;
Reading table information for completion of table and column names
You can turn off this feature to get a quicker startup with -A

Database changed
mysql> show tables;
+-----+
| Tables_in_estore |
+-----+
| Customer          |
| Employees         |
| Orders            |
| User              |
+-----+
4 rows in set (0.00 sec)

mysql> 

```

Eclipse IDE (right side):

```

getAllCustomers();
ut.println(cRef);

tem.in);
);

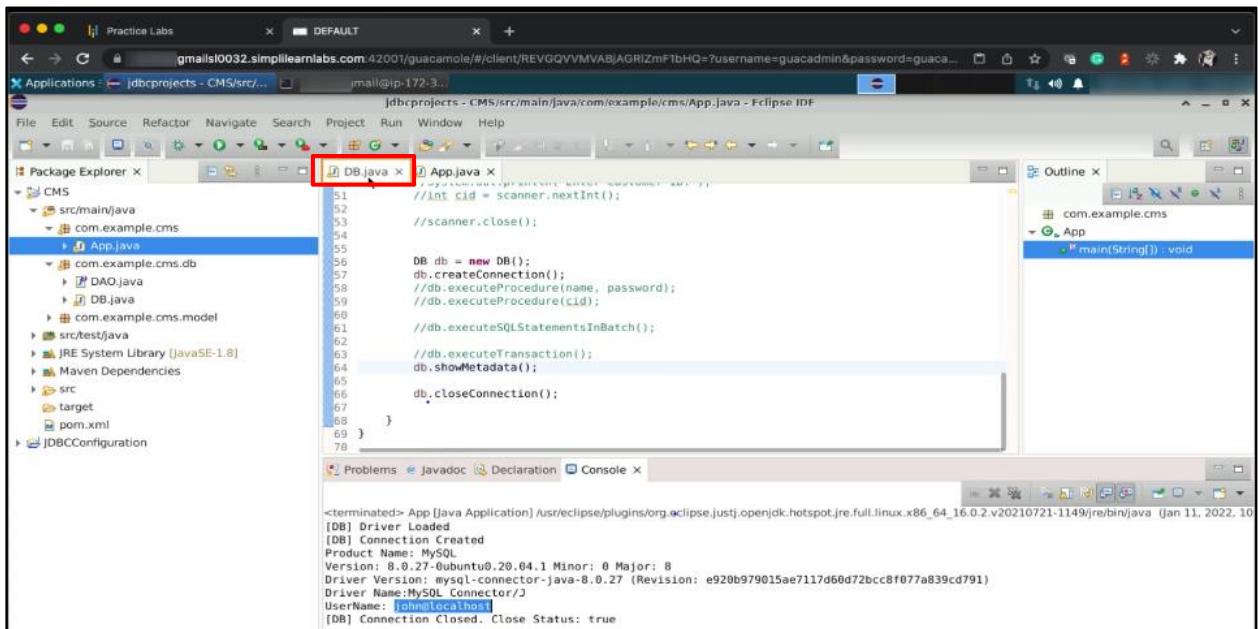
d: ");
e());
r ID:");

```

<terminated> App [Java Application] /usr/eclipse/plugins/org.eclipse.jstj.openjdk.hotspot.jre.full.linux.x86_64_16.0.2.v
[DB] Driver Loaded
[DB] Connection Created
Product Name: MySQL
Version: 8.0.27-0ubuntu0.20.04.1 Minor: 0 Major: 8

Step 2: Extract Metadata containing the list of tables

2.1 Open Eclipse IDE and go to the **DB.java** file to see the metadata containing the list of tables



```
51     //int cid = scanner.nextInt();
52
53     //scanner.close();
54
55     DB db = new DB();
56     db.createConnection();
57     //db.executeProcedure(name, password);
58     //db.executeProcedure(cid);
59
60     //db.executeUpdateStatementsInBatch();
61
62     //db.beginTransaction();
63     db.showMetadata();
64
65     db.closeConnection();
66
67
68 }
69 }
```

<terminated> App [Java Application] /usr/eclipse/plugins/org.eclipse.justj.openjdk.hotspot.jre.full.linux.x86_64_16.0.2.v20210721-1149/jre/bin/java (Jan 11, 2022, 10:00:00 AM)
[DB] Driver Loaded
[DB] Connection Created
Product Name: MySQL Connector/J
Version: 8.0.27-0ubuntu0.20.04.1 Minor: 0 Major: 8
Driver Version: mysql-connector-java-8.0.27 (Revision: e920b979015ae7117d68d72bcc8f077a839cd791)
Driver Name:MySQL Connector/J
UserName: guacamole
[DB] Connection Closed. Close Status: true

2.2 Create a projection array for extracting a table

The screenshot shows the Eclipse IDE interface with the code editor open. The code in DB.java contains a method showMetadata() which prints database metadata. A line of code is highlighted with a red box: `String projectoin[] = {"TABLE"};`. The code editor has tabs for Problems, Javadoc, Declaration, and Console. The Console tab shows the output of the application running, including driver information and connection details.

```

public void showMetadata() {
    try {
        DatabaseMetaData dbMetaData = connection.getMetaData();
        System.out.println("Product Name: " + dbMetaData.getDatabaseProductName());
        System.out.println("Version: " + dbMetaData.getDatabaseProductVersion() + " Minor: " +
        System.out.println("Driver Version: " + dbMetaData.getDriverVersion());
        System.out.println("Driver Name: " + dbMetaData.getDriverName());
        System.out.println("UserName: " + dbMetaData.getUserName());
    } catch (Exception e) {
        System.out.println("Exception Occurred: " + e);
    }
}

```

<terminated> App [Java Application] /usr/eclipse/plugins/org.eclipse.justj.openjdk.hotspot.jre.full.linux.x86_64_16.0.2.v20210721-1149/jre/bin/java (Jan 11, 2022, 10:27:27 AM) [DB] Driver Loaded [DB] Connection Created Product Name: MySQL Version: 8.0.27-Ubuntu0.20.04.1 Minor: 0 Major: 8 Driver Version: mysql-connector-java-8.0.27 (Revision: e920b979015ae7117d60d72bcc8f077a839cd791) Driver Name: MySQL Connector/J UserName: john@localhost [DB] Connection Closed. Close Status: true

2.3 Create a result set using the ResultSet object to filter the data

The screenshot shows the Eclipse IDE interface with the code editor open. The code in DB.java contains a modified version of the showMetadata() method. A line of code is highlighted with a red box: `ResultSet set = dbMetaData.getTables(null, null, null, projectoin);`. The code editor has tabs for Problems, Javadoc, Declaration, and Console. The Console tab shows the output of the application running, including driver information and connection details.

```

public void showMetadata() {
    try {
        DatabaseMetaData dbMetaData = connection.getMetaData();
        System.out.println("Product Name: " + dbMetaData.getDatabaseProductName());
        System.out.println("Version: " + dbMetaData.getDatabaseProductVersion() + " Minor: " +
        System.out.println("Driver Version: " + dbMetaData.getDriverVersion());
        System.out.println("Driver Name: " + dbMetaData.getDriverName());
        System.out.println("UserName: " + dbMetaData.getUserName());
    } catch (Exception e) {
        System.out.println("Exception Occurred: " + e);
    }
}

```

<terminated> App [Java Application] /usr/eclipse/plugins/org.eclipse.justj.openjdk.hotspot.jre.full.linux.x86_64_16.0.2.v20210721-1149/jre/bin/java (Jan 11, 2022, 10:27:27 AM) [DB] Driver Loaded [DB] Connection Created Product Name: MySQL Version: 8.0.27-Ubuntu0.20.04.1 Minor: 0 Major: 8 Driver Version: mysql-connector-java-8.0.27 (Revision: e920b979015ae7117d60d72bcc8f077a839cd791)

2.4 Loop through the result set and add the print statement

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

- Project Explorer:** Shows files like DB.java, App.java, db, model, and JavaSE-1.8.
- DB.java (Content View):** Displays Java code for a DatabaseMetaData object. A red box highlights the following code block:

```
System.out.println("Listing Tables: ");
String projectoin[] = {"TABLE"};
ResultSet set = dbMetaData.getTables(null, null, null, projectoin);
while(set.next()) {
    System.out.println(set.getString("TABLE_NAME"))
}
```
- Outline View:** Shows the class structure with methods like callableStatement, TAG, DB(), createConnection(), closeConnection(), createCustomer(), updateCustomer(), deleteCustomer(), getAllCustomers(), executeProcedure(), executeProcedure(int), executeSQLStatementsInBatch(), executeTransaction(), and showMetadata().
- Console View:** Displays the output of the executed Java application:

```
<terminated> App [Java Application] /usr/eclipse/plugins/org.eclipse.justj.openjdk.hotspot.jre.full.linux.x86_64_16.0.2.v20210721-1149/jre/bin/java (Jan 11, 2022, 10:26:27 AM)
[DB] Driver Loaded
[DB] Connection Created
Product Name: MySQL
Version: 8.0.27-0ubuntu0.20.04.1 Minor: 0 Major: 8
Driver Version: mysql-connector-java-8.0.27 (Revision: e920b979015ae7117d60d72bcc8f077a839cd791)
Driver Name:MySQL Connector/J
UserName: john@localhost
[DB] Connection Closed. Close Status: true
```

2.5 Run the program from the **App.java** file. See the output similar to the **show tables;** command in the console.

The screenshot shows the Eclipse IDE interface. In the top-left, there's a code editor with two tabs: DB.java and App.java. The App.java tab contains Java code that interacts with a MySQL database. The code includes creating a connection, executing procedures, and listing tables. In the bottom-right, there's a terminal window titled 'Console' showing the output of the program. The output displays database product information and a list of tables: Customer, Employees, Orders, and User. The 'Customer' table name is highlighted with a red box.

```
DB.java
App.java X
50
51
52
53
54
55
56
57
58
59
60
61
62
63
64
65
66
67
68
69
70

db
model
avaSE-1.8]

<terminated> App [Java Application] /usr/eclipse/plugins/org.eclipse.justj.openjdk.hotspot.jre.full.linux.x86_64_16.0.2.v20210721-1149/jre/bin/java (Jan 11, 2022, 10
Product Name: MySQL
Version: 8.0.27-Ubuntu0.20.04.1 Minor: 0 Major: 8
Driver Version: mysql-connector-java-8.0.27 (Revision: e920b979015ae7117d60d72bcc8f077a839cd791)
Driver Name:MySQL Connector/J
UserName:john@localhost
Listing Tables:
Customer
Employees
Orders
User
[DB] Connection Closed. Close Status: true
```

By following these steps, you have successfully extracted information about the database in a Java program using metadata for understanding the database structure and schema. This approach allows for dynamic retrieval of database details such as tables, columns, and data types.